Unpaired Image Translation via Vector Symbolic Architectures

Justin Theiss[1,2(✉)], Jay Leverett[1], Daeil Kim[1], and Aayush Prakash[1]

[1] Meta Reality Labs, Burlingame, USA
{theiss,jayleverett,daeilkim,aayushp}@fb.com
[2] University of California, Berkeley, CA 94720, USA

Abstract. Image-to-image translation has played an important role in enabling synthetic data for computer vision. However, if the source and target domains have a large semantic mismatch, existing techniques often suffer from source content corruption aka semantic flipping. To address this problem, we propose a new paradigm for image-to-image translation using Vector Symbolic Architectures (VSA), a theoretical framework which defines algebraic operations in a high-dimensional vector (hypervector) space. We introduce VSA-based constraints on adversarial learning for source-to-target translations by learning a hypervector mapping that inverts the translation to ensure consistency with source content. We show both qualitatively and quantitatively that our method improves over other state-of-the-art techniques.

Keywords: Image-to-image translation · Adversarial learning · Vector symbolic architectures · Semantic flipping

1 Introduction

Image-to-image translation techniques [8,12,14,23,25,30,33,39] have been instrumental in improving synthetic data for computer vision. They have been used to bridge the domain gap for synthetic data [12,30,39] and for photo-realistic enhancement in virtual reality and gaming applications [33]. Some researchers [7,16,31] argue that the domain gap can be further factorized into a content (shift in semantic statistics) and appearance gap. In this work, we are interested in the unpaired image-to-image translation method in the challenging scenario where the content gap between the source and target domains is large.

There are several techniques for unpaired image-to-image translation [12,30,39]. Some approaches assume that content and style can be separated in order to translate style without corrupting content (e.g., [12]). Others have used a bijective mapping to reconstruct the source image from the translated image (i.e., a "cyclic loss" [39]). However, these methods do not work well if there is a *large*

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-031-19803-8_2.

S. Avidan et al. (Eds.): ECCV 2022, LNCS 13681, pp. 17–32, 2022.
https://doi.org/10.1007/978-3-031-19803-8_2

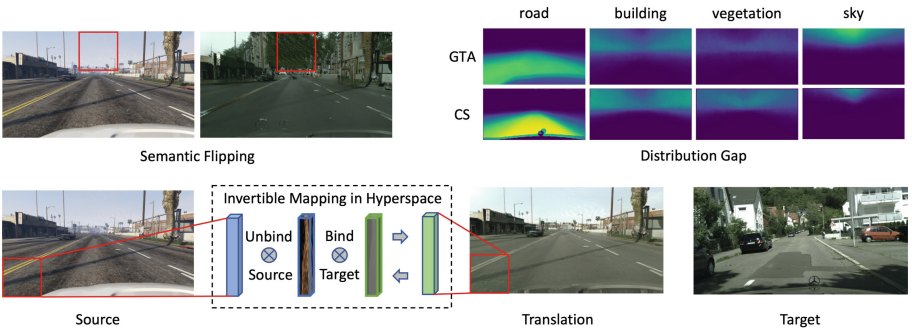

Fig. 1. We propose Vector Symbolic Architecture based image-to-image translation technique (VSAIT) which addresses semantic flipping (source content corruption) that happens when the distribution gap (shift in semantic statistics) between source and target domains is large. Our method learns a mapping in high-dimensional vector space (hyperspace), which encourages translated images to be consistent with the source domain. Conceptually, our approach aims to "unbind" source-related information (e.g., texture and color) and "bind" target-related information as well as vice versa to recover source content. (Color figure online)

shift in distribution between source and target domains leading to the problem of *semantic flipping*. Semantic flipping is characterized by image artifacts, object or feature hallucinations that are a result of adversarial training for datasets with a large content gap. Specifically, this is observed as a change in content between source and translated images (e.g., sky to trees in Fig. 1). Semantic flipping is a critical issue for improving photorealism in computer graphics applications [33] as well as training downstream tasks using translated images, as we want to preserve the source semantic labels of translations.

Relatively few works have directly focused on the semantic flipping problem; however, they can be broadly categorized into three approaches: image-level consistency, domain-invariant encoding, and task-level consistency. Methods using an image-level consistency loss attempt to ensure that pixel-level information is highly correlated between source and translated images [3,8]. Such methods may fail to account for feature-level differences that do not correlate well with pixel-level differences. Approaches that focus on domain-invariant encoding train the generator to encode domain-invariant content either using a shared latent space [23] or contrastive learning [14]. These methods will fail to reduce semantic flipping if content and style cannot be sufficiently disentangled. Finally, others have used pre-trained task networks to generate pseudo-labels for each domain to ensure a consistent translation with respect to source labels (i.e., semantic masks) [11,33]. However, these methods fail if the task network cannot generate pseudo-labels for both domains. We instead focus on a method that provides feature-level consistency between source and translated images without explicit assumptions of domain-invariant encoding or quality of pseudo-labels. This addresses gaps in previous methods which make them vulnerable to semantic flipping.

In the current paper, we propose a novel usage of a theoretical framework known as vector symbolic architectures (VSA [9,15]; also referred to as hyperdimensional computing) as a new paradigm for unpaired image-to-image translation. Although much of the VSA research has been conducted in theoretical neuroscience (see [20,21]), it has more recently been applied to computer vision problems using extracted features from deep neural networks [27,28]. VSA defines a high-dimensional vector (hypervector) space and operators used for symbolic computation, which allows for mathematical formulations of conceptual queries such as, "what color is the car?". In the case of unpaired image translation, such formulations are useful because they enable us to recover attributes from the source image and ensure consistent relationships among features when translating images from one domain to another. VSA is well suited for this approach as it can represent arbitrary symbols and formulations generally without supervision or training [32]. The important difference from previous methods addressing semantic flipping is that VSA ensures that hypervector representations of different semantic content are almost orthogonal to each other (e.g., sky and trees) [15]. The cosine distance between translated and source hypervectors therefore is greatest when semantic flipping occurs.

Using this framework, we propose a method for learning a hypervector mapping between source and target domains that is robust to semantic flipping (Fig. 1). By inverting this mapping, we are able to minimize the distance between features extracted from source and translated images without requiring that content and style be fully disentangled. We demonstrate qualitatively and quantitatively that this approach significantly reduces the image artifacts and hallucinations observed for unpaired image translation between domains with a large content gap. We hope that our work provides inspiration for incorporating VSA into new areas of computer vision research. Our contributions include:

- Our method addresses important artifacts and feature hallucinations (semantic flipping) that often occur with other unsupervised image translation methods as a result of content gap between source and target domains.
- To the best of our knowledge, we are the first to show that Vector Symbolic Architectures (VSA) can be used for a challenging task of image translations in the wild. We hope this opens up an exciting area of research around VSA.
- We demonstrate qualitative and quantitative improvement over state-of-the-art methods that directly address semantic flipping across multiple experiments with unmatched semantic statistics.

2 Related Work

Unpaired Image-to-Image Translation. Unpaired image-to-image translation is a widely studied problem with many existing techniques [2,3,8,12–14,18,23,25,30,33,38,39]. One popular approach is to impose the cycle-consistency constraint [18,38,39], which states that a source-to-target generator and a target-to-source generator are bijections. Building on cycle-consistency, UNIT [25]

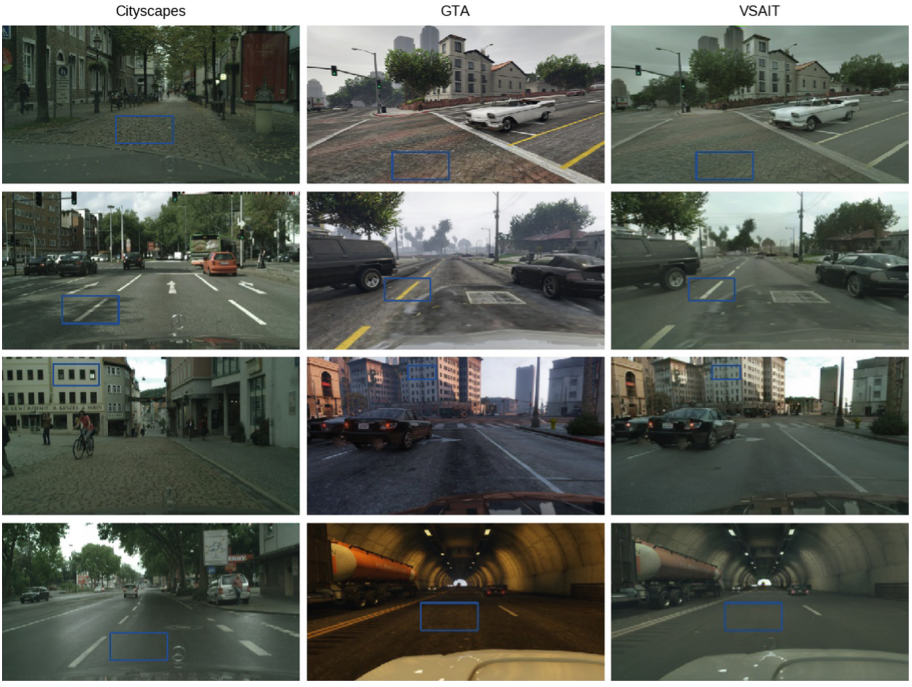

Fig. 2. Example GTA translations from our method (VSAIT) alongside representative examples from Cityscapes. We can see that VSAIT successfully translates GTA images to Cityscapes' style and that, in particular, our method is able to learn specific textures and attributes (e.g. gray cobblestone, white lane lines rather than yellow) that are commonly found in Cityscapes images. Moreover, VSAIT is able to handle unseen scenarios (e.g. tunnel) that do not occur in Cityscapes without semantic flipping. (Color figure online)

mapped the source and target domains to a shared latent space, while DRIT [23] and other methods [2,12] factorized the latent space into a shared content space and a domain-specific style space. DistanceGAN [3] showed that source to target translations can be learnt unidirectionally. CUT [30] subsequently proposed a unidirectional method based on contrastive learning. Whereas some of these methods require separate networks per domain or assume that content and style can be disentangled, our method uses a single generator without assumption of disentanglement.

Semantic Flipping. The body of research on semantic flipping is relatively small. Methods based on cycle-consistency combat semantic flipping through pixel-level reconstruction, but they are unable to prevent semantic flipping in the face of large distributional shifts. GcGAN [8] enforces geometry-consistent constraints on image translations by ensuring robustness for transformations such as flipping or rotation. This does not necessarily address semantic flip-

ping for small, local features that could still be geometry-consistent. EPE [33] addresses semantic flipping by conditioning on G-buffers (e.g. albedo, glossiness, depth, normals) as well as incorporating semantic segmentation pseudo-labels into the discriminator. Although their method addresses semantic flipping, it does not generalize to many tasks or datasets since it assumes access to the synthetic image rendering process and a downstream task network that can generate pseudo-labels for both source and target domains. Alternatively, SRUNIT [14] proposed to address semantic flipping by using contrastive learning (following CUT [30]) to encode domain-invariant semantic representations that are robust to small feature-space perturbations. Although their approach does not require access to any labels or rendering process like EPE, it does not sufficiently address semantic flipping. Unlike these approaches, we reduce semantic flipping by inverting the translation in hyperspace using VSA in order to ensure recovery of source information.

Computer Vision Applications of VSA. Although much of the research associated with VSA has been conducted in theoretical neuroscience, Neubert et al. [28] provided examples of how VSA can be used for computer vision tasks such as object and place recognition, which demonstrated how image features extracted from pre-trained neural networks can be used within the VSA framework. Other works have used VSA for visual question answering [26] and scene transformation [17], albeit with simple shapes or MNIST digits. More recently, Osipov et al. [29] used VSA to learn to unbind category representations among unlabelled data vectors bound to a shared representational space. This is most similar to the challenge that we are addressing in the current paper, as we can consider the source and target features to be bound to a similar content space; however, unlike their study we do not have access to the underlying shared representational space.

3 Method

3.1 Preliminary: VSA

Vector Symbolic Architectures (VSA) [9,15] provide a framework for encoding and manipulating information in a hypervector space (hyperspace) with high capacity and robustness to noise [15]. In VSA framework, hypervectors are randomly sampled from a vector space $\mathbb{V} = [-1, 1]^n$ (where $n >> 1000$) in order to represent symbols that can be typically manipulated using two operators – binding/unbinding (element-wise multiplication) and bundling (element-wise addition). The binding operator can be used to assign an attribute to a symbol (e.g., "gray car"), whereas bundling can be used to group symbols together (e.g., "gray car and red bike"). These operators define the Multiplication-Addition-Permutation (MAP) architecture. Other operators and architectures (see [35]) are beyond the scope of our work.

The benefit of using VSA framework for image translation is its versatility in representing arbitrary symbols without supervision. This allows us to infer

underlying representational spaces (e.g., content distribution) without explicitly defining them. Since the challenge of semantic flipping in unpaired image translation is typically to constrain the generator, we do not need to learn how to generate images from hypervectors but instead learn a mapping that ensures we recover the same source information for a given translated hypervector.

In order to assign random hypervectors to image features, Neubert et al. [28] used locality sensitive hashing (LSH) to project image features extracted using a pre-trained neural network (AlexNet [22]) to a relatively lower-dimensional space (from $13 \times 13 \times 384$ to 8192). In this case, the entire image was encoded into a single hypervector; however, image patches can also be encoded as separate hypervectors [27]. As an example, imagine an image patch from the source domain that contains a car and pedestrian. We can assume without loss of generality that the image patch is represented as a hypervector:

$$v_{src} = c \otimes c_{src} + p \otimes p_{src}, \tag{1}$$

where c and p are hypervectors representing the car and pedestrian bound with source-specific hypervectors c_{src} and p_{src}, respectively.

Since we do not have access to the underlying symbols associated with source and target domains, instead we must learn a mapping that unbinds source-specific information and binds target-specific information (and vice versa). Specifically, we would like to learn a hypervector mapping that can unbind these source-specific attributes and bind target-specific attributes in order to obtain a target representation v_{tgt} as shown in Eq. 2.

$$\begin{aligned} &v_{src} \otimes u_{src \leftrightarrow tgt} \approx v_{tgt}, \\ &\text{where } u_{src \leftrightarrow tgt} = (c_{src} \otimes c_{tgt} + p_{src} \otimes p_{tgt}) \end{aligned} \tag{2}$$

Since the binding/unbinding operation is invertible, c_{src} and p_{src} are unbound while c_{tgt} and p_{tgt} are correspondingly bound. Importantly, since these random hypervectors are very likely to be almost orthogonal, binding/unbinding the incorrect attributes (i.e., $c \otimes p_{src}$) simply adds noise to the resulting vector as demonstrated in Eq. 3.

$$\begin{aligned} v_{src} \otimes u_{src \leftrightarrow tgt} &= c \otimes c_{src} \otimes (c_{src} \otimes c_{tgt} + p_{src} \otimes p_{tgt}) \\ &\quad + p \otimes p_{src} \otimes (c_{src} \otimes c_{tgt} + p_{src} \otimes p_{tgt}) \\ &= (c \otimes c_{tgt} + noise) + (noise + p \otimes p_{tgt}) \\ &= c \otimes c_{tgt} + p \otimes p_{tgt} + noise \approx v_{tgt} \end{aligned} \tag{3}$$

It is also worth noting that $u_{src \leftrightarrow tgt}$ is equivalent to $u_{tgt \leftrightarrow src}$, which means a single vector can be used to map between the domains. Hence, $v_{tgt} \otimes u_{src \leftrightarrow tgt} \approx v_{src}$. Note that hypervectors in the VSA framework are distributed representations that are very robust to noise [1] and further that cosine similarity will still be high relative to similarity with random vectors. The above example demonstrates the unique properties of the VSA framework, which allow for algebraic formulations of symbolic representations.

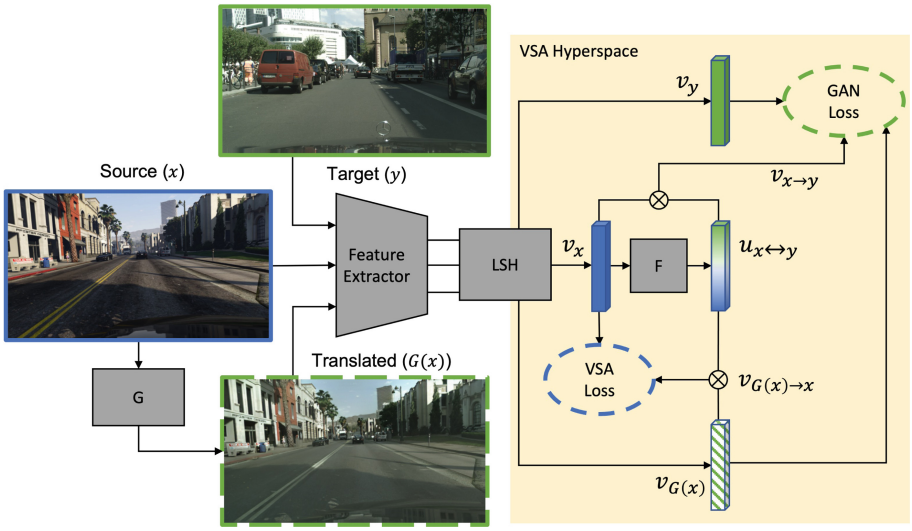

Fig. 3. Our method addresses semantic flipping by learning an invertible mapping in a high-dimensional vector space (VSA hyperspace) in order to ensure consistency between source and translated images. We extract features and use locality sensitive hashing (LSH) to encode source (x, blue), target (y, green), and translated ($G(x)$, striped green) images into this random VSA hyperspace. We use a GAN loss to train G to generate images with hypervectors similar to those in the target domain. Finally, the hypervector mapping ($u_{x\leftrightarrow y}$, green-and-blue vertical bar) is used to invert our translation (Eq. 4) to recover the source hypervectors (VSA Loss). See Sect. 3.3 for more details. (Color figure online)

3.2 VSA-Based Image Translation (VSAIT)

Overview. Our goal is to translate images from a source to target domain, while minimizing semantic flipping caused by differences in the content distribution between the two domains. As shown in Fig. 3, our method uses a hypervector adversarial loss (GAN Loss in Fig. 3) that operates on VSA-based hypervector representations of image patches. In order to address semantic flipping we constrain the generator to preserve the content of the source domain via a VSA-based cyclic loss (VSA Loss in Fig. 3). We do so by leveraging VSA to invert our translation in hypervector space to ensure consistency with the source content. We will cover these two loss functions later in this section. Our method has three major network components– Generator G, Discriminator D_Y and Mapper F that we will discuss next. We present training details in Sect. 3.3.

Generator. Our approach uses a generator $G : X \to Y$ (shown in Fig. 3) and feature-level discriminator D_Y in order to learn an unpaired image translation between X (source) and Y (target) domains, where D_Y is trained to discriminate between hypervectors extracted from target and translated images.

As previously introduced in Sect. 3.1, to assign hypervectors to image patches, we follow Neubert et al. [28] and extract features using a pre-trained neural network (in our case, concatenated multiple layers of VGG19 [36]). These concatenated features are then randomly projected from the extracted feature vector $f_i \in \mathbb{R}^m$ to the random vector space $\mathbb{V} = [-1, 1]^n$ where $m >> n$. Using this approach, we denote hypervectors extracted from target images as v_Y, source images as v_X, and translated images as $v_{G(X)}$ (see Fig. 3).

Source ↔ Target Mapper. Furthermore, we train a network F to generate a hypervector mapping $u_{X \leftrightarrow Y}$ (e.g., Eq. 3) between source and target domains. We use this hypervector mapping to invert our translation, providing a VSA-based cyclic loss in hyperspace. The invertible mapping accomplishes two operations with a single step: unbinding of source (resp. target) representations and binding of target (resp. source) representations. This means that if we apply this mapping to our translated hypervectors $v_{G(X)}$, we should approximately obtain the original source hypervectors v_X. Similarly, if we apply this mapping to source hypervectors v_X, we approximately obtain translated hypervectors $v_{G(X)}$. We denote translated hypervectors that have been mapped to the source domain as $v_{G(X)\to X}$ and those mapped from source to target domain as $v_{X\to Y}$ as shown in Eq. 4 as well as Fig. 3.

$$\begin{aligned} v_{G(X)} \otimes u_{X \leftrightarrow Y} &= v_{G(X)\to X} \approx v_X \\ v_X \otimes u_{X \leftrightarrow Y} &= v_{X\to Y} \approx v_{G(X)} \end{aligned} \tag{4}$$

Hypervector Adversarial Loss. In order to train G, D_Y and F to translate images that match the target distribution, we use an adversarial loss [10]. Specifically, we want the hypervectors of our translated image $v_{G(X)}$ to be similar to those from the target domain v_Y. Furthermore, by binding the hypervector mapping $u_{X \leftrightarrow Y}$ to source vectors v_X, we should obtain a hypervector of the source features mapped to the target domain (Eq. 4). Therefore, we train F along with G and D_Y using the following adversarial loss:

$$\begin{aligned} \mathcal{L}_{GAN}(G, D_Y, F, X, Y) = &\mathbb{E}_{y \sim p_Y(y)}[\log D_Y(v_y)] \\ &+ \mathbb{E}_{x \sim p_X(x)}[\log(1 - D_Y(v_{G(x)})] \\ &+ \mathbb{E}_{x \sim p_X(x)}[\log(1 - D_Y(v_x \otimes F(v_x))], \end{aligned} \tag{5}$$

where G and F networks are trained to fool the discriminator D_Y, thereby minimizing the objective while D_Y attempts to maximize it.

VSA-Based Cyclic Loss. Although adversarial training provides a method to generate images that match the target distribution, the differences in content between domains will result in semantic flipping. We therefore need a loss that constrains the generator to preserve source content and reduce semantic flipping. To do so, we incorporate our VSA-based cyclic loss, which ensures that the same hypervectors are obtained when mapping translated vectors back from target to source domain ($v_{G(X)\to X}$):

$$\mathcal{L}_{VSA}(G, X) = \mathbb{E}_{x \sim p_X(x)} \left[\frac{1}{n} \sum_{i=1}^{n} dist \left(v_x^i, v_{G(x) \to x}^i \right) \right], \tag{6}$$

where $dist(\cdot, \cdot)$ is the cosine distance averaged across the n hypervectors (representing image patches indexed by i) for the image x. By minimizing the cosine distance ($1 -$ cosine similarity) between $v_{G(X) \to X}$ and v_X, we ensure that the same features are recovered after inverting our translation (i.e., representations of "car" and "pedestrian" are maintained in the example from Eq. 1).

Our overall objective combines our adversarial and VSA-based cyclic losses, training G to generate images matching the target domain and F to invert the translation to ensure consistency with the source domain:

$$\mathcal{L}(G, D_Y, F, X, Y) = \mathcal{L}_{GAN}(G, D_Y, F, X, Y) + \lambda \mathcal{L}_{VSA}(G, X), \tag{7}$$

where λ controls the relative importance of our VSA-based cyclic loss.

3.3 VSAIT Training

We train our method with a GAN-based training framework using the objective described in Eq. 7. We begin by randomly sampling an image from each domain and translating the source image into the target domain via the generator G. In order to use the VSA framework, we must first encode data into the hypervector space. Therefore, we extract features from each image (source, target, and translated) at multiple layers of a pre-trained neural network (Feature Extractor in Fig. 3) as the first step in encoding image patches into the VSA hyperspace. We consider each image patch to be represented by the extracted features sharing its receptive field. We flatten and concatenate these features across layers, resulting in a set of feature vectors (one per image patch) with a dimensionality of m.

We then use LSH (Fig. 3) to reduce the dimensionality of the extracted feature vectors into our random hyperspace $\mathbb{V}$, which reduces the dimensionality to $n << m$. Specifically, we normalize and project the extracted feature vectors using a random matrix where each row is sampled from a standard normal distribution and normalized to unit length. The resulting vectors are therefore in the range $[-1, 1]$, which is necessary to implement the VSA binding operation as described in Sect. 3.1. These hypervectors represent the image patches for the source, target, and translated images as shown in Fig. 3. Note that this process does not require training nor is it necessarily dependent on a specific feature extractor, although different feature extractors may encode different information.

Finally, we use F to generate a hypervector mapping for each source hypervector. We use these hypervector mappings ($u_{x \leftrightarrow y}$ in Fig. 3) to unbind source (resp. target) information and correspondingly bind target (resp. source) information (Eq. 4). By applying this mapping to the source hypervectors, we should obtain a hypervector representation of the source image translated into the target domain ($v_{x \to y}$ in Fig. 3), which is used in our adversarial loss (Eq. 5) to

train F. The ultimate goal of this step, however, is to use the mapping to invert our translation $v_{G(x)}$ and ensure that we recover the same source information v_x (Eq. 6). Doing so will reduce semantic flipping by constraining G to generate images that have hypervectors that are consistent with the source domain.

4 Experiments

We evaluate our method using experiments across multiple datasets. First, we compare against baseline techniques for GTA [34] to Cityscapes [6], where the two domains have inherent semantic differences. We then perform experiments using datasets sub-sampled to create differences in semantic statistics. Specifically, we follow the method in [14] to sub-sample the Google Maps [13] dataset, which is typically designed for paired image translation tasks. We show that VSAIT works as intended to reduce semantic flipping while still generating diverse image translations. Our qualitative results in Fig. 2 demonstrate that the hyperspace mapping constrains the generator to translate features that are consistent between source and target domains. Furthermore, we show that other methods still have significant artifacts and hallucinations related to semantic flipping (Fig. 4). Our quantitative results show improvements against the other baselines, particularly for the GTA to Cityscapes task. Finally, we perform an ablation study to evaluate the contributions of different components in VSAIT.

4.1 Implementation Details

We follow CUT [30] in our choice of the generator network architecture. For the discriminator network, we use a three-layer fully-convolutional network with 1×1 convolutional filters. For the mapping network F, we use a two-layer fully-convolutional network. We train VSAIT using the Adam optimizer [19] ($\beta_1 = 0$, $\beta_2 = 0.9$) with a batch size of 1 and learning rate of 0.0001 for the generator (and mapping network F) and 0.0004 for the discriminator. To improve adversarial training, we use the "hinge loss" [24] rather than the negative log-likelihood objective in $\mathcal{L}_{GAN}$ (Eq. 5). For GTA to Cityscapes, we use $\lambda = 10$ in Eq. 7 and $\lambda = 5$ for other experiments. See Supplemental for more details.

4.2 Datasets

We perform our experiments using three datasets: GTA [34], Cityscapes [6], and Google Maps [13]. GTA [34] is a synthetic dataset of 24966 images generated from the video game Grand Theft Auto V. Cityscapes [6] is a real dataset of driving scenarios accompanied with finely-annotated semantic segmentation labels, which includes 2975 training and 500 validation images. The Google Maps dataset [13] is a collection of 2194 paired maps and aerial photos (1096 training and 1098 validation images).

4.3 Baselines

We compare our method against several baselines that are relevant to the semantic flipping problem: GcGAN, DRIT, CUT, SRUNIT, and EPE. We choose these methods for comparison as they reflect the recent approaches focused on reducing semantic flipping. Specifically, they represent methods that use image-level consistency (GcGAN [8]), domain-invariant encoding (DRIT [23], CUT [30], SRUNIT [14]), and task-level consistency (EPE [33]). It is worth reiterating that EPE only works for synthetic datasets with access to G-buffers (e.g., albedo, glossiness, depth, normals), which are not publicly available for GTA and do not exist for real datasets (e.g., Google Maps). Therefore we compare qualitatively to EPE and rely on quantitative values reported in their work.

4.4 GTA → Cityscapes

We first demonstrate our method using GTA to Cityscapes, which are unpaired datasets that naturally differ in their semantic statistics. For training, we use 20000 of GTA images for our source dataset and all 2975 training images for our target dataset. Following [33], we evaluate our image translation performance using the Kernel Inception Distance (KID) [4] which measures the distance of extracted features from translated and target images using the pre-trained InceptionV3 network [37]. As seen in the Table 2, our method outperforms the baseline methods in the KID metric. As shown in Fig. 4, only EPE has similar quality of image translations for GTA to Cityscapes. However, whereas other methods hallucinate trees and Mercedes hood ornaments, EPE has a tendency to remove palm trees and add unnatural lighting to cars even in dark scenes.

We additionally evaluate translation quality via semantic segmentation metrics computed using DeepLab V3 [5] pretrained on Cityscapes. In line with [14], we report three metrics related to semantic segmentation performance (Table 1): pixel accuracy, class accuracy, and mean intersection over union (mIoU). As shown in Table 1, we outperform the other baselines by a wide margin. Whereas KID computes the distance between translated and target images in feature space, semantic segmentation metrics better reflect semantic flipping directly. As seen in Fig. 4, other methods often generate trees and other features in the sky, which will lower semantic segmentation performance in those regions.

4.5 Google Map → Aerial Photo

We then demonstrate performance on the Google Maps [13] dataset. Here we sub-sample the 1096 training images using K-means clustering of the histograms from grayscale map images to obtain two datasets with different semantic statistics (as in [14]). We evaluate translation performance on all 1098 validation images and report three pixel-level metrics (Table 1). The first metric is the L2 distance between translated and target images. We also report pixel accuracy defined as the percentage of pixels with a maximum absolute difference less than a given threshold (i.e., $max(|r_i - r'_i|, |g_i - g'_i|, |b_i - b'_i|) < \delta$). For the map

Fig. 4. We compare our VSAIT translation to those of baseline methods on an example GTA image. Typical methods often suffer from semantic flipping in open sky regions of GTA, as such regions are observed less often in Cityscapes (i.e., content gap). Here we see that SRUNIT [14], CUT [30], GcGAN [8] and DRIT [23] each suffer from sky hallucinations, while EPE [33] removes palm trees (features that are absent from Cityscapes) in the distance. Meanwhile, our method is able to translate to Cityscapes style while preserving semantics from the original GTA image.

to aerial photo task, we use thresholds of $\delta = 30, 50$ [14]. Similar to GTA to Cityscapes, our approach substantially outperforms the baseline methods. Particularly interesting is the improvement we see in the pixel accuracy for both thresholds ($\delta = 30, 50$), whereas the other methods perform very similarly.

Table 1. Quantitative evaluation across datasets with unmatched semantics. The metrics included are average pixel prediction accuracy (pxAcc), average class prediction accuracy (clsAcc), mean IoU (mIoU), average L2 distance (Dist), and pixel accuracy with task-specific thresholds (Acc).

	GTA → Cityscapes			Map → Photo			Photo → Map		
Method	pxAcc	clsAcc	mIoU	Dist	Acc (δ_1)	Acc (δ_2)	Dist	Acc (δ_1)	Acc (δ_2)
GcGAN [8]	65.62	32.38	22.64	71.47	28.87	43.48	23.62	15.00	30.65
DRIT [23]	64.28	32.17	20.99	70.87	28.97	43.56	24.19	13.94	29.01
CUT [30]	64.59	32.19	20.35	70.28	28.86	44.07	23.44	16.25	31.34
SRUNIT [14]	67.21	32.97	22.69	68.55	30.41	45.91	23.00	**17.67**	**32.78**
VSAIT (ours)	**76.48**	**45.33**	**30.89**	**64.98**	**45.3**	**69.28**	**22.86**	15.2	32.13

4.6 Aerial Photo → Google Map

We additionally demonstrate performance for the task of photo to map using the same sub-sampled datasets obtained from the Google Maps training dataset as described above. For this experiment, we evaluate the same metrics as with map to photo but with pixel accuracy threshold of $\delta = 3, 5$ [14] (Table 1). Although we outperform the baseline methods on the L2 distance metric (Dist in Table 1), we did not observe improvements in the pixel accuracy metrics. We do believe that our approach using VSA is capable of improving these metrics with a more suitable encoding method (as opposed to VGG) for images that have regions without contours or complex features (as is the case for some regions in Google Map images representing landscape or water). However, we leave study of encoding methods for VSA in computer vision applications to future research.

4.7 Ablation Study

We demonstrate the effect of VSAIT by evaluating the contribution of the VSA-based cyclic loss (Eq. 6), the learned hypervector mapping $u_{X \leftrightarrow Y}$, and the hypervector dimensionality. We use the GTA to Cityscapes task to evaluate these ablations on semantic segmentation performance metrics (Table 2). We first demonstrate that by removing the VSA-based cyclic loss, the generator learns to translate images without preserving content, demonstrating the serious problem of semantic flipping (Fig. 5B). Next, we show the importance of the invertible mapping $u_{X \leftrightarrow Y}$ by generating a random hypervector mapping instead of learning the mapping adversarially (Eq. 5). When using a random hypervector instead of the learned mapping generated by F, the generator maintains global structure but since the invertible mapping does not recover the source hypervector (Eq. 4) the local content is often changed (semantic flipping shown in Fig. 5C). Finally, we demonstrate the importance of using the VSA hypervector space by reducing the dimensionality of $\mathbb{V}$ from 4096 to 128. Reducing the dimensionality of the hyperspace from 4096 to 128 results in noisy image translations that seem to only reflect global changes in style. However, even using a low-dimensional hypervector VSAIT still outperforms most methods compared in Table 1.

Table 2. Evaluations on GTA to Cityscapes. Left: quantitative comparison for KID metric. *Mean values reported in [33]. Right: ablation study task demonstrating the contributions for each component of VSAIT method.

Method	KID
CUT	21.18*
SRUNIT	16.27
DRIT	14.69
GcGAN	12.32
EPE	10.95*
VSAIT (ours)	**8.74**

Ablation	pxAcc	clsAcc	mIoU
w/o VSA Loss	52.17	16.14	10.21
Random Hypervector Mapping	65.75	29.32	17.53
Reduced Hypervector Dim	66.42	40.34	25.04
VSAIT (ours)	**76.48**	**45.33**	**30.89**

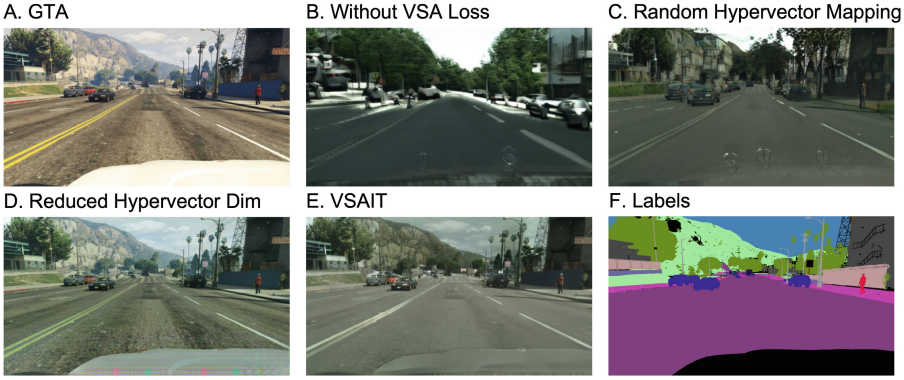

Fig. 5. Ablation study for GTA to Cityscapes. (A) and (F) show the ground truth image and semantic labels for the GTA source image, respectively. (B) demonstrates the challenge of semantic flipping when using GAN-based methods without constraining the generator. (C) shows the importance of learning the hypervector mapping. (D) demonstrates the importance of high-dimensional space for this method. (E) shows the effect on image translation when incorporating all components of our approach.

5 Conclusion

In this paper, we address the semantic flipping problem in unpaired image translation using a novel approach based on VSA framework [9,15]. We show important qualitative and quantitative improvements over previous methods that have attempted to address this problem, demonstrating that VSA can be used to invert image translations and ensure consistency with the source domain. Given its inherent versatility, we hope this work inspires future research using VSA in more computer vision applications.

Acknowledgments. We thank Mihir Jain, Shingo Takagi, Patrick Rodriguez, Sarah Watson, Zijian He, Peizhao Zhang, and Tao Xu for their helpful feedback.

References

1. Ahmad, S., Hawkins, J.: Properties of sparse distributed representations and their application to hierarchical temporal memory. arXiv preprint arXiv:1503.07469 (2015)
2. Almahairi, A., Rajeshwar, S., Sordoni, A., Bachman, P., Courville, A.: Augmented cyclegan: Learning many-to-many mappings from unpaired data. In: International Conference on Machine Learning, pp. 195–204. PMLR (2018)
3. Benaim, S., Wolf, L.: One-sided unsupervised domain mapping. In: Advances in neural Information Processing Systems 30 (2017)
4. Bińkowski, M., Sutherland, D.J., Arbel, M., Gretton, A.: Demystifying mmd gans. arXiv preprint arXiv:1801.01401 (2018)
5. Chen, L.C., Papandreou, G., Schroff, F., Adam, H.: Rethinking atrous convolution for semantic image segmentation. arXiv preprint arXiv:1706.05587 (2017)
6. Cordts, M., et al.: The cityscapes dataset for semantic urban scene understanding. In: CVPR (2016)
7. Devaranjan, J., Kar, A., Fidler, S.: Meta-sim2: unsupervised learning of scene structure for synthetic data generation. In: European Conference on Computer Vision, pp. 715–733. Springer (2020). https://doi.org/10.1007/978-3-030-58520-4_42
8. Fu, H., Gong, M., Wang, C., Batmanghelich, K., Zhang, K., Tao, D.: Geometry-consistent generative adversarial networks for one-sided unsupervised domain mapping. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 2427–2436 (2019)
9. Gayler, R.W.: Vector symbolic architectures answer jackendoff's challenges for cognitive neuroscience. arXiv preprint cs/0412059 (2004)
10. Goodfellow, I., et al.: Generative adversarial nets. In: Advances in Neural Information Processing Systems 27 (2014)
11. Hoffman, J., et al.: Cycada: cycle-consistent adversarial domain adaptation. In: International Conference on Machine Learning, pp. 1989–1998. PMLR (2018)
12. Huang, X., Liu, M.Y., Belongie, S., Kautz, J.: Multimodal unsupervised image-to-image translation. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 172–189 (2018)
13. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 1125–1134 (2017)
14. Jia, Z., et al.: Semantically robust unpaired image translation for data with unmatched semantics statistics. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 14273–14283 (2021)
15. Kanerva, P.: Hyperdimensional computing: an introduction to computing in distributed representation with high-dimensional random vectors. Cogn. Comput. **1**(2), 139–159 (2009). https://doi.org/10.1007/s12559-009-9009-8
16. Kar, A., et al: Meta-sim: learning to generate synthetic datasets. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 4551–4560 (2019)
17. Kent, S., Olshausen, B.: A vector symbolic approach to scene transformation. Cognitive computational neuroscience (ccn 2017) (extended abstract) [link] (2017)
18. Kim, T., Cha, M., Kim, H., Lee, J.K., Kim, J.: Learning to discover cross-domain relations with generative adversarial networks. In: International Conference on Machine Learning, pp. 1857–1865. PMLR (2017)

19. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: ICLR (2015)
20. Kleyko, D., Rachkovskij, D.A., Osipov, E., Rahim, A.: A survey on hyperdimensional computing aka vector symbolic architectures, part ii: applications, cognitive models, and challenges. arXiv preprint arXiv:2112.15424 (2021)
21. Kleyko, D., Rachkovskij, D.A., Osipov, E., Rahimi, A.: A survey on hyperdimensional computing aka vector symbolic architectures, part i: models and data transformations. arXiv preprint arXiv:2111.06077 (2021)
22. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in Neural Information Processing Systems 25 (2012)
23. Lee, H.Y., Tseng, H.Y., Huang, J.B., Singh, M., Yang, M.H.: Diverse image-to-image translation via disentangled representations. In: Proceedings of the European Conference on Computer Vision (ECCV), pp. 35–51 (2018)
24. Lim, J.H., Ye, J.C.: Geometric gan. arXiv preprint arXiv:1705.02894 (2017)
25. Liu, M.Y., Breuel, T., Kautz, J.: Unsupervised image-to-image translation networks. In: Advances in Neural Information Processing Systems 30 (2017)
26. Montone, G., O'Regan, J.K., Terekhov, A.V.: Hyper-dimensional computing for a visual question-answering system that is trainable end-to-end. arXiv preprint arXiv:1711.10185 (2017)
27. Neubert, P., Schubert, S.: Hyperdimensional computing as a framework for systematic aggregation of image descriptors. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 16938–16947 (2021)
28. Neubert, P., Schubert, S., Protzel, P.: An introduction to hyperdimensional computing for robotics. KI-Künstliche Intelligenz **33**(4), 319–330 (2019). https://doi.org/10.1007/s13218-019-00623-z
29. Osipov, E., et al.: Hyperseed: unsupervised learning with vector symbolic architectures. arXiv preprint arXiv:2110.08343 (2021)
30. Park, T., Efros, A.A., Zhang, R., Zhu, J.Y.: Contrastive learning for unpaired image-to-image translation. In: European Conference on Computer Vision, pp. 319–345. Springer (2020). https://doi.org/10.1007/978-3-030-58545-7_19
31. Prakash, A., Debnath, S., Lafleche, J.F., Cameracci, E., Birchfield, S., Law, M.T., et al.: Self-supervised real-to-sim scene generation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp. 16044–16054 (2021)
32. Purdy, S.: Encoding data for htm systems. arXiv preprint arXiv:1602.05925 (2016)
33. Richter, S.R., AlHaija, H.A., Koltun, V.: Enhancing photorealism enhancement. arXiv preprint arXiv:2105.04619 (2021)
34. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: Ground truth from computer games. In: European Conference on Computer Vision, pp. 102–118. Springer (2016). https://doi.org/10.1007/978-3-319-46475-6_7
35. Schlegel, K., Neubert, P., Protzel, P.: A comparison of vector symbolic architectures. Artif. Intell. Rev. 1–33 (2021). https://doi.org/10.1007/s10462-021-10110-3
36. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
37. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 2818–2826 (2016)
38. Yi, Z., Zhang, H., Tan, P., Gong, M.: Dualgan: unsupervised dual learning for image-to-image translation. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2849–2857 (2017)
39. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision, pp. 2223–2232 (2017)