Journal of Machine Learning Research 22 (2021) 1-42 Submitted 7/18; Revised 2/21; Published 5/21

Risk Bounds for Unsupervised Cross-Domain Mapping with IPMs

Tomer Galanti TOMERGA2@POST.TAU.AC.IL
School of Computer Science
Tel Aviv University
Ramat Aviv, Tel Aviv 69978, Israel

Sagie Benaim SAGIEB@MAIL.TAU.AC.IL
School of Computer Science
Tel Aviv University
Ramat Aviv, Tel Aviv 69978, Israel

Lior Wolf WOLF@CS.TAU.AC.IL
Facebook AI Research and
School of Computer Science
Tel Aviv University
Ramat Aviv, Tel Aviv 69978, Israel

Editor: Christoph Lampert

Abstract

The recent empirical success of unsupervised cross-domain mapping algorithms, in mapping between two domains that share common characteristics, is not well-supported by theoretical justifications. This lacuna is especially troubling, given the clear ambiguity in such mappings.

We work with adversarial training methods based on integral probability metrics (IPMs) and derive a novel risk bound, which upper bounds the risk between the learned mapping h and the target mapping y, by a sum of three terms: (i) the risk between h and the most distant alternative mapping that was learned by the same cross-domain mapping algorithm, (ii) the minimal discrepancy between the target domain and the domain obtained by applying a hypothesis h^* on the samples of the source domain, where h^* is a hypothesis selectable by the same algorithm, and (iii) an approximation error term that decreases as the capacity of the class of discriminators increases and is empirically shown to be small. The bound is directly related to Occam's razor and encourages the selection of the minimal architecture that supports a small mapping discrepancy.

The bound leads to multiple algorithmic consequences, including a method for hyperparameter selection and early stopping in cross-domain mapping.

Keywords: unsupervised learning, cross-domain alignment, integral probability metrics, adversarial training, image to image translation.

1. Introduction

The recent literature contains many examples of unsupervised learning that are beyond the classical work on clustering and density estimation, most of which revolve around generative models that are trained to capture a certain distribution, D. In many cases, the generation is unconditioned, and the learned hypothesis takes the form of $g(z)$ for a random vector z. It is obtained based on a training set containing i.i.d. samples from D.

A large portion of the recent literature on this problem employs adversarial training, and specifically a variant of generative adversarial networks (GANs), which were introduced by Goodfellow et al. (2014). GAN-based schemes typically employ two functions that are learned jointly: a generator g and a discriminator d. The discriminator is optimized to distinguish between "real" training samples from the distribution D and "fake" samples that are generated as $g(z)$, where z is distributed according to a predefined latent distribution D_z (typically, a low-dimensional normal or uniform distribution). The generator is optimized to generate adversarial samples, that is, samples $g(z)$, such that d would classify as real. This *unconditional generation using GANs* task is explored theoretically (Arora et al., 2017), and since intuitive non-adversarial (interpolation-based) techniques exist (Bojanowski et al., 2018), their success is also not surprising.

Much less understood is the ability to learn, in a completely unsupervised manner, in the conditioned case, where the learned function h maps a sample from a source domain A to the analogous sample in a target domain B. In this case, we have two distributions D_A, D_B and one aims at mapping a sample $a \sim D_A$ to an analogous sample $h(a) \sim D_B$. This computational problem is known as "Unsupervised Cross-Domain Mapping" or "Image to Image Translation" when considering visual domains. There are a few issues with this computational problem that cause concern. First, it is unclear what analogous means, let alone to capture it in a formula. Second, as detailed in Section 4.2, the mapping problem is inherently ambiguous.

Despite these theoretical challenges, the field of unsupervised cross-domain mapping, in which a sample from domain A is translated to a sample in a second domain B, is enjoying a great deal of empirical success, e.g, (He et al., 2016; Kim et al., 2017; Zhu et al., 2017; Yi et al., 2017; Benaim and Wolf, 2017; Liu et al., 2017). Many of these contributions are based on what is called *circularity losses*, in which one learns a mapping from one domain to another that is also approximately invertible. However, as we discuss in Section 4.3 and as we show empirically, these constraints do not eliminate the inherent ambiguity in these problems. This raises interesting questions, such as, under what conditions (i.e., type of data, architecture, etc') do these methods succeed in unsupervised cross-domain mapping?

In this paper, we attribute this success to what we term, the "simplicity hypothesis", which means that these solutions learn the minimal complexity mapping, such that the discrepancy between the fitted distribution and the target distribution is small. As we show empirically, training a neural network of a small depth eliminates the ambiguity of the problem.

In addition to the empirical validation, we present an upper bound on the generalization risk that supports the simplicity hypothesis. Bounding the error obtained with unsupervised methods is subject to an inherent challenge: without the ability to directly evaluate the risk on the training set, it is not clear on which grounds to build the bound. Specifically, typical generalization bounds of the form of training risk plus a regularization term cannot be used.

The bound we construct has a different form. As one component, it has the success of the fitting process. This is captured by the mapping discrepancy, measured by the integral probability metric (IPM) (Müller, 1997) between the target distribution D_B and the distribution of generated samples $h \circ D_A$ (that is, the distribution of $h(a)$ for $a \sim D_A$), and is typically directly minimized by the learner. Another component is the maximal risk within the hypothesis class to any other hypothesis that also provides a good fit. This term is linked to the complexity of the hypothesis class, since it is expected to be small for hypothesis classes of small capacity, and it can be estimated empirically for any hypothesis class.

In addition to explaining the plausibility of unsupervised cross-domain mapping despite the inherent ambiguities, our analysis also directly leads to a set of new unsupervised cross-domain mapping algorithms. By training pairs of networks that are distant from each other with both minimizing the mapping discrepancy, we can obtain a measure of confidence on the mapping's outcome. This is surprising for two main reasons: first, in unsupervised settings, confidence estimation is almost unheard of, since it typically requires a second set of supervised samples. Second, confidence is hard to calibrate for multidimensional outputs. The confidence estimation is used as a criterion for early stopping (Algorithm 1) and can be used for hyperparameter selection (Algorithm 2). These algorithms serve as high-level improvements over pre-existing cross-domain mapping algorithms and can be applied to a wide variety of methods.

2. Contributions

The work described here is part of the line of work on the role of minimal complexity in unsupervised learning that we have been following in conference publications (Galanti et al., 2018; Benaim et al., 2018). Our contributions in this line of work are as follows.

1. Theorem 1 provides a rigorous statement of a risk bound for unsupervised cross-domain mapping with IPMs, which is the basis of this work. This bound sums three main terms: (a) The maximal risk within the hypothesis class to any other hypothesis that also provides a good fit. (b) The error of fitting between the two domains. This is captured by the IPM (Müller, 1997) which is typically directly minimized by the learner. (c) An approximation error term that decreases as the capacity of the class of discriminators increases and is empirically shown to be small.

2. Theorem 1 yields a concrete prediction that is verified experimentally in Section 8. Based on this theorem, we also introduce Algorithms 1 and 2. The first serves as a method for early stopping in unsupervised cross-domain mapping. The second algorithm provides a method for hyperparameter selection for unsupervised cross-domain mapping.

3. Our line of work shows that unsupervised cross-domain mapping succeeds, when the architecture of the learned generator is of minimal depth.

4. In Section 7, we extend our analysis to the non-unique case. In this case, there are multiple possible target functions and we wish our algorithm to return a hypothesis that is close to one of them. This extension leads to Algorithm 3 that extends Algorithm 1, which is then verified experimentally.

The algorithms presented here, and the empirical results, are extensions of those in the conference publications, except for Algorithm 3 that extends Algorithm 1 to the non-unique case, which is new. The contributions in this manuscript over the previous conference publications include: (i) In this paper, we employ integral probability metrics (IPMs), while previous work employed a different measure of discrepancy (a specific type of IPM). (ii) We derive a precise bound for cross-domain mapping (Theorem 1), which was missing in our previous work. While in (Benaim et al., 2018), we provide bounds for unsupervised cross-domain mapping, it is mainly used for motivating the methods and it strongly relies on their "Occam's razor property" that does not necessarily hold in practice. (iii) As mentioned, in Section 7, we extend our analysis for the non-unique case.

3. Background

We briefly review IPMs and WGANs. All notations are listed in Table 1.

3.1 Terminology and Notations

We introduce some necessary terminology and notations. We denote by $\mathbb{P}$ and $\mathbb{E}$ the probability and expectation operators. We denote by $\text{Id}_{\mathcal{X}} : \mathcal{X} \to \mathcal{X}$ the identity function. Throughout the paper, we follow the convention of treating vectors $x \in \mathbb{R}^n$ as column vectors $x \in \mathbb{R}^{n\times 1}$. For a vector $x \in \mathbb{R}^n$, $\|x\|_2$ denotes the Euclidean norm of x and for a matrix $W \in \mathbb{R}^{m\times n}$, $\|W\|_2 := \max_{x\neq 0}(\|Wx\|_2/\|x\|_2)$ stands for the induced operator norm of W.

A hypothesis class $\mathcal{H}$ is a set of functions $h : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ is considered as the instances space and $\mathcal{Y}$ is referred as the labels space. We let $\ell : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ be a loss function. For a given hypothesis class $\mathcal{H}$ and loss function ℓ, we denote, $\ell_{\mathcal{H}} = \{x \mapsto \ell(h(x), h'(x))\}_{h,h'\in\mathcal{H}}$. For simplicity, when $\mathcal{X}$ is clear from the context, instead of writing $\inf_{x\in\mathcal{X}}$ and $\sup_{x\in\mathcal{X}}$, we will write $\inf_x$ and $\sup_x$.

Let $f : \mathcal{X} \to \mathcal{Y}$ be a function, such that, $\mathcal{X} \subset \mathbb{R}^m$ and $\mathcal{Y} \subset \mathbb{R}^n$. If f is differentiable, we denote by $\mathcal{J}_f(x) \in \mathbb{R}^{n\times m}$ the Jacobian matrix of f at x. Additionally, if $n = 1$, we denote by $\nabla f(x) \in \mathbb{R}^{m\times 1}$ (column vector) the gradient of f, and if f is twice differentiable, $\text{H}_f(x) \in \mathbb{R}^{m\times m}$ stands for the Hessian matrix of f at x. We denote $f \in C^r$ if f is r-times continuously differentiable. We define, $\|f\|_{\infty,\mathcal{X}} := \sup_{x\in\mathcal{X}} \|f(x)\|_2$ and $\|f\|_{\text{Lip}} = \sup_{x,y\in\mathcal{X}} (\|f(x) - f(y)\|_2/\|x - y\|_2)$. For a twice differentiable function f, we denote $\beta(f) := \|\text{H}_f\|_{\infty,\mathcal{X}}$. Given a set E and two functions $F : E \to \mathbb{R}$ and $G : E \to \mathbb{R}$, we denote, $F \lesssim G$ if and only if $\exists\, C > 0\ \forall\, e \in E : F(e) \leq C \cdot G(e)$.

3.2 IPMs and WGANs

In this section, we provide some general background on integral probability metrics (IPMs) and Wasserstein GANs (WGANs).

3.2.1 INTEGRAL PROBABILITY METRICS

IPMs, first introduced by Müller (1997), is a family of pseudometric[1] functions between distributions. Formally, for a given Polish space $\mathcal{S} = (\mathcal{X}, \|\cdot\|)$ (that is, separable and completely metrizable topological space), two distributions D_1 and D_2 over $\mathcal{X}$ and a class $\mathcal{C}$ of discriminator functions $d : \mathcal{X} \to \mathbb{R}$, the $\mathcal{C}$-IPM between D_1 and D_2 is defined as follows:

$$\rho_{\mathcal{C}}(D_1, D_2) := \sup_{d\in\mathcal{C}} \Big\{\mathbb{E}_{x\sim D_1}[d(x)] - \mathbb{E}_{x\sim D_2}[d(x)]\Big\}. \tag{1}$$

This family of functions includes a wide variety of pseudometric functions (Arjovsky et al., 2017; Zhao et al., 2017; Berthelot et al., 2017; Li et al., 2015, 2017; Mroueh and Sercu, 2017; Mroueh et al., 2018). In order to guarantee that $\rho_{\mathcal{C}}$ is non-negative throughout the paper, we assume that $\mathcal{C}$ is symmetric, that is, if $d \in \mathcal{C}$, then, $-d \in \mathcal{C}$.

[1] A pseudometric $d : X^2 \to [0, \infty)$ is a non-negative, symmetric function that satisfies the triangle inequality and $d(x, x) = 0$ for all $x \in X$.

3.2.2 WGANS OPTIMIZATION

In this work, we give special attention to the WGAN algorithm (Arjovsky et al., 2017) and its extensions (Zhao et al., 2017; Berthelot et al., 2017; Li et al., 2015, 2017; Mroueh and Sercu, 2017; Mroueh et al., 2018). These are variants of GAN that use the IPM, instead of the original GAN loss. In general, their aim is to find a mapping $g : \mathcal{X}_1 \to \mathcal{X}_2$ (generator) that takes one distribution, D_1 over $\mathcal{X}_1$, and maps it into a second distribution D_2, by minimizing the distance between $g \circ D_1$ (the distribution of $g(z)$ for $z \sim D_1$) and D_2. Hence, the goal is to select a mapping (generator) g from a class, $\mathcal{H}$, of neural networks of a fixed architecture, that minimizes the *mapping discrepancy*:

$$\rho_C(g \circ D_1, D_2) = \sup_{d \in C} \Big\{ \mathbb{E}_{z \sim D_1}[d(g(z))] - \mathbb{E}_{x \sim D_2}[d(x)] \Big\}, \tag{2}$$

where $g \circ D$ is the distribution of $g(x)$ for $x \sim D$. For this purpose, these methods make use of finite sets of i.i.d. samples $\mathcal{S}_1 = \{z_i\}_{i=1}^{m_1}$ and $\mathcal{S}_2 = \{x_j\}_{j=1}^{m_2}$ from D_1 and D_2 (resp.). The optimization process iteratively minimizes $\Big\{ \frac{1}{m_1} \sum_{i=1}^{m_1} [d(g(z_i))] - \frac{1}{m_2} \sum_{j=1}^{m_2} [d(x_j)] \Big\}$ with respect to g and maximizes it with respect to d. In each iteration, the algorithm runs a few gradient based optimization steps for g or d.

In the task of unconditional generation (Goodfellow et al., 2014; Arjovsky et al., 2017), the set $\mathcal{X}_1 = \mathcal{X}_z$ is considered as a latent space and is typically a convex subset of a Euclidean space, such as $\mathbb{R}^d$, $[-1,1]^d$ or the d-dimensional closed unit ball, $\mathbb{B}_d := \{x \in \mathbb{R}^d \mid \|x\|_2 \leq 1\}$. Additionally, the input distribution $D_1 = D_z$ is typically a normal distribution (for $\mathcal{X}_z = \mathbb{R}^d$) or a uniform distribution (for $\mathcal{X}_z = [-1,1]^d$ or $\mathcal{X}_z = \mathbb{B}_d$). As we see next, we focus on conditional generation (He et al., 2016; Kim et al., 2017; Zhu et al., 2017; Yi et al., 2017; Benaim and Wolf, 2017; Liu et al., 2017), where $D_1 = D_A$ and $D_2 = D_B$ are two distributions over analogous visual domains.

As a side note, to derive Equation (2), we used (cf. Varadhan, 2002, Theorem 1.9).

4. Problem Setup

In this paper, we consider the *unsupervised cross-domain mapping problem*. In this setting, there are two *domains* $A = (\mathcal{X}_A, D_A)$ and $B = (\mathcal{X}_B, D_B)$, where D_A and D_B are distributions over the sample spaces $\mathcal{X}_A \subset \mathbb{R}^N$ and $\mathcal{X}_B \subset \mathbb{R}^M$ respectively (formally, we assume that both spaces are equipped with σ-algebras). In addition, there is a hypothesis class $\mathcal{H}$ of functions $h : \mathcal{X}_A \to \mathcal{X}_B$ from which candidate hypotheses are being selected and a loss function $\ell : \mathbb{R}^M \times \mathbb{R}^M \to [0, \infty)$. Our results are shown for the L_2-loss $\ell(x_1, x_2) = \|x_1 - x_2\|_2^2$.

In this setting, there is an unknown target function y that maps the first domain to the second domain, that is, $y : \mathcal{X}_A \to \mathcal{X}_B$ and $D_B = y \circ D_A$. The function y will also be referred as the *"semantic alignment"* between D_A and D_B, as opposed to non-semantic alignments $f \neq y$ that map $f \circ D_A = D_B$. In Section 7, we extend the framework and the results to include multiple target functions.

As an example that is often used in the literature, $\mathcal{X}_A$ is a set of images of shoes, and $\mathcal{X}_B$ is a set of images of shoe edges, see Figure 1(a). Here, D_A is a distribution of images of shoes and D_B a distribution of images of shoe edges. The function y takes an image of a shoe and maps it to an image of the edges of the shoe. The assumption that $y \circ D_A = D_B$ simply means that the target function, y, takes a sampled image of a shoe $x \sim D_A$ and maps it to a sample $y(x)$ from the distribution of images of edges.

$\mathbb{P}, \mathbb{E}$	The probability and expectation operators
ℓ	The L_2 loss function, that is, $\ell(a, b) = \|a - b\|_2^2$
$\text{Id}_{\mathcal{X}}$	The identity function
A, B	Two domains $A = (\mathcal{X}_A, D_A)$ and $B = (\mathcal{X}_B, D_B)$
R_D, $R_{\mathcal{S}}$	The generalization and empirical risk functions
$\mathcal{H}, h$	A hypothesis class and a specific hypothesis
$\mathcal{C}, d$	A class of discriminators and a specific discriminator
$\mathcal{T}, y$	A set of target functions and a specific target function
$\rho_{\mathcal{C}}$	The $\mathcal{C}$-IPM (integral probability metric, see Equation 1)
Ω, ω	A set of vectors of hyperparamers and a specific vector of hyperparameters
$\mathcal{A}_\omega$	A cross-domain mapping algorithm with hyperparameters ω
$\mathcal{P}_\omega / \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$	The set of possible outputs of $\mathcal{A}_\omega$ provided with inputs $(\mathcal{S}_A, \mathcal{S}_B)$
$\mathcal{H}_k$	A hypothesis class of functions of depth $\leq k$
$\mathcal{A}_k$	A cross-domain mapping algorithm of generators from $\mathcal{H}_k$
$\mathcal{P}_k / \mathcal{P}_k(\mathcal{S}_A, \mathcal{S}_B)$	The set of possible outputs of $\mathcal{A}_k$ provided with input $(\mathcal{S}_A, \mathcal{S}_B)$
$\|x\|_2$, $\|W\|_2$	The Euclidean and induced operator norms
$\mathcal{J}_f$, ∇f, H_f	The Jacobian, gradient and Hessian operators
$\|f\|_{\infty,\mathcal{X}}$	The infinity norm of $f : \mathcal{X} \to \mathbb{R}^n$
$\|f\|_{\text{Lip}}$	The Lipschitz norm $f : \mathcal{X} \to \mathbb{R}^n$
$\beta(f)$	The maximal operator norm of the Hessian of $f : \mathcal{X} \to \mathbb{R}$
C^r	The set of r-times continuously differentiable functions
$F \lesssim G$	$\exists\, C > 0\, \forall\, e \in E : F(e) \leq C \cdot G(e)$

Table 1: Summary of Notation

In contrast to the supervised case, where the learning algorithm is provided with a data set of labeled samples $(x, y(x))$ for $x \sim D$ and y is the target function, in the unsupervised case that we study, the only inputs of the learning algorithm $\mathcal{A}$ are i.i.d. samples from the two distributions D_A and D_B independently,

$$\mathcal{S}_A \overset{\text{i.i.d.}}{\sim} D_A^{m_1} \text{ and } \mathcal{S}_B \overset{\text{i.i.d.}}{\sim} D_B^{m_2}.$$

The set $\mathcal{S}_A$ consists of instances from $\mathcal{X}_A$ and the set $\mathcal{S}_B$ consists of instances from $\mathcal{X}_B$. We also do not assume that for any $a \in \mathcal{S}_A$ there is a corresponding $b \in \mathcal{S}_B$, such that, $b = y(a)$.
The goal of the learning algorithm $\mathcal{A}$ is to fit a function $h \in \mathcal{H}$ that minimizes $R_{D_A}[h, y]$. Here, $R_D[f_1, f_2]$ is the generalization risk function between f_1 and f_2 with respect to a distribution D, that is defined in the following manner:

$$R_D[f_1, f_2] := \mathbb{E}_{x \sim D}\left[\ell(f_1(x), f_2(x))\right].$$

In supervised learning, the algorithm is provided with the labels of the target function y on the training set $\mathcal{S}_A$ and estimates the generalization risk $R_{D_A}[h, y]$ using the empirical risk $R_{\mathcal{S}_A}[h, y] := \frac{1}{|\mathcal{S}_A|}\sum_{x \in \mathcal{S}_A} \ell(h(x), y(x))$. In the proposed unsupervised setting, one cannot estimate this risk on the training samples, since the algorithm is not provided with the labeled samples $(x, y(x))$. Instead, the learner must rely on the two independent sets $\mathcal{S}_A$ and $\mathcal{S}_B$.

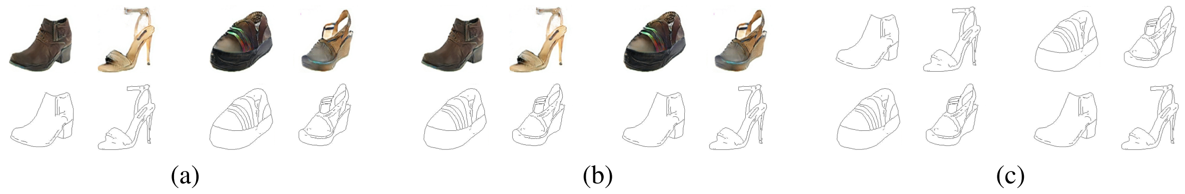


Figure 1: **The alignment problem.** Domain A consists of shoes and domain B consists of edges of shoes. (a) The correct alignment y between the two domains. (b) A wrong alignment $\hat{h}$ between the two domains. The algorithm is provided with independent samples from domain A and from domain B. It is not obvious what makes the algorithm return the mapping (a) instead of any other mapping between the two domains. (c) A permutation function Π that gives (b) when applied on (a), that is, $\hat{h} = \Pi \circ y$.

With regards to the example above, the learning algorithm is provided with a set of m_1 images of shoes and m_2 images of shoe edges. The two sets are independent and unmatched. The goal of the learning algorithm is to provide a hypothesis h that approximates y. Informally, we want to have $h(a) \approx y(a)$ in expectation over $a \sim D_A$, that is, h and y map the same image of a shoe to the same image of shoe edges.

Two modes of failure Even if the algorithm is provided with an infinite amount of samples, it can fail in two different ways. (i) h can fail to produce the output domain, that is, $h \circ D_A$ will diverge from D_B. This is typically a result of limited expressivity. (ii) Even if $h \circ D_A = D_B$, h could be distant from y, that is, there would be a high probability for samples $a \sim D_A$, such that, $\ell(h(a), y(a))$ is large, which is discussed in Section 4.2.

4.1 Assumptions

Several assumptions were made to obtain the theoretical results.

Assumption 1 (Setting and data) *The sets $\mathcal{X}_A \subset \mathbb{R}^N$ and $\mathcal{X}_B \subset \mathbb{R}^M$ are convex, bounded and equipped with σ-algebras. D_A and D_B are two distributions over (but not necessarily supported by) $\mathcal{X}_A$ and $\mathcal{X}_B$ (resp.). The data sets $\mathcal{S}_A$ and $\mathcal{S}_B$ consist of m_1 and m_2 i.i.d. samples from the two distributions D_A and D_B (resp.). We use the L_2-loss function $\ell(x_1, x_2) = \|x_1 - x_2\|_2^2$. In Section 5, the target function y is assumed to be unique and to satisfy $y \circ D_A = D_B$. In Section 7, we consider the case where there are multiple target functions $\mathcal{T}$.*

Assumption 2 (Architectures) *The hypothesis class $\mathcal{H}$ consists of functions $h : \mathcal{X}_A \to \mathcal{X}_B$. The class of discriminators $\mathcal{C}$ (see Section 4.4) is a subset of C^2 and satisfies that $\sup_{d \in \mathcal{C}} \|d\|_{\infty, \mathcal{X}_B} < \infty$.*

The second assumption is a technical assumption that holds, for example, when $\mathcal{X}_B$ is the closed ball of some radius $r \geq 1$ around 0, $\mathcal{H}$ is a class of neural networks that output vectors of norm ≤ 1 and $\mathcal{C}$ is a class of bounded neural networks with twice-continuously differentiable activation functions.

4.2 The Unsupervised Alignment Problem

We next address that the proposed unsupervised learning setting suffers from what we term as the *"alignment problem"*. The problem arises from the fact that when observing samples $\mathcal{S}_A$ and $\mathcal{S}_B$ only from the marginal distributions D_A and D_B, one cannot uniquely link the samples in the source domain to those of the target domain, see Figure 1(b).

As a simple example, let D_A and D_B be two discrete distributions, such that, there are two points $a, a' \in \mathcal{X}_A$ that satisfy $\mathbb{P}_{x \sim D_A}[x = a] = \mathbb{P}_{x \sim D_A}[x = a']$. Assuming that the mapping y is one-to-one, then $y(a)$ and $y(a')$ have the same likelihood in the density function $\mathbb{P}_{x \sim D_B}[x = \cdot]$. Therefore, a-priori it is unclear if the target mapping takes a and maps it to $y(a)$ or to $y(a')$.

More generally, given the target function y between the two domains, in many cases it is possible to define many alternative mappings of the form $\hat{h} = \Pi \circ y$, where Π is a mapping that satisfies $\Pi \circ D_B = D_B$. For such functions, we have, $\hat{h} \circ D_A = \Pi \circ y \circ D_A = \Pi \circ D_B = D_B$, and therefore, they satisfy the same assumptions we had regarding the target function y.

Thus, a-priori it is unclear why a cross-domain mapping algorithm that only observes samples from D_A and D_B would recover the target mapping y instead of any arbitrary mapping $\hat{h} \neq y$, such that, $\hat{h} \circ D_A = D_B$. In Figure 1(b), the mapping can be represented as $\hat{h} = \Pi \circ y$, where Π is the mapping illustrated in Figure 1(c).

4.3 Circularity Constraints do not Eliminate All of the Inherent Ambiguity

In the field of unsupervised cross-domain mapping, most contributions learn the mapping h between the two domains A and B by employing two main constraints. Firstly, h is restricted to minimize a GAN loss. In this work, in order to support a more straightforward analysis, we employ IPMs and h minimizes $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$ (Equation 1). In (Lucic et al., 2018), it has been shown that many of the GAN methods in the literature perform similarly.

A large portion of the cross-domain mapping algorithms also employ what is called the circularity constraint (He et al., 2016; Kim et al., 2017; Zhu et al., 2017; Yi et al., 2017). Circularity requires learning a second mapping h' that maps between B and A (the opposite direction of h) and serves as an inverse function to h. Similar to h, h' is trained to minimize a GAN loss but in the other direction, that is, $\rho_C(h' \circ \mathcal{S}_B, \mathcal{S}_A)$. The circularity terms, which are minimized by h and h' take the form $R_{\mathcal{S}_A}[h' \circ h, \text{Id}_{\mathcal{X}_A}]$ and $R_{\mathcal{S}_B}[h \circ h', \text{Id}_{\mathcal{X}_B}]$, where $\text{Id}_{\mathcal{X}} : \mathcal{X} \to \mathcal{X}$ is the identity function, that is, $\forall x \in \mathcal{X} : \text{Id}_{\mathcal{X}}(x) = x$. In other words, for a sample $a \in \mathcal{S}_A$, we expect to have, $h'(h(a)) \approx a$ and for a random sample $b \in \mathcal{S}_B$, we expect to have, $h(h'(b)) \approx b$.

Therefore, the complete minimization objective of both h and h' is as follows:

$$\inf_{h,h' \in \mathcal{H}} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) + \rho_C(h' \circ \mathcal{S}_B, \mathcal{S}_A) \tag{3a}$$

$$+ R_{\mathcal{S}_A}[h' \circ h, \text{Id}_{\mathcal{X}_A}] + R_{\mathcal{S}_B}[h \circ h', \text{Id}_{\mathcal{X}_B}]. \tag{3b}$$

The terms in Equation (3a) ensure that the samples generated by mapping domain A to domain B follow the distribution of samples in domain B and vice versa. The terms in Equation (3b) ensure that mapping a sample from one domain to the second and back, results in the original sample. Note that the first two terms match distributions (via the IPM scores) and the last two match individual samples (via the loss ℓ in the risk).

The circularity terms are shown empirically to improve the obtained results. However, these terms do not eliminate all of the inherent ambiguity, as shown in the following observation. For instance, consider the favorable case where the algorithm has full access to D_A and D_B, that is, $\mathcal{S}_A = D_A$ and $\mathcal{S}_B = D_B$. Let Π be an invertible permutation of D_B, that is, $\Pi : \mathcal{X}_B \to \mathcal{X}_B$ is an invertible mapping and $\Pi \circ D_B = D_B$. Then, the pair $h = \Pi \circ y$ and $h' = y^{-1} \circ \Pi^{-1}$ achieves:

$$\rho_C(h \circ D_A, D_B) + \rho_C(h' \circ D_B, D_A) + R_{D_A}[h' \circ h, \text{Id}_{\mathcal{X}_A}] + R_{D_B}[h \circ h', \text{Id}_{\mathcal{X}_B}] = 0.$$

Informally, if Π is an invertible permutation of the samples in domain B (*not* a permutation of the vector elements of the representation of samples in B), then, if y is the target function and y^{-1} is its inverse function, the pair of functions $h = \Pi \circ y$ and $h' = y^{-1} \circ \Pi^{-1}$ achieves zero losses. Therefore, even though the function $h = \Pi \circ y$ might correspond to an incorrect alignment between the two domains A and B (that is, the function h is very different from y), the pair h and h' can still achieve a zero value on each of the losses proposed by He et al. (2016); Kim et al. (2017); Zhu et al. (2017); Yi et al. (2017).

Since both low discrepancy and circularity cannot, separately or jointly, eliminate the ambiguity of the mapping problem, a complete explanation of the success of unsupervised cross-domain mapping must consider the hypothesis classes $\mathcal{H}$ and $\mathcal{C}$. This is what we intend to do in Section 5.

4.4 Cross-Domain Mapping Algorithms

A central goal in this work is the derivation of risk bounds that can be used to compare different cross-domain mapping algorithms. The set of cross-domain mapping algorithms $\{\mathcal{A}_\omega\}_{\omega \in \Omega}$ that are compared, are indexed by a vector of hyperparameters $\omega \in \Omega$. The vector of hyperparameters ω can include the architecture of the hypothesis class from which $\mathcal{A}_\omega$ selects candidates, the learning rate, batch size, etc'. To compare the performance of the algorithms, an upper bound on the term $R_{D_A}[h, y]$ is provided. Fortunately, this bound can be estimated without the need for supervised data, that is, without paired matches $(x, y(x))$. Here, h is the selected hypothesis by a cross-domain mapping algorithm $\mathcal{A}_\omega$ provided with access to the distributions D_A and D_B.

The outcome of every deep learning algorithm often depends on the random initialization of its parameters and the order in which the samples are presented. Such non-deterministic algorithm $\mathcal{A}_\omega$ can be seen as a mapping from the training data $(\mathcal{S}_A, \mathcal{S}_B)$ to a subset of the hypothesis space $\mathcal{H}$. This subset, which is denoted as $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$, contains all the hypotheses that the algorithm may return for the given training data. Typically, the set $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$ is much sparser than the original hypothesis class. Since the algorithm is not assumed to be deterministic, to measure the performance of a cross-domain mapping algorithm $\mathcal{A}_\omega$, an upper bound on $R_{D_A}[h, y]$ is derived for any $h \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$. For simplicity, we will sometimes simply write $\mathcal{P}_\omega$ as a reference to $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$.

In this paper, special attention is given to IPM minimization algorithms applied to unsupervised cross-domain mapping. The algorithm, $\mathcal{A}_\omega$, given access to two data sets $\mathcal{S}_A$ and $\mathcal{S}_B$, a hypothesis class $\mathcal{H}_\omega$ and a class of discriminators $\mathcal{C}$, returns a hypothesis $h \in \mathcal{H}_\omega$ (see Equation (1)), that minimizes $\rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)$.

The following are concrete examples of the proposed framework. We specify, ω, $\mathcal{H}$, $\mathcal{H}_\omega$ and $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$ for different settings:

1. **The hyperparameters are the learning rate and batch size:** in this case, each $\omega = (\mu, s, T)$ includes a different learning rate $\mu > 0$, batch size $s \in \mathbb{N}$ and number of iterations $T \in \mathbb{N}$. The hypothesis class $\mathcal{H}_\omega$ from which $\mathcal{A}_\omega$ selects candidates is $\mathcal{H}$ itself. The set $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$ consists of the minimizers $h \in \mathcal{H}$ of $\rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)$ trained using the procedure in Section 3.2 with a learning rate μ, a batch size s and the number of epochs T.

2. **The hyperparameter is the number of layers:** in this case, $\mathcal{H}$ is a class of neural networks of varying number of layers, each of size $\in [r_1, r_2]$, for some predefined $r_1, r_2 \in \mathbb{N}$. The hyperparameter is the maximal number of layers $\omega = k \in \mathbb{N}$ of the trained neural network $h \in \mathcal{H}$. The hypothesis class $\mathcal{H}_k$ is the set of neural networks in $\mathcal{H}$ that have a depth $\leq k$.

The algorithm $\mathcal{A}_k$ returns a hypothesis $h \in \mathcal{H}_k$ that minimizes $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$. More formally, $\mathcal{P}_k(\mathcal{S}_A, \mathcal{S}_B) = \{h \in \mathcal{H}_k \mid \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) \leq c \inf_{h^* \in \mathcal{H}_k} \rho_C(h^* \circ \mathcal{S}_A, \mathcal{S}_B)\}$ for some predefined multiplicative tolerance parameter $c \geq 1$ of our choice.

3. **The hyperparameters are the weights of the first layers:** in this case, $\mathcal{H} = \{h_{\theta,\omega} = g_\theta \circ f_\omega \mid \theta \in \Theta, \omega \in \Omega\}$ is a set of neural networks, each parameterized by two sets of parameters $\theta \in \Theta$ and $\omega \in \Omega$. We can think of f_ω as the first l_1 layers of $h_{\theta,\omega}$ and g_θ as the last l_2 layers of it. For instance, f_ω can be an encoder and g_θ a decoder. Here, $\mathcal{H}_\omega = \{h_{\theta,\omega} \mid \theta \in \Theta\}$ is the set of neural networks $h_{\theta,\omega} \in \mathcal{H}$ with fixed ω. In this setting, $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$ consists of the set of the possible minimizers $h \in \mathcal{H}_\omega$ of $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$. More formally, $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B) = \{h \in \mathcal{H}_\omega \mid \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) \leq c \inf_{h^* \in \mathcal{H}} \rho_C(h^* \circ \mathcal{S}_A, \mathcal{S}_B)\}$ for some predefined multiplicative tolerance parameter $c \geq 1$ of our choice.

In general, our bounds hold for any set of classes $\{\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)\}_{\omega \in \Omega}$ (not even minimizers of the mapping discrepancy). However, to obtain the simplified analysis in Section 5.1, we consider sets $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$, such that, the hypotheses $h \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$ achieve a fairly similar degree of success at minimizing $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$, that is, there is a multiplicative tolerance constant $c \geq 1$, such that, for all $h \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$, we have: $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) \leq c \inf_{h^* \in \mathcal{P}_\omega} \rho_C(h^* \circ \mathcal{S}_A, \mathcal{S}_B)$. We note that this assumption is not necessary to our analysis and one can obtain a more general version of Theorem 1 without it (see Lemma 8 in Appendix A). Specifically, this condition is met for examples (2) and (3) above. We note that $\mathcal{P}_k(\mathcal{S}_A, \mathcal{S}_B)$ and $\cup_\omega \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$ are always non-empty.

5. Risk Bounds for Unsupervised Cross-Domain Mapping

In this section, we discuss sufficient conditions for overcoming the alignment problem. For the sake of simplicity, we focus on the unique case, where there is only one target function y. The results are then extended, in Section 7, to the non-unique case, where there are multiple target functions.

5.1 Risk Bounds

We derive risk bounds for unsupervised cross-domain mapping, comparing alternative hyperparameters $\omega \in \Omega$. As discussed in Section 4, our goal is to select ω that provides the best performing algorithm $\mathcal{A}_\omega$, in terms of minimizing $R_{D_A}[h_1, y]$ for an output h_1 of $\mathcal{A}_\omega$ provided with access to D_A and D_B. For this purpose, Theorem 1 will provide us with an upper bound (for every $\omega \in \Omega$) on the generalization risk $R_{D_A}[h_1, y]$, for an arbitrary hypothesis h_1 selected by the algorithm $\mathcal{A}_\omega$, that is, $h_1 \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$. The terms of the bound can be estimated in an unsupervised manner, except one term that is empirically shown to be small. See Section 6 for the derived algorithms.
Our risk bounds take into account the transition between the population distribution and an empirical set of samples from it. This analysis is based on the following data-dependent measure of the complexity of a class of functions.

Definition 1 (Rademacher Complexity) *Let $\mathcal{H}$ be a set of real-valued functions $f : \mathcal{X} \to \mathbb{R}$ defined over a set $\mathcal{X}$. Given a fixed sample $S \in \mathcal{X}^m$, the empirical Rademacher complexity of $\mathcal{H}$ is defined as follows:*

$$\hat{\mathscr{R}}_S(\mathcal{H}) := \frac{2}{m} \mathbb{E}_\sigma \left[\sup_{h \in \mathcal{H}} \Big| \sum_{i=1}^{m} \sigma_i h(x_i) \Big| \right].$$

The expectation is taken over $\sigma = (\sigma_1, \dots, \sigma_m)$, where, $\sigma_i \in \{\pm 1\}$ are i.i.d. and uniformly distributed samples.

The Rademacher complexity measures the ability of a class of functions to fit noise. The empirical Rademacher complexity has the added advantage that it is data-dependent and can be measured from finite samples. It can lead to tighter bounds than those based on other measures of complexity, such as the VC-dimension (Koltchinskii and Panchenko, 2000).
The following theorem introduces a risk bound for unsupervised cross-domain mapping.

Theorem 1 (Cross-Domain Mapping with IPMs) *Assume that $\mathcal{X}_A \subset \mathbb{R}^N$ and $\mathcal{X}_B \subset \mathbb{R}^M$ are convex and bounded sets. Let $\mathcal{H}$ be the hypothesis class and $\mathcal{C}$ the class of discriminators. Assume that $\mathcal{C} \subset C^2$ and $\sup_{d \in \mathcal{C}} \|d\|_{\infty, \mathcal{X}_B} < \infty$. Then, for any $\delta \in (0,1)$, with probability at least $1-\delta$ over the selection of $\mathcal{S}_A \sim D_A^{m_1}$ and $\mathcal{S}_B \sim D_B^{m_2}$, for every $\omega \in \Omega$ and $h_1 \in \mathcal{P}_\omega := \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$, we have:*

$$\begin{aligned} R_{D_A}[h_1, y] \lesssim & \sup_{h_2 \in \mathcal{P}_\omega} R_{\mathcal{S}_A}[h_1, h_2] + c \inf_{h \in \mathcal{P}_\omega} \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B) + \inf_{h \in \mathcal{P}_\omega} \inf_{\substack{d \in \mathcal{C} \\ \beta(d) \leq 1}} \mathcal{K}(h, d; y) \\ & + \hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + \hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{C} \circ \mathcal{H}) + \hat{\mathscr{R}}_{\mathcal{S}_B}(\mathcal{C}) + \sqrt{\frac{\log(1/\delta)}{\min(m_1, m_2)}}. \end{aligned} \tag{4}$$

Here, $\mathcal{K}(h, d; y) := \mathbb{E}_{x \sim D_A}\left[\|\nabla_{y(x)} d(y(x)) - (h(x) - y(x))\|_2\right]$.

The proof of this theorem can be found in Section A of the Appendix.

5.2 Analyzing the Bound

Theorem 1 provides an upper bound on the generalization risk, $R_{D_A}[h_1, y]$, of a hypothesis h_1 that was selected by an algorithm $\mathcal{A}_\omega$, which is the argument that we would like to minimize.
This bound is decomposed into four parts. The first term, $\sup_{h_2 \in \mathcal{P}_\omega} R_{\mathcal{S}_A}[h_1, h_2]$, measures the maximal distance between h_1 and a second candidate $h_2 \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$. The second and third terms behave as approximation errors. The second term, $c \cdot \inf_{h \in \mathcal{P}_\omega} \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)$, measures the discrepancy between the distributions $h \circ \mathcal{S}_A$ and $\mathcal{S}_B$ for the best fitting hypothesis $h \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$. This is captured by the $\mathcal{C}$-IPM between $h \circ \mathcal{S}_A$ and $\mathcal{S}_B$. The coefficient $c \geq 1$ is the tolerance constant defined in Section 4.4. In Section 6.1, we show how these terms are being estimated.
The fourth part (including three Rademacher complexities and the square root) is a result of the transition from empirical to expected quantities. It consists of the empirical Rademacher complexities of the classes $\ell_{\mathcal{H}}$, $\mathcal{C} \circ \mathcal{H}$ and $\mathcal{C}$ and the term $\sqrt{\frac{\log(1/\delta)}{\min(m_1, m_2)}}$. These terms are standard when considering generalization bounds. When these classes have a finite pseudo-dimension, which is the typical case of neural networks, their corresponding Rademacher complexities are of order $\mathcal{O}\left(\sqrt{\log(m_i)/m_i}\right)$ (Mohri et al., 2018). Several publications, for example, (Bartlett et al., 2017; Golowich et al., 2018), have shown that the Rademacher complexity of neural networks is proportional to the spectral norm of the neural networks. For simplicity, we neglect these terms since we focus on the conditions for solving the unsupervised learning task, rather than on the generalization capabilities of the algorithm.
The third term, $\mathcal{K} := \inf_{h,d} \mathcal{K}(h, d; y)$, serves as a mutual approximation error of the classes $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$ and $\nabla\mathcal{C} := \{\nabla d \mid d \in \mathcal{C}\}$. We note that if the zero function $d_0 \equiv 0$ is a member of $\mathcal{C}$,

then, $\mathcal{K}(h, d_0; y) \leq \mathbb{E}_{x\sim D_A}[\|h(x) - y(x)\|_2]$ and when considering setting (2) in Section 4.4, we have:

$$\begin{aligned} R_{D_A}[h_1, y] \lesssim & \sup_{h_2 \in \mathcal{P}_k} R_{\mathcal{S}_A}[h_1, h_2] + \inf_{h \in \mathcal{P}_k} \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B) + \inf_{h \in \mathcal{P}_k} \mathbb{E}_{x\sim D_A}[\|h(x) - y(x)\|_2] \\ & + \hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + \hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{C} \circ \mathcal{H}) + \hat{\mathscr{R}}_{\mathcal{S}_B}(\mathcal{C}) + \sqrt{\frac{\log(1/\delta)}{\min(m_1, m_2)}}, \end{aligned} \tag{5}$$

which is essentially the bound in (Benaim et al., 2018). The main disadvantage of Equation (5) follows from the fact that the bound is tight only when $\inf_{h\in\mathcal{P}_k} \mathbb{E}_{x\sim D_A}[\|h(x) - y(x)\|_2]$ is small, that is, there is a good approximator h of y. We note that for a large enough depth k, we expect $\inf_{h\in\mathcal{P}_k} \mathbb{E}_{x\sim D_A}[\|h(x) - y(x)\|_2]$ to be small. However, for larger values of k, we also expect $\sup_{h_2\in\mathcal{P}_k} R_{\mathcal{S}_A}[h_1, h_2]$ to be larger. This is especially crucial, as it indicates that the bound in (Benaim et al., 2018) is effective only when the term $\inf_{h\in\mathcal{P}_k} \mathbb{E}_{x\sim D_A}[\|h(x) - y(x)\|_2]$ is small for small values of k. This property is termed "Occam's razor property" by Benaim et al. (2018). On the other hand, in general, the term $\mathcal{K}$ in Theorem 1 decreases when increasing the capacity of $\mathcal{C}$. In particular, for a wide class of discriminators $\mathcal{C}$, we do not have to assume the existence of a particularly good approximator $h \in \mathcal{P}_k(\mathcal{S}_A, \mathcal{S}_B)$ of y in order to guarantee that the value of $\mathcal{K}$ is small, in contrast to the analysis in (Benaim et al., 2018). Therefore, our bound does not rely on the assumption that y can be approximated by small depth networks. In Section 8.3, we empirically compare between the two terms.

Finally, when assuming that $\inf_{h,d} \mathcal{K}(h, d; y)$ is small and neglecting the generalization gap terms, we informally have:

$$R_{D_A}[h_1, y] \lesssim \sup_{h_2 \in \mathcal{P}_\omega} R_{\mathcal{S}_A}[h_1, h_2] + c \inf_{h \in \mathcal{P}_\omega} \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B). \tag{6}$$

This inequality is a cornerstone in the derivation of the predictions and algorithms for cross-domain mapping presented in Section 6.

6. Consequences of the Bound

Theorem 1 leads to a concrete prediction, when applied to setting (2) in Section 4.4. The prediction is verified in Section 8.

We believe that a crucial part in the success of the recent methods results from selecting the architecture used in an appropriate way. For example, DiscoGAN (Kim et al., 2017) employs either eight or ten layers, depending on the data set. We make the following prediction:

Prediction 1 *The term* $\inf_{h\in\mathcal{P}_k} \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)$ *decreases as k increases and* $\sup_{h_2\in\mathcal{P}_k} R_{\mathcal{S}_A}[h_1, h_2]$ *increases as k increases. Therefore, to make both of the terms small, it is preferable to select the minimal depth $k \in \mathbb{N}$ that provides a hypothesis $h \in \mathcal{H}_k$ that has a small $\rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)$.*

According to this prediction, the strongest clue that helps identify the alignment of the semantic mapping from the other mappings, is the suitable depth of the network that is learned. A network of depth that is too low cannot replicate the target distribution, when taking inputs in the source domain (high mapping discrepancy). A network that is too deep, would not learn the target mapping, since it could be distracted by other alignment solutions. As shown in Section 4.2, this ambiguity exists regardless of the size of the training data.

This is surprising because in supervised learning, extra depth is not as detrimental, as long as the training data set is large enough. As far as we know, this is the first time that this clear distinction between supervised and unsupervised learning is made[2].

6.1 Estimating the Ground Truth Error

As mentioned in Section 5.2, the expression in Equation (6) provides us with an approximate version of the upper bound in Theorem 1,

$$\sup_{h_2 \in \mathcal{P}_k} R_{\mathcal{S}_A}[h_1, h_2] + c \inf_{h \in \mathcal{P}_k} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B), \tag{7}$$

where $\mathcal{P}_k = \mathcal{P}_k(\mathcal{S}_A, \mathcal{S}_B)$ is chosen to be $\mathcal{P}_k := \{h \in \mathcal{H}_k \mid \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) \leq c\rho^*_k\}$, for some predefined tolerance parameter $c \geq 1$ (see Section 4.4) and $\rho^*_k := \inf_{h \in \mathcal{H}_k} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$.
In this section, we would like to estimate an upper bound of the expression in Equation (7). For this purpose, we will show how to estimate an upper bound of $\sup_{h_2 \in \mathcal{P}_k} R_{\mathcal{S}_A}[h_1, h_2]$ and compute an upper bound of $\inf_{h \in \mathcal{P}_k} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$.
To upper bound the second term, we note that for any fixed $k \in \mathbb{N}$, by the definition of $\mathcal{P}_k$, we have,

$$\inf_{h \in \mathcal{P}_k} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) = \rho^*_k,$$

which is a constant. To estimate this term, we train a hypothesis $h \in \mathcal{H}_k$ to minimize $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$, as discussed in Section 3.2. Since this is a non-convex optimization problem, there is no guarantee to perfectly minimize $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$. Therefore, this process only gives us an upper bound $\hat{\rho}^*_k := \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$ of the second term $\rho^*_k = \inf_{h \in \mathcal{H}_k} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$.
Next, we would like to show how to estimate the first term. Namely, for each h_1, we would like to approximately solve the following objective:

$$\max_{h_2} R_{\mathcal{S}_A}[h_1, h_2] \text{ s.t: } h_2 \in \mathcal{P}_k, \tag{8}$$

or alternatively,

$$\max_{h_2 \in \mathcal{H}_k} R_{\mathcal{S}_A}[h_1, h_2] \text{ s.t: } \rho_C(h_2 \circ \mathcal{S}_A, \mathcal{S}_B) \leq c\rho^*_k. \tag{9}$$

We do not know the value of the value of ρ^*_k explicitly, instead, we consider the following relaxed version of Equation (9):

$$\max_{h_2 \in \mathcal{H}_k} R_{\mathcal{S}_A}[h_1, h_2] \text{ s.t: } \rho_C(h_2 \circ \mathcal{S}_A, \mathcal{S}_B) \leq c\hat{\rho}^*_k. \tag{10}$$

We note that any $h_2 \in \mathcal{H}_k$ that satisfies the condition in Equation (9) also satisfies the condition in Equation (10), since $\rho^*_k \leq \hat{\rho}^*_k$. Therefore, the value in Equation (10) is an upper bound on the value in Equation (8). To summarize, if h^*_2 is the solution to Equation (10), the expression $R_{\mathcal{S}_A}[h_1, h^*_2] + c\hat{\rho}^*_k$ upper bounds the RHS in Equation (7). Finally, in order to train h_2, inspired by Lagrange relaxation, we employ the following relaxed version of it:

$$\min_{h_2 \in \mathcal{H}_k} \Big\{\rho_C(h_2 \circ \mathcal{S}_A, \mathcal{S}_B) - \lambda R_{\mathcal{S}_A}[h_1, h_2]\Big\}. \tag{11}$$

[2] The minimum description length (MDL for short) literature was developed when people believed that small hypothesis classes are desired for both supervised and unsupervised learning.

Algorithm 1 Early stopping

```
Require: $\mathcal{S}_A$ and $\mathcal{S}_B$: training samples; $\mathcal{H}$: a hypothesis class; $\mathcal{C}$: a class of discriminators; $c$: a
    tolerance scale; $k$: maximal depth; $\lambda$: a trade-off parameter; $T$: a fixed number of epochs.
1: Initialize $h_1^0 \in \mathcal{H}_k$ and $h_2^0 \in \mathcal{H}_k$ at random.
2: Train a hypothesis $h \in \mathcal{H}_k$ to minimize $\rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)$.
3: Define $\hat{\rho}_k^* := \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)$ (upper bounds $\inf_{h \in \mathcal{H}_k} \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)$).
4: for $t = 1, \dots, T$ do
5:     Train $h_1^{t-1} \in \mathcal{H}_k$ for one epoch to minimize $\rho_{\mathcal{C}}(h_1^t \circ \mathcal{S}_A, \mathcal{S}_B)$, obtaining $h_1^t \in \mathcal{H}_k$.
6:     Train $h_2^{t-1} \in \mathcal{H}$ for one epoch to minimize $\rho_{\mathcal{C}}(h_2^t \circ \mathcal{S}_A, \mathcal{S}_B) - \lambda R_{\mathcal{S}_A}[h_1^t, h_2^t]$, obtaining $h_2^t \in \mathcal{H}_k$.
7: end for
8: return $h_1^t$ such that: $t = \arg\min_{i \in [T_1]} \{R_{\mathcal{S}_A}[h_1^i, h_2^i] \mid \forall j = 1,2 : \rho_{\mathcal{C}}(h_j^i \circ \mathcal{S}_A, \mathcal{S}_B) \leq c\hat{\rho}_k^*\}$.
```

To minimize the term $\rho_{\mathcal{C}}(h_2 \circ \mathcal{S}_A, \mathcal{S}_B)$ in Equation (11), we train h_2 against a discriminator, as discussed in Section 3.2. Throughout the optimization process of h_2, we only keep instances of it if $\rho_{\mathcal{C}}(h_2 \circ \mathcal{S}_A, \mathcal{S}_B) \leq c\hat{\rho}_k^*$ is valid. We note that this process only gives us an estimation of Equation (10), due to the relaxation in Equation (11) and its non-convex nature.

Based on the above analysis, we present a method for early stopping (Algorithm 1). In this algorithm, we iteratively train the mapping h_1 to minimize $\rho_{\mathcal{C}}(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$. This process generates a list of T_1 mappings $\{h_1^i\}_{i=1}^{T_1}$, for each epoch $i \in [T_1]$ during training. In order to pick a well-performing candidate h_1^i from the list, we use the estimation above during training to measure the performance of each h_1^i. For this purpose, we train a second network h_2^i to minimize the objective in Equation (11) (step 6). The two neural networks h_1^i and h_2^i are trained iteratively throughout each epoch. Finally, we choose the candidate h_1^t that minimizes the objective of Equation (10) among the set $\{h_1^i\}_{i=1}^{T_1}$ and satisfies $\forall j = 1,2 :\ \rho_{\mathcal{C}}(h_j^t \circ \mathcal{S}_A, \mathcal{S}_B) \leq c\hat{\rho}_k^*$ (step 8).

It is worth mentioning that the value of c is a matter of choice. In principle, we could select $c = 1$ and the bound would still be valid. However, in practice, it is advantageous to take $c > 1$ since it lets us reject fewer candidates h_2.

6.2 Deriving an Unsupervised Variant of Hyperband Using the Bound

In order to optimize multiple hyperparameters ω simultaneously, we create an unsupervised variant of the Hyperband method (Li et al., 2018). In a nutshell, Hyperband is a high-level hyperparameter selection method that searches for a configuration of hyperparameters ω that minimizes a certain objective function $\mathcal{L}(\omega)$. For this purpose, Hyperband requires the ability to evaluate the objective function for every configuration of hyperparameters. This is done using a plug-in function, called 'run_then_return_val_val_loss' (see Algorithm 1, Li et al. 2018), that evaluates the objective function for a given configuration. In our case, the objective function is the risk function, $R_{D_A}[h_1, y]$, where h_1 is a mapping that was trained to minimize $\rho_{\mathcal{C}}(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$ with the hyperparameters ω. Since we cannot evaluate the actual risk, we replace it with an estimated value of the following expression:

$$\sup_{h_2 \in \mathcal{P}_\omega} R_{\mathcal{S}_A}[h_1, h_2] + \rho_{\mathcal{C}}(h_1 \circ \mathcal{S}_A, \mathcal{S}_B), \tag{12}$$

This expression differs from the original bound (Equation 4) in two ways. First, we neglect the term $\mathcal{K}$ as we already explained in Section 5.2 that it tends to be small. In addition, the term $\inf_{h \in \mathcal{P}_\omega} \rho_{\mathcal{C}}(h \circ$

Algorithm 2 Unsupervised run_then_return_val_loss for Hyperband

```
Require: $\mathcal{S}_A$ and $\mathcal{S}_B$: training samples; $\lambda$: a trade-off parameter; $T$: number of epochs; $\omega$: set of hyperparameters.
[$h_1, h_2$, $T_{\text{last}}$] = return_stored_functions($\omega$)
Train $h_1 \in \mathcal{H}$ for $T - T_{\text{last}}$ epochs to minimize $\rho_C(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$ with hyperparameters $\omega$.
Train $h_2 \in \mathcal{H}$ for $T - T_{\text{last}}$ epochs to minimize $\rho_C(h_2 \circ \mathcal{S}_A, \mathcal{S}_B) - \lambda R_{\mathcal{S}_A}[h_1, h_2]$ with hyperparameters $\omega$.
store_functions($\omega$, [$h_1, h_2, T$])
return $R_{\mathcal{S}_A}[h_1, h_2] + \rho_C(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$.
```

$\mathcal{S}_A, \mathcal{S}_B)$ is replaced with $\rho_C(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$, which can be easily estimated using a discriminator (see Section 3.2). This term can fit as a good replacement, since $\inf_{h \in \mathcal{P}_\omega} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) \leq \rho_C(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$. Secondly, we neglect the terms that arise from the transition from a population distribution to finite sample sets, as they are constant and small for large enough m_1 and m_2. In addition, we choose ω to include: the depth of the trained network, batch size, learning rate, etc'. In order to estimate $\sup_{h_2 \in \mathcal{P}_\omega} R_{\mathcal{S}_A}[h_1, h_2]$, we train a second network h_2 to minimize $\rho_C(h_2 \circ \mathcal{S}_A, \mathcal{S}_B) - \lambda R_{\mathcal{S}_A}[h_1, h_2]$ with the hyperparameters given by ω, using the analysis provided in Section 6.1.

In particular, the function 'run_then_return_val_loss' in the Hyperband algorithm (see Algorithm 1, Li et al. 2018), is provided with our estimated value of the expression in Equation (12). Our variant of this function is listed in Algorithm 2. It employs two additional procedures that are used to store the learned models h_1 and h_2 at a certain point in the training process and to retrieve these to continue the training for a large number of epochs. The retrieval function is simply a map between a vector of hyperparameters and a tuple of the learned networks and the number of epochs T when stored. For a new vector of hyperparameters, it returns $T = 0$ and two randomly initialized networks, with architectures that are determined by the given set of hyperparameters. When a network is retrieved, it is then trained for a number of epochs that is the difference between the required number of epochs T, which is given by the Hyperband method, and the number of epochs it was already trained, denoted by T_{last}.

7. The Non-Unique Case

In various cases, there are multiple unknown target functions from A to B, that is, there is a set $\mathcal{T}$ of alternative target functions y. For example, the domain $\mathcal{X}_A$ is a set of images of shoe edges and $\mathcal{X}_B$ is a set of images of shoes. There are multiple mappings that take edges of a shoe and return a pair of shoes that fit these edges (each mapping colors the shoes in a different way). It is important to note that $\mathcal{T}$ contains only a subset of the alternative mappings between A and B. For instance, in the edges to shoes example, there are mappings that take edges and return a shoe that does not fit these edges.

As before, the cross-domain mapping algorithm is provided with access to the distributions D_A and D_B. In this case, the goal of the algorithm is to return a hypothesis $h_1 \in \mathcal{H}$ that is close to one of the target functions $y \in \mathcal{T}$, i.e., minimizes $\inf_{y \in \mathcal{T}} R_{D_A}[h_1, y]$.

The bound in Theorem 1 can be readily extended to the non-unique case, by simply taking $\inf_{y \in \mathcal{T}}$ on both sides of the inequality in Theorem 1. This results in an upper bound on $\inf_{y \in \mathcal{T}} R_{D_A}[h_1, y]$,

instead of $R_{D_A}[h_1, y]$ for a specific target function y. In addition, similar to Section 5.2, we can extend the assumption that $\inf_{h,d} \mathcal{K}(h, d; y)$ is small for the term $\inf_{y \in \mathcal{T}} \inf_{h,d} \mathcal{K}(h, d; y)$.
However, in the non-unique case, this bound might not be tight, even for successful cross-domain mapping algorithms. For instance, since there are multiple possible target functions $y \in \mathcal{T}$ and the term $\sup_{h_2 \in \mathcal{P}_\omega} R_{\mathcal{S}_A}[h_1, h_2]$ can be large, if $h_1 \approx y_1$ and $h_2 \approx y_2$, where $y_1, y_2 \in \mathcal{T}$, such that, $y_1 \neq y_2$ and $h_1, h_2 \in \mathcal{P}_\omega$. On the other hand, in this case we have: $\inf_{y \in \mathcal{T}} R_{D_A}[h_i, y] \approx 0$ (for $i = 1, 2$). A tighter bound would result, if we are able to select $\omega \in \Omega$ that minimizes the bound, as we show next.

7.1 Equivalence Classes According to a Fixed Encoder

To deal with the problem discussed above, we take an encoder-decoder perspective of each target function $y \in \mathcal{T}$, such that, by fixing the first layers of the mapping y, the ambiguity between these functions vanishes, and only one possible mapping remains, e.g., the function is determined by the encoder part.
To formalize this idea, we take a hypothesis class $\mathcal{H} := \{h_{\theta,\omega} = g_\theta \circ f_\omega \mid \theta \in \Theta, \omega \in \Omega\}$ consisting of hypotheses that are parameterized by two sets of parameters $\theta \in \Theta$ and $\omega \in \Omega$. In addition, we denote $\mathcal{H}_\omega := \{h_{\theta,\omega} \mid \theta \in \Theta\}$. Specifically, $\mathcal{H}$ serves as a set of neural networks of a fixed architecture with $l_1 + l_2$ layers. Each hypothesis $h_{\theta,\omega}$ is a neural network of an encoder-decoder architecture. The encoder, f_ω, consists of the first l_1 layers and the decoder, g_θ, consists of the last l_2 layers. In addition, ω and θ denote the sets of weights of f_ω and g_θ (resp.). We take $\mathcal{A}_\omega$ that returns a hypothesis h from $\mathcal{H}_\omega$ that achieves $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) \leq c \inf_{h^* \in \mathcal{H}} \rho_C(h^* \circ \mathcal{S}_A, \mathcal{S}_B)$ and $\mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$ is defined accordingly (see Section 4.4).
This leads to an extended version of Algorithm 1 for the non-unique case. Similar to Section 6.1, as a first step, we produce an upper bound estimation $\hat{\rho}^*$ of $\rho^* := \inf_{h \in \mathcal{H}} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$ by training a hypothesis $h \in \mathcal{H}$ to minimize $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$, as detailed in Section 3.2. We use the class $\hat{\mathcal{P}}_\omega := \{h \mid \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B) \leq c\hat{\rho}^*\}$ as an approximation of the set $\mathcal{P}_\omega$.
We let $h_1 = g_{\theta_1} \circ f_\omega$ and $h_2 = g_{\theta_2} \circ f_\omega$ be two neural networks with the same encoder-decoder architecture with shared parameters ω (encoder) and un-shared parameters θ_1 and θ_2 (decoder). The decoder of h_1 (parameterized by θ_1) is trained to minimize $\rho_C(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$ (step 7). The decoder of h_2 (parameterized by θ_2) is trained to maximize $R_{\mathcal{S}_A}[h_1, h_2]$ and to minimize $\rho_C(h_2 \circ \mathcal{S}_A, \mathcal{S}_B)$, since we would like to find a function $h_2 \in \hat{\mathcal{P}}_\omega$ that maximizes $R_{\mathcal{S}_A}[h_1, h_2]$ (step 8). In order to guarantee that $h_1, h_2 \in \hat{\mathcal{P}}_\omega$, we reject instances of h_1 and h_2, for which, $\forall i = 1, 2 : \ \rho_C(h_i \circ \mathcal{S}_A, \mathcal{S}_B) \leq c\hat{\rho}^*$ is invalid (step 10). Finally, we would like to train a shared encoder f_ω, such that, the decoders would be able to map between the two distributions and will yield a low maximal risk between them. For this purpose, the parameters ω are trained to minimize the following objective $R_{\mathcal{S}_A}[h_1, h_2] + \rho_C(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$ (step 6). Since θ_2 is trained to maximize $R_{\mathcal{S}_A}[h_1, h_2]$ for $h_2 \in \hat{\mathcal{P}}_\omega$, we can think of the optimization of ω as a method for minimizing the expression $\sup_{h_2 \in \hat{\mathcal{P}}_\omega} R_{\mathcal{S}_A}[h_1, h_2] + \rho_C(h_1 \circ \mathcal{S}_A, \mathcal{S}_B)$.

8. Experiments

The first group of experiments is intended to test the validity of the prediction made in Section 6. The next group of experiments is dedicated to Algorithms 1, 2 and 3.
We note that our algorithms (Algorithms 1, 2 and 3) are widely applicable, since they can be seen as high-level improvements over pre-existing cross-domain mapping algorithms. In order to demonstrate this, we run our experiments using a wide variety of GAN-based unsupervised cross-

Algorithm 3 Early stopping (non-unique case)

```
Require: $\mathcal{S}_A$ and $\mathcal{S}_B$: training samples; $\mathcal{H}$: a hypothesis class; $\lambda$: a trade-off parameter; $T_0$: a fixed number of epochs for $\omega$; $T$: a fixed number of epochs for $\theta_1$ and $\theta_2$.
Train a hypothesis $h \in \mathcal{H}$ to minimize $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$.
Define $\hat{\rho}^* := \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$ (upper bounds $\inf_{h \in \mathcal{H}} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$).
Initialize the shared parameters $\omega_0 \in \Omega$ at random.
Initialize the parameters $\theta_{1,0}, \theta_{2,0} \in \Theta$ of $h_1$ and $h_2$ (resp.) at random.
for $t = 1, \dots, T_0$ do
    Train $\omega_{t-1}$ for one epoch to minimize $R_{\mathcal{S}_A}[h_1^{t-1}, h_2^{t-1}] + \rho_C(h_1^{t-1} \circ \mathcal{S}_A, \mathcal{S}_B)$, obtaining $\omega_t \in \Omega$.
    Train $\theta_{1,t-1} \in \Theta$ for $T$ epochs to minimize $\rho_C(h_1^t \circ \mathcal{S}_A, \mathcal{S}_B)$, obtaining $\theta_{1,t} \in \Theta$.
    Train $\theta_{2,t-1} \in \Theta$ for $T$ epochs to minimize $\rho_C(h_2^t \circ \mathcal{S}_A, \mathcal{S}_B) - \lambda R_{\mathcal{S}_A}[h_1^t, h_2^t]$, obtaining $\theta_{2,t} \in \Theta$.
    ▷ Here, $h_i^t := g_{\theta_{i,t-1}} \circ f_{\omega_{t-1}}$ for $i = 1, 2$.
end for
Define $t := \arg\min_{i \in [T_0]} \{R_{\mathcal{S}_A}[h_1^i, h_2^i] + \rho_C(h_1^i \circ \mathcal{S}_A, \mathcal{S}_B) \mid \forall j = 1, 2 : \rho_C(h_j^i \circ \mathcal{S}_A, \mathcal{S}_B) \le c\hat{\rho}^*\}$.
return $h_1^t$.
```

domain mapping algorithms, including CycleGAN (Zhu et al., 2017), DiscoGAN (Kim et al., 2017), DistanceGAN (Benaim and Wolf, 2017), UNIT (Liu et al., 2017) and WGAN (Arjovsky et al., 2017).

8.1 Empirical Validation of Prediction 1

In this prediction, we claim that the selection of the number of layers k is crucial in unsupervised learning. Using fewer layers than needed, will not support a small mapping discrepancy, that is, $\inf_{h \in \mathcal{P}_k} \rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$ is large. In contrast, adding superfluous layers would mean that there exist many alternative functions in $\mathcal{H}_k$ that map between the two domains, that is, $\sup_{h_2 \in \mathcal{P}_k} R_{\mathcal{S}_A}[h_1, h_2]$ is large.

To see the influence of the number of layers of the generator h_1 on the results, we employed the DiscoGAN (Kim et al., 2017) official public implementation and added or removed layers from the generator. The experiment was done on the CelebA data set, where 8 layers are employed in the experiments of Kim et al. (2017).

The results for male to female conversion are visually illustrated in Figure 3. Note that since the encoder and the decoder parts of the learned network are symmetrical, the number of layers is always even. As can be seen visually, changing the number of layers has a dramatic effect on the results. The best results are obtained at 6 or 8 layers, with 6 having the best alignment and 8 having better discrepancy. The results degrade quickly, as one deviates from the optimal value. Using fewer layers, the GAN fails to produce images of the desired class. Adding layers, the semantic alignment is lost, just as expected. The experiment is repeated for both CycleGAN (Zhu et al., 2017) and WGAN (Bojanowski et al., 2018) for other data sets in Figures 10-12 (Appendix B).

To quantitatively validate the prediction, we trained a mapping h from A to B using DiscoGAN with and without circularity losses and measured the expected VGG similarity between its input and output, that is, $\mathbb{E}_{x \sim D_A}[cs(f(x), f(h(x)))]$. The VGG descriptor similarity between two images x_1 and x_2 computes $cs(f(x_1), f(x_2))$, where cs is the cosine similarity function and $f(x)$ is a deep layer in the VGG network. Since the VGG network was trained to identify the content of a wide variety of classes, these vector representations, $f(x)$, are treated as compressed content descriptors of the images and $cs(f(x_1), f(x_2)) \in [-1, 1]$ as the degree of content similarity between the images x_1 and x_2. The higher $cs(f(x_1), f(x_2))$ is, the more similar we consider x_1 and x_2 to be. In Table 4 (Appendix B), we report the best VGG similarity and discrepancy values among an extensive parameter search, when trying to change the learning rate (between 10^{-5} to 1), the number of kernels per layer (between 10 and 300), and the weight between circularity losses and the GANs (between 10^{-5} and 1).

As can be seen in Table 4 (Appendix B), when varying the number of layers, both the discrepancy and the VGG similarity decrease (with or without the circularity losses). It is not surprising that the discrepancy decreases, since when increasing the depth of the network, it can capture the target distribution better. However, the decreasing value of the VGG similarity indicates that the alignment is lost, as the mapping generates images that are not similar to the input images. This validates Prediction 1, since we can see that the number of layers has a dramatic effect on the results.

While the depth seems to be highly related to the quality of the results, the norm of the weights do not seem to point to a clear architecture, as shown in Table 5(a) (Appendix B). Since the table compares the norms of architectures of different sizes, we also approximated the functions using networks of a fixed depth $k = 18$ and then measured the norm. These results are presented in Table 5(b) (Appendix B). In both cases, the optimal depth, which is 6 or 8, does not appear to have a be an optimum in any of the measurements.

8.1.1 EXPERIMENTS WITHOUT THE CIRCULARITY LOSSES

To further demonstrate our observation, we conducted a series of experiments for comparing the performance of circularity-based methods (Equation 3) with and without the circularity losses (that is, by only minimizing $\rho_C(h \circ \mathcal{S}_A, \mathcal{S}_B)$). We empirically observe that one can achieve comparable results with and without the circularity losses.

This observation has been partially verified in (Zhu et al., 2017). In their ablation study (Tables 4-5), it is shown that the performance of CycleGAN without the circularity losses is slightly worse than the performance of CycleGAN on the Cityscapes data set (Cordts et al., 2016). The evaluation is done using the the standard metrics from the Cityscapes benchmark (Cordts et al., 2016), including per-pixel accuracy, per-class accuracy, and mean class Intersection-Over-Union (Class IOU) (Cordts et al., 2016). In Figure 2(b) of (Liu et al., 2017), it is shown that the translation accuracy of UNIT is slightly worse when training it without its cycle consistency losses on the Maps data set (Isola et al., 2017).

As can be seen in Table 4 (Appendix B), the results (e.g., discrepancy and VGG similarity scores) of DiscoGAN with and without the circularity losses are fairly similar when varying the number of layers. The results for hair color conversions are shown in Figure 2. It is evident that the output image is closely related to the input images, despite the fact that circularity loss terms were not used.

As an additional experiment, we compared the performance of three variations of the CycleGAN method: with the circularity and GAN losses, without the circularity losses and without the GAN losses. We also compare them with the identity mapping $h(x) = x$. We used the official public implementation with the standard ResNet generator with the default hyperparameters. In this experiment, we trained the models on two data sets: (i) aerial photographs to maps, using data scraped from Google Maps (Isola et al., 2017) and (ii) architectural photographs to their labels from the CMP Facades data set (Radim Tyleček, 2013). We used a 6-block ResNet generator on the Maps dataset and a 4-block ResNet on the Facades dataset. These data sets are unimodal, meaning, that

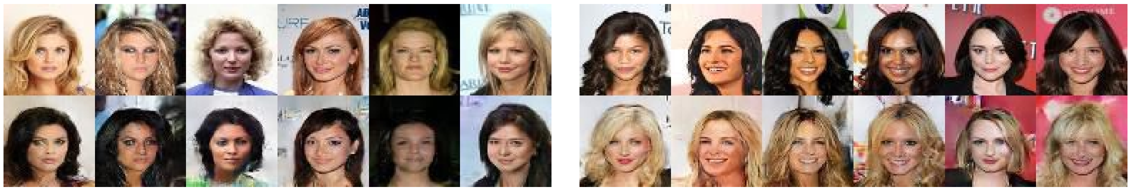


Figure 2: Results for the celebA data set for converting blond to black hair and vice versa, when the mapping is obtained by the GAN loss without additional losses.

paired matches $(x, y(x))$ exist in the data set, even though the learning algorithm is not provided with them.

We measure the performance of each generator h using three different scores. The first score is the expected VGG similarity between $h(x)$ and $y(x)$, that is, $\mathbb{E}_{x\sim D_A}[cs(f(h(x)), f(y(x)))]$. The second and third scores are based on the expected Learned Perceptual Image Patch Similarity (LPIPS) metric (Zhang et al., 2018) between $h(x)$ and $y(x)$. For a given pretrained neural network F of depth L, the LPIPS of two images x_1 and x_2 is computed as the average over the cosine distance of the activations of F across various layers, that is, LPIPS$(x_1, x_2) := \frac{1}{L}\sum_{l=1}^{L} cd(F_l(x_1), F_l(x_2))$, where cd is the cosine distance function and F_l is the l'th layer of F. We measured the LPIPS once using a pretrained VGG network and once with a pretrained AlexNet. The lower LPIPS(x_1, x_2) is, the more similar we consider x_1 and x_2 to be. We used the official public implementation of LPIPS and the networks were pretrained on ILSVRC2012 (Russakovsky et al., 2015).

The scores are averaged over 20 trials and the expectations $\mathbb{E}_{x\sim D_A}$ are estimated using the test data. As can be seen in Table 2 (Appendix B), the performance of CycleGAN without the circularity losses are comparable to those of CycleGAN with the circularity losses (are slightly worse on the Maps data set and are slightly better on the Facades data set). On the other hand, the results of CycleGAN without the GAN losses and the identity function are significantly worse than the results of CycleGAN with or without the circularity losses. In Figure 13 (Appendix B), we observe that CycleGAN without the circularity losses learns an approximation of the target function in both directions.

8.2 Results for Algorithms 1, 2 and 3

We test the three algorithms on three unsupervised alignment methods: DiscoGAN (Kim et al., 2017), CycleGAN (Zhu et al., 2017), and DistanceGAN (Benaim and Wolf, 2017). In DiscoGAN and CycleGAN, we train h_1 (and h_2), using two GANs with two circularity constraints; in DistanceGAN, to train h_1 (and h_2), one GAN and one distance correlation loss are used. The published hyperparameters for each data set are used, except when using Hyperband, where we vary the number of layers, the learning rate and the batch size.

Five data sets were used in the experiments: (i) aerial photographs to maps, using data scraped from Google Maps (Isola et al., 2017), (ii) the mapping between photographs from the cityscapes data set and their per-pixel semantic labels (Cordts et al., 2016), (iii) architectural photographs to their labels from the CMP Facades data set (Radim Tyleček, 2013), (iv) handbag images (Zhu et al., 2016) to their binary edge images, as obtained from the HED edge detector (Xie and Tu, 2015), and (v) a similar data set for the shoe images from (Yu and Grauman, 2014).

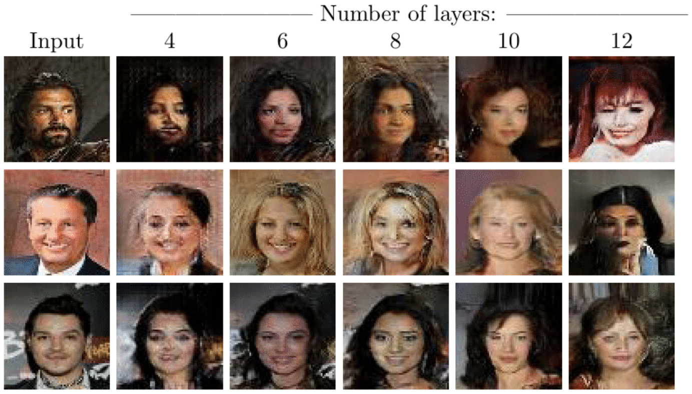


Figure 3: **Varying the number of layers of the generator.** Results of DiscoGAN on CelebA Male to Female transfer. The best results are obtained for 6 or 8 layers. For more than 6 layers, the alignment is lost.

Throughout the experiments of Algorithm 1, fixed values are used as the tolerance hyperparameter ($c = 2$).

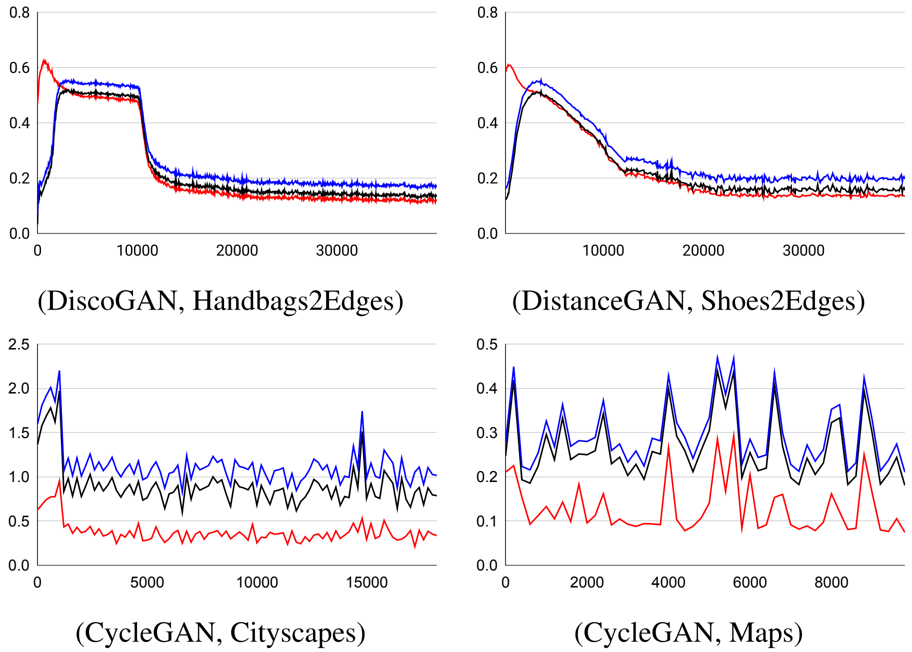
8.2.1 EARLY STOPPING CRITERION (ALGORITHM 1)

For testing the early stopping criterion suggested in Algorithm 1, we ran DiscoGAN (Kim et al., 2017), DistanceGAN (Benaim and Wolf, 2017) and CycleGAN (Zhu et al., 2017) with the default hyperparameters in the papers and compared, three scores, at each time point. The first is an estimated value of $\sup_{h_2\in\mathcal{P}_k} R_{\mathcal{S}_A}[h_1,h_2] + c\hat{\rho}^*_k + \inf_{h\in\mathcal{P}_k}\inf_{d\in C,\ \beta(d)\leq 1}\mathcal{K}(h,d;y)$, which is our bound in Theorem 1, when neglecting the generalization gap terms. Here, $\rho^*_k := \inf_{h\in\mathcal{P}_k}\rho_C(h\circ\mathcal{S}_A,\mathcal{S}_B)$ is replaced with its upper bound $\hat{\rho}^*_k$ (see Section 6.1), $\sup_{h_2\in\mathcal{P}_k} R_{\mathcal{S}_A}[h_1,h_2]$ is estimated according to Section 6.1 and the estimation of $\inf_{h,d}\mathcal{K}(h,d;y)$ is described in Section 8.3. The second is an estimated value of $\sup_{h_2\in\mathcal{P}_k} R_{\mathcal{S}_A}[h_1,h_2] + c\hat{\rho}^*_k$, which is our bound, excluding the term $\inf_{h,d}\mathcal{K}(h,d;y)$ and the generalization gap terms. The third is the ground-truth error, $R_{D_A}[h_1,y] = \mathbb{E}_{x\sim D_A}[\ell(h(x),y(x))]$, where y is the ground-truth mapping and the expected value is taken with respect to the test data set.

The results are depicted in the main results table (Table 3) as well as in Figure 4 for DiscoGAN, DistanceGAN and CycleGAN.

Table 3 presents the correlation and p-value between the ground-truth error, as a function of the training iteration, and the estimated value of the bound. A high correlation (low p-value) between the estimated value of the bound and the ground-truth error, as a function of the iteration, indicates the validity of the bound and the utility of the algorithm. Similar correlations are shown with the GAN losses and the reconstruction losses (DiscoGAN and CycleGAN) or the distance correlation loss (DistanceGAN), in order to demonstrate that these are much less correlated with the ground-truth error. In Figure 4, we omit the other scores in order to reduce clutter.

As can be seen, there is an excellent match between the mean ground-truth error of the learned mapping h_1 and the predicted error. No such level of correlation is present when considering the GAN losses or the reconstruction losses (for DiscoGAN and CycleGAN), or the distance correlation



(DiscoGAN, Handbags2Edges) (DistanceGAN, Shoes2Edges)

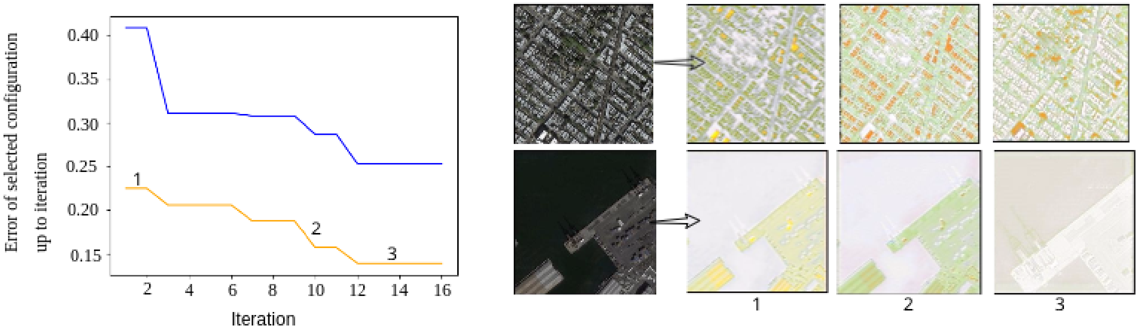
(CycleGAN, Cityscapes) (CycleGAN, Maps)

Figure 4: **Results of Algorithm 1.** Ground-truth errors $R_{D_A}[h_1^t, y]$ are in red, the estimated values of $R_{\mathcal{S}_A}[h_1^t, h_2^t] + c\hat{\rho}_k^*$ and $R_{\mathcal{S}_A}[h_1^t, h_2^t] + c\hat{\rho}_k^* + \mathcal{K}$ are in black and blue respectively ($c = 2$). x-axis is the iteration. y-axis is the expected risk/estimation of the bound. It takes a few epochs for h_1^t to have a small enough discrepancy, until which the bound is ineffective. We ran the algorithms using the default hyperparameters in their papers.

loss of DistanceGAN. Specifically, the very low p-values in the first column of Table 3 shows that there is a clear correlation between the ground-truth error and our bound for all data sets and methods. For the other columns, the values in question are chosen to be the losses used for h_1. The lower scores in these columns show that none of these values are as correlated with the ground-truth error, and so cannot be used to estimate this error.

In the experiment of Algorithm 1 for DiscoGAN, which has a large number of sample points, the cycle from B to A and back to B is significantly correlated with the ground-truth error with very low p-values in four out of five data sets. However, its correlation is significantly lower than that of the estimation of our bound.

These results also demonstrate the tightness of the bound. As can be seen, the bound is always highly correlated with the test error and in most cases, it is tight as well (close to the test error). Bounds that are highly correlated with the test error are very useful, since they faithfully indicate when the test error is smaller.



(a) (b)

Figure 5: **Applying unsupervised Hyperband for selecting the best configuration for UNIT for the Maps data set.** The x-axis is the epoch count and the y-axis is the error/estimated value of Equation (12) of the selected configuration. **(a)** Blue and orange lines are the estimated value of Equation (12) and the ground-truth error, as in Figure 6. **(b)** Images produced for three different configurations, as indicated on the plot in (a).

8.2.2 HYPERPARAMETER SELECTION WITH THE MODIFIED HYPERBAND ALGORITHM (ALGORITHM 2)

Our bound is used in Section 6.2 to create an unsupervised variant of the Hyperband method. In addition to selecting the architecture, this allows for the optimization of multiple hyperparameters at once, while enjoying the efficient search strategy of the Hyperband method (Li et al., 2018).

Figure 6 demonstrates the applicability of our unsupervised Hyperband-based method for different data sets, employing both DiscoGAN and DistanceGAN. The graphs show the error and the estimated value of Equation (12) obtained for the selected configuration after up to 35 Hyperband iterations. As can be seen, in all cases, the method is able to recover a configuration that is significantly better than what is recovered when only optimizing for the number of layers. To further demonstrate the generality of our method, we applied it on the UNIT (Liu et al., 2017) architecture. Specifically, for DiscoGAN and DistanceGAN, we optimize over the number of encoder and decoder layers, batch size and learning rate, while for UNIT, we optimize over the number of encoder and decoder layers, number of resnet layers and learning rate. Figure 5 and Figure 6(b) show the convergence on the Hyperband method.

8.2.3 STOPPING CRITERION FOR THE NON-UNIQUE CASE (ALGORITHM 3)

For testing the stopping criterion suggested in Algorithm 3, we plotted the value of the estimated value of the bound and attached a specific sample for a few epochs. For this purpose, we employed DiscoGAN for both h_1 and h_2, such that the encoder part is shared between them. As we can see in Figures 7– 8, for smaller values of the bound, we obtain more realistic images and the alignment also improves.

8.3 Estimating the Approximation Error Term

We conducted an experiment for validating that the term $\mathcal{K} = \inf_{h\in\mathcal{P}_\omega} \inf_{d\in C,\ \beta(d)\leq 1} \mathcal{K}(h,d;y)$ is small, in comparison with $\mathcal{R} = \inf_{h\in\mathcal{P}_\omega} \mathbb{E}_{x\sim D_A}[\|h(x)-y(x)\|_2]$. In order to estimate $\mathcal{K}$, we trained a generator h and a discriminator d to minimize $\mathcal{K}(h,d;y) = \mathbb{E}_{x\sim D_A}[\|\nabla_{y(x)} d(y(x)) - (h(x)-y(x))\|_2]$ given supervised data $\{(x_i, y(x_i))\}_{i=1}^m$ and similarly we trained a generator h to minimize $\mathbb{E}[\|h(x)-y(y)\|_2]$. In this experiment, the generator's architecture is the standard 4-layer generator of DiscoGAN/ DistanceGAN/ CycleGAN and is trained to minimize $\mathcal{K}(h,d;y)$ or $\mathbb{E}[\|h(x)-y(y)\|_2]$ along with its standard losses (that is, GAN and circularity/distance correlation losses). The discriminator d is trained to minimize $\mathcal{K}(h,d;y)$ along with a constraint to minimize the loss $\frac{1}{m}\sum_{i=1}^m \|\mathrm{H}_d(y(x_i))\|_2$, where $\mathrm{H}_d(x)$ is the Hessian matrix of d at x (see Section 3.1 for details). We plot the values of the term $\mathcal{K}(h,d;y)$ only for d's that satisfy $\frac{1}{m}\sum_{i=1}^m \|\mathrm{H}_d(y(x_i))\|_2 \leq 1$. Finally, the values of $\mathcal{K}(h,d;y)$ and $\mathbb{E}[\|h(x)-y(y)\|_2]$ at the last iteration of the optimization process are used as the estimations of $\mathcal{K}$ and $\mathcal{R}$. The architecture of d consists of four convolutional layers, each one has w channels, kernel size 4, stride size 2 and a padding value 1. The activation function in each layer is Leaky ReLU with slope 0.2. The number of channels is treated as the width of d.

In order to investigate the effect of the size of d on the value of $\mathcal{K}(h,d;y)$, we ran the experiment of discriminators with $w \in \{10, 200, 500\}$. Figure 9 depicts the results of the comparison of the values of the two terms on test data as a function of the number of iterations. As can be seen, the value of $\mathcal{K}$ is significantly smaller than $\mathcal{R}$ for all values of w, and it also significantly decreases as w increases. This behavior is consistent over all iterations.

9. Conclusions

The recent success in mapping between two domains in an unsupervised way and without any existing knowledge, other than network hyperparameters, is nothing less than extraordinary and has far reaching consequences. As far as we know, nothing in the existing machine learning or cognitive science literature suggests that this would be possible.

In Section 5, we derived a novel risk bound for the unsupervised learning of mappings between domains. The bound takes into account the ability of the hypothesis classes (including both the generator and the discriminator) to model the cross-domain mapping task and the ability to generalize from a finite set of samples.

This bound leads directly to a method for estimating the success of the learned mapping between the two domains without relying on a validation set. By training pairs of networks that are distant from each other, we are able to obtain a confidence measure on the mapping's outcome. The confidence estimation has application to hyperparameter selection and for performing early stopping. The bound is extended to the non-unique case mapping case in Section 7.

Acknowledgements

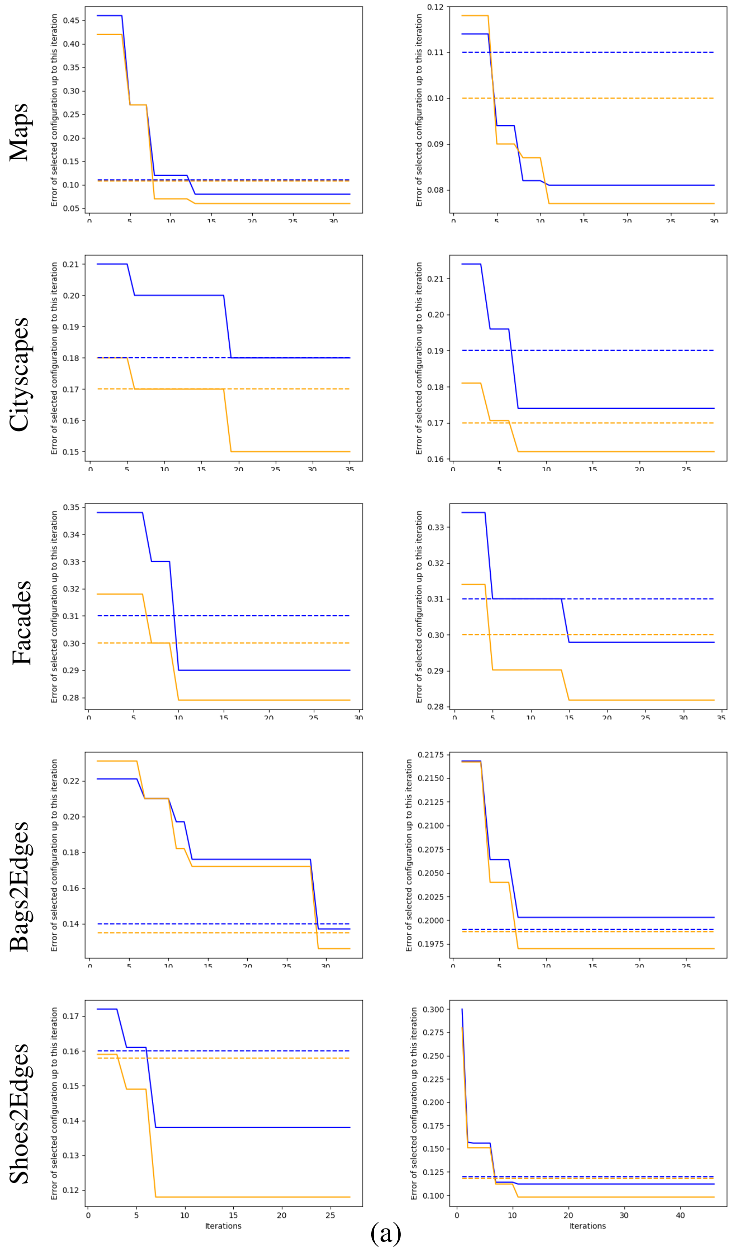
This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant ERC CoG 725974). The contribution of Tomer Galanti is part of a Ph.D. thesis conducted at Tel Aviv University.

	CycleGAN		w.o circ		w.o GAN losses		Identity	
Metric	Maps	Facades	Maps	Facades	Maps	Facades	Maps	Facades
VGG similarity ↑	0.601	0.319	0.585	0.363	0.391	0.275	0.358	0.242
LPIPS (AlexNet) ↓	0.213	0.561	0.235	0.530	0.542	0.696	0.711	0.725
LPIPS (VGG) ↓	0.399	0.654	0.421	0.601	0.594	0.749	0.623	0.804

Table 2: **Comparing the performance of CycleGAN with different loss functions.** We report the classification accuracy, VGG descriptor similarity and LPIPS (with a pretrained AlexNet and VGG network) between $h(x)$ and $y(x)$ for the trained generators h. The generators h are trained using the CycleGAN method with its standard losses (first column), without the circularity losses (second column) and without the GAN losses (third column). The last column specifies the performance of the identity mapping $h(x) = x$. A down arrow indicates that a higher score means more similar (and vice versa). The scores are averaged over the test data.

Method	Data set	Estimated value of the bound	GAN_A	GAN_B	$Cycle_A/Dist$	$Cycle_B$
Disco-GAN (Kim et al., 2017)	Shoes2Edges	**1.00** (<1E-16)	-0.15 (3E-03)	-0.28 (1E-08)	0.76(<1E-16)	0.79(<1E-16)
	Bags2Edges	**1.00** (<1E-16)	-0.26 (6E-11)	-0.57 (<1E-16)	0.85 (<1E-16)	0.84 (<1E-16)
	Cityscapes	**0.94** (<1E-16)	-0.66 (<1E-16)	-0.69 (<1E-16)	-0.26 (1E-07)	0.80 (<1E-16)
	Facades	**0.85** (<1E-16)	-0.46 (<1E-16)	0.66 (<1E-16)	0.92 (<1E-16)	0.66 (<1E-16)
	Maps	**1.00** (<1E-16)	-0.81 (<1E-16)	0.58 (<1E-16)	0.20 (9E-05)	-0.14 (5E-03)
Distance-GAN (Benaim and Wolf, 2017)	Shoes2Edges	**0.98** (<1E-16)	-	-0.25 (2E-16)	-0.14 (1E-05)	-
	Bags2Edges	**0.93** (<1E-16)	-	-0.08 (2E-02)	0.34 (<1E-16)	-
	Cityscapes	**0.59** (<1E-16)	-	0.22 (1E-11)	-0.41 (<1E-16)	-
	Facades	**0.48** (<1E-16)	-	0.03 (5E-01)	-0.01 (9E-01)	-
	Maps	**1.00** (<1E-16)	-	-0.73 (<1E-16)	0.39 (4E-16)	-
Cycle-GAN (Zhu et al., 2017)	Shoes2Edges	**0.99** (<1E-16)	0.44 (5E-10)	0.038 (3E-12)	-0.44 (5E-13)	-0.40 (3E-11)
	Bags2Edges	**0.99** (<1E-16)	-0.23 (<1E-16)	0.21 (<2E-14)	-0.20 (5E-15)	-0.34 (4E-10)
	Cityscapes	**0.91** (<1E-16)	0.30 (6E-11)	0.024 (4E-11)	0.37 (3E-05)	0.42 (2E-14)
	Facades	**0.73** (<1E-16)	-0.02 (<1E-16)	-0.1 (<1E-16)	-0.14 (4E-10)	0.2 (3E-11)
	Maps	**0.85** (<1E-16)	0.01 (5E-16)	0.26 (3E-16)	-0.39 (1E-15)	-0.32 (4E-10)

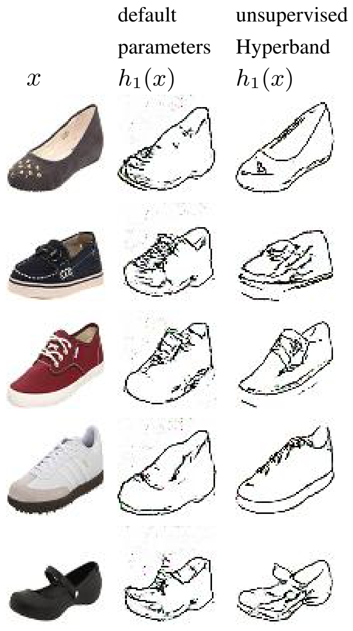
Table 3: Pearson correlations and the corresponding p-values (in parentheses) of the ground-truth error with: (i) the estimated value of the bound, (ii) the GAN losses, and (iii) the circularity losses/distance similarity loss.



(a)

Data set	Number Layers	Batch Size	Learning Rate
DiscoGAN (Kim et al., 2017)			
Shoes2Edges	3	24	0.0008
Bags2Edges	2	59	0.0010
Cityscapes	3	27	0.0009
Facades	3	20	0.0008
Maps	3	20	0.0005
DistanceGAN (Benaim and Wolf, 2017)			
Shoes2Edges	3	15	0.0007
Bags2Edges	3	33	0.0007
Cityscapes	4	21	0.0006
Facades	3	8	0.0006
Maps	3	20	0.0005
Data set	#Layers	#Res	L.Rate
UNIT (Liu et al., 2017)			
Maps	3	1	0.0003

(b)



(c)

Figure 6: **Applying unsupervised Hyperband for selecting a well-performing configuration.** For DiscoGAN **(left)** and DistanceGAN **(right)**, we optimize over the number of encoder and decoder layers, batch size and learning rate, while for UNIT, we optimize over the number of encoder and decoder layers, number of residual layers and learning rate. **(a)** Each graph shows the error of the best configuration selected by Hyperband, as a function the number of Hyperband iterations, when optimizing the estimated value of Equation (12) (blue). The corresponding ground-truth errors are shown in orange. Dotted lines represent the best configuration errors, when varying only the number of layers without Hyperband.**(b)** The corresponding hyperparameters of the best configuration as selected by Hyperband. **(c)** Images produced for DiscoGAN's shoes2edges: 1st column is the input, the 2nd is the result of DiscoGAN's default configuration, 3rd is the result of the configuration selected by our method.

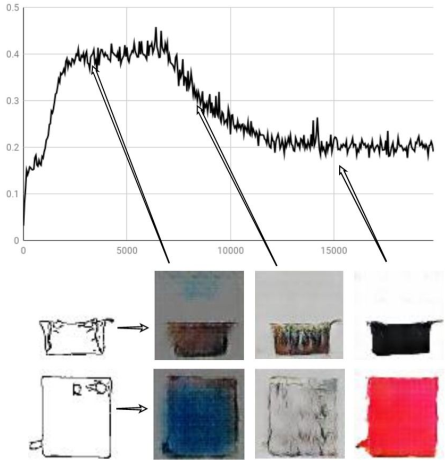


Figure 7: Results for Algorithm 3 for non-unique translation of Edges to Handbags. The black line stands for the estimated value of the bound. The images correspond to different values of the estimated bound.

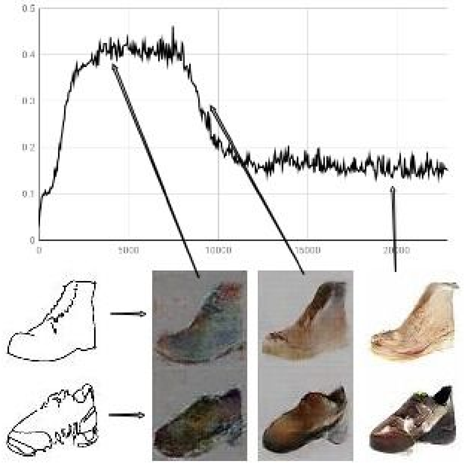
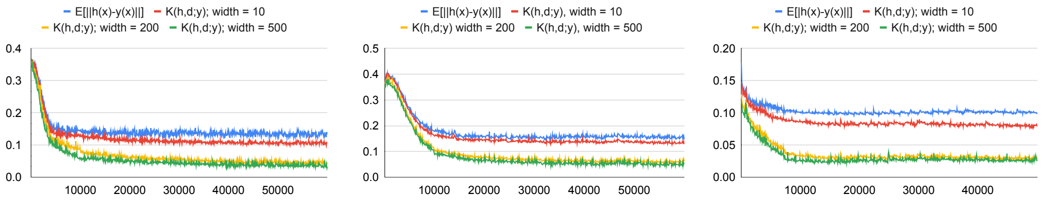


Figure 8: Results for Algorithm 3 for non-unique translation of Edges to Shoes. The black line stands for the estimated value of the bound. The images correspond to different values of the estimated bound.



(DistanceGAN, Shoes2Edges) (DiscoGAN, Bags2Edges) (CycleGAN, Cityscapes)

Figure 9: **Comparing $\mathcal{K}$ with $\mathcal{R}$.** The x-axis is the training iteration. The blue curve stands for the value of $\mathbb{E}_{x\sim D_A}[\|h(x) - y(y)\|_2]$, the red, yellow and green curves stand for the values of $\mathcal{K}(h, d; y) = \mathbb{E}_{x\sim D_A}\left[\|\nabla_{y(x)} d(y(x)) - (h(x) - y(x))\|_2\right]$ for d of widths $10, 200, 500$ (resp.). The expectations $\mathbb{E}_{x\sim D_A}$ are estimated using the test data.

Appendix A. Proofs of the Main Results

In this section, we prove Theorem 1.

A.1 Useful Lemmas

Lemma 2 *Let $\mathcal{C}$ be a symmetric class of functions $d : \mathcal{X} \to \mathbb{R}$ and D_1, D_2, D_3, D_4 be four distributions over $\mathcal{X}$, then,*

$$\Big|\rho_{\mathcal{C}}(D_1, D_2) - \rho_{\mathcal{C}}(D_3, D_4)\Big| \leq \rho_{\mathcal{C}}(D_1, D_3) + \rho_{\mathcal{C}}(D_2, D_4).$$

Proof We consider that:

$$\begin{aligned}
&\Bigg|\rho_{\mathcal{C}}(D_1, D_2) - \rho_{\mathcal{C}}(D_3, D_4)\Bigg| \\
=&\Bigg|\sup_{d\in\mathcal{C}}\Big\{\mathbb{E}_{x\sim D_1}[d(x)] - \mathbb{E}_{x\sim D_2}[d(x)]\Big\} - \sup_{d\in\mathcal{C}}\Big\{\mathbb{E}_{x\sim D_3}[d(x)] - \mathbb{E}_{x\sim D_4}[d(x)]\Big\}\Bigg| \\
\leq&\Bigg|\sup_{d\in\mathcal{C}}\Big\{\mathbb{E}_{x\sim D_1}[d(x)] - \mathbb{E}_{x\sim D_2}[d(x)] - \mathbb{E}_{x\sim D_3}[d(x)] + \mathbb{E}_{x\sim D_4}[d(x)]\Big\}\Bigg| \\
\leq&\Bigg|\sup_{d\in\mathcal{C}}\Big\{\mathbb{E}_{x\sim D_1}[d(x)] - \mathbb{E}_{x\sim D_3}[d(x)]\Big\} + \sup_{d\in\mathcal{C}}\Big\{\mathbb{E}_{x\sim D_2}[d(x)] - \mathbb{E}_{x\sim D_4}[d(x)]\Big\}\Bigg| \\
=&\sup_{d\in\mathcal{C}}\Big\{\mathbb{E}_{x\sim D_1}[d(x)] - \mathbb{E}_{x\sim D_3}[d(x)]\Big\} + \sup_{d\in\mathcal{C}}\Big\{\mathbb{E}_{x\sim D_2}[d(x)] - \mathbb{E}_{x\sim D_4}[d(x)]\Big\} \\
=&\rho_{\mathcal{C}}(D_1, D_3) + \rho_{\mathcal{C}}(D_2, D_4).
\end{aligned}$$

The last two equations follow from the definition of $\rho_{\mathcal{C}}$ and the assumption that $\mathcal{C}$ is symmetric. ■

The following lemma is a variation of the Occam's Razor theorem from (Benaim et al., 2018).

Lemma 3 *Let $y \in \mathcal{T}$ be a target function and $\mathcal{P}$ a class of functions $h : \mathcal{X}_A \to \mathbb{R}^M$. Then, for any function $h_1 \in \mathcal{P}$, we have:*

$$R_{D_A}[h_1, y] \leq 3 \sup_{h_2\in\mathcal{P}} R_{D_A}[h_1, h_2] + 3 \inf_{h\in\mathcal{P}} R_{D_A}[h, y]. \tag{13}$$

Proof First, we prove Equation (13). We consider that

$$\begin{aligned}
\ell(a, c) &= \|a - c\|_2^2 \\
&= \|a - b + b - c\|_2^2 \\
&\leq (\|a - b\|_2 + \|b - c\|_2)^2 \\
&= \|a - b\|_2^2 + \|b - c\|_2^2 + 2\|a - b\|_2 \cdot \|b - c\|_2 \\
&\leq \|a - b\|_2^2 + \|b - c\|_2^2 + 2\max(\|a - b\|_2^2, \|b - c\|_2^2) \\
&\leq 3(\|a - b\|_2^2 + \|b - c\|_2^2) \\
&= 3(\ell(a, b) + \ell(b, c)).
\end{aligned}$$

Therefore, for any function $h' \in \mathcal{P}$, we have:

$$\begin{aligned} R_{D_A}[h_1, y] &= \mathbb{E}_{x \sim D_A}[\|h_1(x) - y(x)\|_2^2] \\ &\leq \mathbb{E}_{x \sim D_A}\left[3\|h_1(x) - h'(x)\|_2^2 + 3\|h'(x) - y(x)\|_2^2\right] \\ &= 3\left[R_{D_A}[h_1, h'] + R_{D_A}[h', y]\right]. \end{aligned}$$

Since $h' \in \mathcal{P}$, we have: $R_{D_A}[h_1, h'] \leq \sup_{h_2 \in \mathcal{P}} R_{D_A}[h_1, h_2]$. Therefore, we have:

$$\forall h' \in \mathcal{P} : \; R_{D_A}[h_1, y] \leq 3\left[\sup_{h_2 \in \mathcal{P}} R_{D_A}[h_1, h_2] + R_{D_A}[h', y]\right].$$

In particular, since the inequality holds uniformly for all $h' \in \mathcal{P}$, we can take $\inf_{h' \in \mathcal{P}}$ in both sides of the inequality and obtain the desired inequality:

$$R_{D_A}[h_1, y] \leq 3\left[\sup_{h_2 \in \mathcal{P}} R_{D_A}[h_1, h_2] + \inf_{h' \in \mathcal{P}} R_{D_A}[h', y]\right].$$

■

The following lemma is a simple extension of (cf. Mohri et al., 2018, Theorem 3.3) for classes of functions that are bounded in $[0, L]$ instead of $[0, 1]$.

Lemma 4 *Let $\mathcal{X} \subset \mathbb{R}^N$ and $\mathcal{G}$ be a family of functions $g : \mathcal{X} \to [0, L]$. Let D be a distribution over $\mathcal{X}$. Then, for any $\delta > 0$, with probability at least $1 - \delta$ over the draw of an i.i.d. sample $\mathcal{S} \sim D^m$, we have:*

$$\forall g \in \mathcal{G} : \; \mathbb{E}_{x \sim D}[g(x)] \leq \frac{1}{m}\sum_{x \in \mathcal{S}} g(x) + 2\hat{\mathscr{R}}_{\mathcal{S}}(\mathcal{G}) + 3L\sqrt{\frac{\log(2/\delta)}{2m}}.$$

Proof Define $\mathcal{F} = \{g/L \mid g \in \mathcal{F}\}$. This class consists of functions from $\mathcal{X}$ to $[0, 1]$. By (cf. Mohri et al., 2018, Theorem 3.3), with probability at least $1 - \delta$ over the selection of $\mathcal{S} \sim D^m$, we have:

$$\forall f \in \mathcal{F} : \; \mathbb{E}_{x \sim D}[f(x)] \leq \frac{1}{m}\sum_{x \in \mathcal{S}} f(x) + 2\hat{\mathscr{R}}_{\mathcal{S}}(\mathcal{F}) + 3\sqrt{\frac{\log(2/\delta)}{2m}}.$$

We can rewrite the above inequality as follows:

$$\forall g \in \mathcal{G} : \; \mathbb{E}_{x \sim D}[g(x)/L] \leq \frac{1}{m}\sum_{x \in \mathcal{S}} g(x)/L + 2\hat{\mathscr{R}}_{\mathcal{S}}(\mathcal{F}) + 3\sqrt{\frac{\log(2/\delta)}{2m}}.$$

Hence, by multiplying both sides of the inequality by L, we have:

$$\forall g \in \mathcal{G} : \; \mathbb{E}_{x \sim D}[g(x)] \leq \frac{1}{m}\sum_{x \in \mathcal{S}} g(x) + 2L\hat{\mathscr{R}}_{\mathcal{S}}(\mathcal{F}) + 3L\sqrt{\frac{\log(2/\delta)}{2m}}.$$

Finally, we recall that the Rademacher complexity is multiplicative, and therefore, $L\hat{\mathscr{R}}_{\mathcal{S}}(\mathcal{F}) = \hat{\mathscr{R}}_{\mathcal{S}}(\mathcal{G})$, that completes the proof. ■

A.2 Proof of Theorem 1

The following lemma bounds the generalization risk between a hypothesis h and a target function y. The upper bound is a function of $\rho_{\mathcal{C}}(h \circ D_A, D_B) = \sup_{d \in \mathcal{C}} \{\mathbb{E}_{x \sim h \circ D_A}[d(x)] - \mathbb{E}_{x \sim D_B}[d(x)]\}$, which is the $\mathcal{C}$-IPM between the distributions $h \circ D_A$ and D_B. An additional term expresses the approximation of $h(x) - y(x)$ by the gradient of a function $d \in \mathcal{C}$. Both terms are multiplied by a term that depends on the smoothness of d.

Lemma 5 *Assume the settings of Section 4 and Section 5. Assume that $\mathcal{X}_A \subset \mathbb{R}^N$ and $\mathcal{X}_B \subset \mathbb{R}^M$ are convex and bounded sets. Assume that $\mathcal{C} \subset C^2$. Let $y \in \mathcal{T}$ be target function and $d \in \mathcal{C}$ such that $\beta(d) := \|\mathrm{H}_d\|_{\infty,\mathcal{X}_B} < 2$. Then, for any function $h \in \mathcal{H}$, we have:*

$$R_{D_A}[h, y] \leq \frac{2\rho_{\mathcal{C}}(h \circ D_A, D_B)}{2 - \beta(d)} + \frac{2 \sup_{u \in \mathcal{X}_A} \|h(u) - y(u)\|_2}{2 - \beta(d)} \cdot \mathcal{K}(h, d; y).$$

Proof First, since each function $f \in \mathcal{H} \cup \mathcal{T}$ is measurable, by a change of variables (cf. Varadhan, 2002, Theorem 1.9), we can represent the $\mathcal{C}$-IPM in the following manner:

$$\begin{aligned}
\rho_{\mathcal{C}}(h \circ D_A, D_B) &= \sup_{d \in \mathcal{C}} \Big\{ \mathbb{E}_{u \sim h \circ D_A}[d(u)] - \mathbb{E}_{v \sim D_B}[d(v)] \Big\} \\
&= \sup_{d \in \mathcal{C}} \Big\{ \mathbb{E}_{u \sim h \circ D_A}[d(u)] - \mathbb{E}_{v \sim y \circ D_A}[d(v)] \Big\} \\
&= \sup_{d \in \mathcal{C}} \Big\{ \mathbb{E}_{x \sim D_A}[d \circ h(x)] - \mathbb{E}_{x \sim D_A}[d \circ y(x)] \Big\} \\
&= \sup_{d \in \mathcal{C}} \Big\{ \mathbb{E}_{x \sim D_A}[d \circ h(x) - d \circ y(x)] \Big\}.
\end{aligned} \tag{14}$$

For fixed $d \in \mathcal{C}$ and $z \in \mathcal{X}_B$, we can write the following Taylor expansion (possible since $\mathcal{C} \subset C^2$):

$$d(z + \delta) - d(z) = (\nabla d(z))^\top \cdot \delta + \frac{1}{2}\delta^\top \cdot \mathrm{H}_d(u^*) \cdot \delta,$$

where u^* is strictly between z and $z + \delta$ (on the line connecting z and $z + \delta$). In particular, for each $d \in \mathcal{C}$ and $x \in \mathcal{X}_A$, if $z = y(x)$ and $\delta = h(x) - y(x)$, we have:

$$\begin{aligned}
d(h(x)) - d(y(x)) =& (\nabla_{y(x)} d(y(x)))^\top \cdot (h(x) - y(x)) \\
&+ \frac{1}{2}(h(x) - y(x))^\top \cdot \mathrm{H}_d(u^*_{d,x}) \cdot (h(x) - y(x)),
\end{aligned} \tag{15}$$

where $u^*_{d,x}$ is strictly between $y(x)$ and $h(x)$ (on the line connecting $y(x)$ and $h(x)$). Therefore, by combining Equations (14) and (15), we obtain that for every $d \in \mathcal{C}$, we have:

$$\begin{aligned}
\rho_{\mathcal{C}}(h \circ D_A, D_B) \geq& \mathbb{E}_{x \sim D_A}[d(h(x)) - d(y(x))] \\
=& \mathbb{E}_{x \sim D_A}\Big[(\nabla_{y(x)} d(y(x)))^\top \cdot (h(x) - y(x)) \Big] \\
&+ \frac{1}{2}\mathbb{E}_{x \sim D_A}\Big[(h(x) - y(x))^\top \cdot \mathrm{H}_d(u^*_{d,x}) \cdot (h(x) - y(x)) \Big] \\
=& \mathbb{E}_{x \sim D_A}\big[\|h(x) - y(x)\|_2^2 \big] \\
&+ \mathbb{E}_{x \sim D_A}\Big[\big(\nabla_{y(x)} d(y(x)) - (h(x) - y(x))\big)^\top \cdot (h(x) - y(x)) \Big] \\
&+ \frac{1}{2}\mathbb{E}_{x \sim D_A}\Big[(h(x) - y(x))^\top \cdot \mathrm{H}_d(u^*_{d,x}) \cdot (h(x) - y(x)) \Big].
\end{aligned}$$

In particular, by $|\mathbb{E}[X]| \leq \mathbb{E}[|X|]$, we have:

$$\begin{aligned}
&\rho_C(h \circ D_A, D_B) \\
\geq &\mathbb{E}_{x\sim D_A}\left[\|h(x)-y(x)\|_2^2\right] - \left|\mathbb{E}_{x\sim D_A}\left[\left(\nabla_{y(x)}d(y(x)) - (h(x)-y(x))\right)^\top \cdot (h(x)-y(x))\right]\right| \\
&- \frac{1}{2}\left|\mathbb{E}_{x\sim D_A}\left[(h(x)-y(x))^\top \cdot \mathrm{H}_d(u^*_{d,x}) \cdot (h(x)-y(x))\right]\right| \\
\geq &\mathbb{E}_{x\sim D_A}\left[\|h(x)-y(x)\|_2^2\right] - \mathbb{E}_{x\sim D_A}\left[\left|\left(\nabla_{y(x)}d(y(x)) - (h(x)-y(x))\right)^\top \cdot (h(x)-y(x))\right|\right] \\
&- \frac{1}{2}\mathbb{E}_{x\sim D_A}\left[\left|(h(x)-y(x))^\top \cdot \mathrm{H}_d(u^*_{d,x}) \cdot (h(x)-y(x))\right|\right].
\end{aligned} \tag{16}$$

By applying the Cauchy-Schwartz inequality,

$$\begin{aligned}
&\left|\left(\nabla_{y(x)}d(y(x)) - (h(x)-y(x))\right)^\top \cdot (h(x)-y(x))\right| \\
&\leq \|\nabla_{y(x)}d(y(x)) - (h(x)-y(x))\|_2 \cdot \|h(x)-y(x)\|_2 \\
&\leq \|\nabla_{y(x)}d(y(x)) - (h(x)-y(x))\|_2 \cdot \sup_{u\in\mathcal{X}_A}\|h(u)-y(u)\|_2.
\end{aligned} \tag{17}$$

Again, by applying the Cauchy-Schwartz inequality,

$$\begin{aligned}
\clubsuit :=&\left|(h(x)-y(x))^\top \cdot \mathrm{H}_d(u^*_{d,x}) \cdot (h(x)-y(x))\right| \\
\leq&\|(h(x)-y(x))^\top \cdot \mathrm{H}_d(u^*_{d,x})\|_2 \cdot \|h(x)-y(x)\|_2 \\
\leq&\|\mathrm{H}_d(u^*_{d,x})\|_2 \cdot \|h(x)-y(x)\|_2^2.
\end{aligned}$$

Since $\mathcal{X}_B$ is convex, $y(x), h(x) \in \mathcal{X}_B$ and $u^*_{d,x}$ is on the line connecting $y(x)$ and $h(x)$, we have: $u^*_{d,x} \in \mathcal{X}_B$. In particular,

$$\begin{aligned}
\clubsuit &\leq \sup_{z\in\mathcal{X}_B}\|\mathrm{H}_d(z)\|_2 \cdot \|h(x)-y(x)\|_2^2 \\
&= \beta(d) \cdot \|h(x)-y(x)\|_2^2.
\end{aligned} \tag{18}$$

Therefore, by combining Equations (16), (17) and (18), we have:

$$\begin{aligned}
\rho_C(h \circ D_A, D_B) \geq &\mathbb{E}_{x\sim D_A}\left[\|h(x)-y(x)\|_2^2\right] - \frac{1}{2}\mathbb{E}_{x\sim D_A}\left[\beta(d)\cdot\|h(x)-y(x)\|_2^2\right] \\
&- \sup_{u\in\mathcal{X}_A}\|h(u)-y(u)\|_2 \cdot \mathbb{E}_{x\sim D_A}\left[\|\nabla_{y(x)}d(y(x)) - (h(x)-y(x))\|_2\right] \\
=&\left(1-\frac{\beta(d)}{2}\right)R_{D_A}[h,y] \\
&- \sup_{u\in\mathcal{X}_A}\|h(u)-y(u)\|_2 \cdot \mathbb{E}_{x\sim D_A}\left[\|\nabla_{y(x)}d(y(x)) - (h(x)-y(x))\|_2\right] \\
=&\left(1-\frac{\beta(d)}{2}\right)R_{D_A}[h,y] - \sup_{u\in\mathcal{X}_A}\|h(u)-y(u)\|_2 \cdot \mathcal{K}(h,d;y).
\end{aligned} \tag{19}$$

By combining Equation (19) and $\beta(d) < 2$, we obtain the desired bound. ■

The following result is obtained by combining Lemma 5 with Lemma 3.

Lemma 6 *Assume the setting of Section 4. Assume that $\mathcal{X}_A \subset \mathbb{R}^N$ and $\mathcal{X}_B \subset \mathbb{R}^M$ are convex and bounded sets. Assume that $\mathcal{C} \subset C^2$. Let $\mathcal{T}$ be a class target functions and $\mathcal{P} \subset \mathcal{H}$ be a class of candidate functions. Then, for any $y \in \mathcal{T}$, $h \in \mathcal{P}$, $d \in \mathcal{C}$, such that, $\beta(d) < 2$ and function $h_1 \in \mathcal{H}$, we have:*

$$\begin{aligned} R_{D_A}[h_1, y] \leq & 3 \sup_{h_2 \in \mathcal{P}} R_{D_A}[h_1, h_2] + \frac{6\rho_{\mathcal{C}}(h \circ D_A, D_B)}{2 - \beta(d)} \\ & + \frac{6 \sup\limits_{u \in \mathcal{X}_A} \|h(u) - y(u)\|_2}{2 - \beta(d)} \cdot \mathcal{K}(h, d; y). \end{aligned}$$

Proof Let $y \in \mathcal{T}$, $h \in \mathcal{P}$ and $d \in \mathcal{C}$, such that, $\beta(d) < 2$. By Lemma 5:

$$R_{D_A}[h, y] \leq \frac{2\rho_{\mathcal{C}}(h \circ D_A, D_B)}{2 - \beta(d)} + \frac{2 \sup\limits_{u \in \mathcal{X}_A} \|h(u) - y(u)\|_2}{2 - \beta(d)} \cdot \mathcal{K}(h, d; y). \tag{20}$$

In particular, since $h \in \mathcal{P}$, we have: $\inf\limits_{h' \in \mathcal{P}} R_{D_A}[h', y] \leq R_{D_A}[h, y]$. By combining Equation (13) with Equation (20), we obtain the desired inequality. ■

Lemma 7 *Assume that $\mathcal{X}_A \subset \mathbb{R}^N$ and $\mathcal{X}_B \subset \mathbb{R}^M$ are convex and bounded sets. Assume that $\mathcal{C} \subset C^2$. Let $\mathcal{T}$ be a class target functions and $\mathcal{P}$ a class of candidate functions. Let $\alpha \in [0, 1)$. Then, for any $h_1 \in \mathcal{H}$, we have:*

$$\inf_{y \in \mathcal{T}} R_{D_A}[h_1, y] \lesssim \sup_{h_2 \in \mathcal{P}} R_{D_A}[h_1, h_2] + \frac{1}{1-\alpha} \inf_{h,d} \left\{ \rho_{\mathcal{C}}(h \circ D_A, D_B) + \inf_{y \in \mathcal{T}} \mathcal{K}(h, d; y) \right\},$$

where the infimum is taken over $h \in \mathcal{P}$ and $d \in \mathcal{C}$, such that, $\beta(d) \leq 1 + \alpha$.

Proof Let $h \in \mathcal{P}$, such that, $h : \mathcal{X}_A \to \mathcal{X}_B$, $d \in \mathcal{C}$, such that $\beta(d) \leq 1$ and $y \in \mathcal{T}$. Then, by Lemma 6, for every $h_1 \in \mathcal{P}$, we have:

$$\begin{aligned} R_{D_A}[h_1, y] \leq & 3 \sup_{h_2 \in \mathcal{P}} R_{D_A}[h_1, h_2] + 6\rho_{\mathcal{C}}(h \circ D_A, D_B) \\ & + 6 \sup_{u \in \mathcal{X}_A} \|h(u) - y(u)\|_2 \cdot \mathcal{K}(h, d; y). \end{aligned}$$

In particular, since $\mathcal{X}_B$ is bounded, there is a constant $L > 0$ such that $\sup\limits_{a,b \in \mathcal{X}_B} \|a - b\|_2 \leq L$. Hence, for every $h, y : \mathcal{X}_A \to \mathcal{X}_B$, we have: $\sup\limits_{u \in \mathcal{X}_A} \|h(u) - y(u)\|_2 \leq L$. Therefore,

$$R_{D_A}[h_1, y] \lesssim \sup_{h_2 \in \mathcal{P}} R_{D_A}[h_1, h_2] + \frac{1}{1-\alpha} \inf_{h,d} \left\{ \rho_{\mathcal{C}}(h \circ D_A, D_B) + \mathcal{K}(h, d; y) \right\}. \tag{21}$$

Finally, by taking $\inf\limits_{y \in \mathcal{T}}$ in both sides of Equation (21), we obtain the desired inequality. ■

Lemma 8 (Cross-Domain Mapping with IPMs) *Assume that $\mathcal{X}_A \subset \mathbb{R}^N$ and $\mathcal{X}_B \subset \mathbb{R}^M$ are convex and bounded sets. Assume that $\mathcal{C} \subset C^2$ and $\sup_{d\in\mathcal{C}} \|d\|_{\infty,\mathcal{X}_B} < \infty$. Let $\alpha \in [0,1)$. Then, for any $\delta \in (0,1)$, with probability at least $1-\delta$ over the selection of $\mathcal{S}_A \sim D_A^{m_1}$ and $\mathcal{S}_B \sim D_B^{m_2}$, for every $\omega \in \Omega$ and $h_1 \in \mathcal{P}_\omega(\mathcal{S}_A,\mathcal{S}_B)$, we have:*

$$\begin{aligned}\inf_{y\in\mathcal{T}} R_{D_A}[h_1,y] \lesssim &\sup_{h_2\in\mathcal{P}_\omega(\mathcal{S}_A,\mathcal{S}_B)} R_{\mathcal{S}_A}[h_1,h_2] \\ &+ \frac{1}{1-\alpha}\inf_{h\in\mathcal{P}_\omega(\mathcal{S}_A,\mathcal{S}_B)}\left\{\rho_{\mathcal{C}}(h\circ\mathcal{S}_A,\mathcal{S}_B) + \inf_{y\in\mathcal{T}}\inf_{\substack{d\in\mathcal{C}\\ \beta(d)\leq 1+\alpha}} \mathcal{K}(h,d;y)\right\} \\ &+ \hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{H}) + \frac{1}{1-\alpha}\left(\hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{C}\circ\mathcal{H}) + \hat{\mathscr{R}}_{\mathcal{S}_B}(\mathcal{C}) + \sqrt{\frac{\log(1/\delta)}{\min(m_1,m_2)}}\right).\end{aligned}$$

Proof By Lemma 7, for any class $\mathcal{P}$, we have:

$$\begin{aligned}\inf_{y\in\mathcal{T}} R_{D_A}[h_1,y] \lesssim &\sup_{h_2\in\mathcal{P}} R_{D_A}[h_1,h_2] \\ &+ \frac{1}{1-\alpha}\inf_{h,d}\left\{\rho_{\mathcal{C}}(h\circ D_A, D_B) + \inf_{y\in\mathcal{T}}\mathcal{K}(h,d;y)\right\},\end{aligned}$$

where the infimum is taken over $h \in \mathcal{P}$ and $d \in \mathcal{C}$, such that, $\beta(d) \leq 1+\alpha$. In particular, for any two data sets $\mathcal{S}_A$ and $\mathcal{S}_B$ and $\omega \in \Omega$, we have:

$$\begin{aligned}\inf_{y\in\mathcal{T}} R_{D_A}[h_1,y] \lesssim &\sup_{h_2\in\mathcal{P}_\omega(\mathcal{S}_A,\mathcal{S}_B)} R_{D_A}[h_1,h_2] \\ &+ \frac{1}{1-\alpha}\inf_{h,d}\left\{\rho_{\mathcal{C}}(h\circ D_A, D_B) + \inf_{y\in\mathcal{T}}\mathcal{K}(h,d;y)\right\},\end{aligned} \tag{22}$$

where the infimum is taken over $h \in \mathcal{P}_\omega(\mathcal{S}_A,\mathcal{S}_B)$ and $d \in \mathcal{C}$, such that, $\beta(d) \leq 1+\alpha$. Therefore, we are left to replace the terms $R_{D_A}[h_1,h_2]$ and $\rho_{\mathcal{C}}(h\circ D_A, D_B)$ with their empirical versions, $R_{\mathcal{S}_A}[h_1,h_2]$ and $\rho_{\mathcal{C}}(h\circ\mathcal{S}_A,\mathcal{S}_B)$.

Let $L := \max_{a,b\in\mathcal{X}_B}\|a-b\|_2$ and let $\ell_{\mathcal{H}} := \{\ell(h_1(x),h_2(x)) \mid h_1,h_2 \in \mathcal{H}\}$. We recall that any $h \in \mathcal{H}$ is a mapping from $\mathcal{X}_A$ to $\mathcal{X}_B$. Therefore, for any $h_1,h_2 \in \mathcal{H}$ and $x \in \mathcal{X}_A$, we have: $0 \leq \ell(h_1(x),h_2(x)) = \|h_1(x)-h_2(x)\|_2^2 \leq 3(\|h_1(x)\|_2^2 + \|h_2(x)\|_2^2) \leq 3L^2$. Hence, each function $q \in \ell_{\mathcal{H}}$ is bounded within $[0,3L^2]$. By Lemma 4, with probability at least $1-\delta/3$ over the selection of $\mathcal{S}_A \sim D^m$, for any $q \in \ell_{\mathcal{H}}$, we have:

$$\forall q \in \ell_{\mathcal{H}} : \mathbb{E}_{x\sim D_A}[q(x)] \leq \frac{1}{m_1}\sum_{x\in\mathcal{S}_A} q(x) + 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + 9L^2\sqrt{\frac{\log(6/\delta)}{2m_1}}.$$

We can rewrite the above inequality as follows:

$$\begin{aligned}\forall h_1,h_2 \in \mathcal{H} : R_{D_A}[h_1,h_2] =& \mathbb{E}_{x\sim D_A}[\ell(h_1(x),h_2(x))] \\ \leq& \frac{1}{m_1}\sum_{x\in\mathcal{S}_A}\ell(h_1(x),h_2(x)) + 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + 9L^2\sqrt{\frac{\log(6/\delta)}{2m_1}} \\ =& R_{\mathcal{S}_A}[h_1,h_2] + 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + 9L^2\sqrt{\frac{\log(6/\delta)}{2m_1}}.\end{aligned}$$

In particular, with probability at least $1 - \delta/3$, for any $h_1 \in \mathcal{H}$, $\mathcal{P} \subset \mathcal{H}$ and $h_2 \in \mathcal{P}$, we have:

$$R_{D_A}[h_1, h_2] \leq R_{\mathcal{S}_A}[h_1, h_2] + 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + 9L^2\sqrt{\frac{\log(6/\delta)}{2m_1}}.$$

Since the inequality holds uniformly for all $\mathcal{P} \subset \mathcal{H}$ and $h_2 \in \mathcal{P}$, we can take $\sup_{h_2 \in \mathcal{P}}$ in both sides of the inequality. Therefore, with probability at least $1 - \delta/3$, we have:

$$\forall h_1 \in \mathcal{H}, \mathcal{P} \subset \mathcal{H} : \sup_{h_2 \in \mathcal{P}} R_{D_A}[h_1, h_2] \leq \sup_{h_2 \in \mathcal{P}} R_{\mathcal{S}_A}[h_1, h_2] + 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + 9L^2\sqrt{\frac{\log(6/\delta)}{2m_1}}.$$

Hence, by selecting $\mathcal{P} := \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)$, we have:

$$\begin{aligned}\sup_{h_2 \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)} R_{D_A}[h_1, h_2] &\leq \sup_{h_2 \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)} R_{\mathcal{S}_\mathcal{A}}[h_1, h_2] + 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + 9L^2\sqrt{\frac{\log(1/\delta)}{m_1}} \\ &\lesssim \sup_{h_2 \in \mathcal{P}_\omega(\mathcal{S}_A, \mathcal{S}_B)} R_{\mathcal{S}_\mathcal{A}}[h_1, h_2] + \hat{\mathscr{R}}_{\mathcal{S}_A}(\ell_{\mathcal{H}}) + \sqrt{\frac{\log(1/\delta)}{m_1}},\end{aligned} \tag{23}$$

where the last inequality follows from the fact that L is a constant. Next, we would like to replace the $\mathcal{C}$-IPM with its empirical counterpart. By Lemma 2, we have:

$$\Big|\rho_{\mathcal{C}}(h \circ D_A, D_B) - \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B)\Big| \leq \rho_{\mathcal{C}}(h \circ D_A, h \circ \mathcal{S}_A) + \rho_{\mathcal{C}}(D_B, \mathcal{S}_B).$$

In particular, by the triangle inequality, we have:

$$\rho_{\mathcal{C}}(h \circ D_A, D_B) \leq \rho_{\mathcal{C}}(h \circ \mathcal{S}_A, \mathcal{S}_B) + \rho_{\mathcal{C}}(h \circ D_A, h \circ \mathcal{S}_A) + \rho_{\mathcal{C}}(D_B, \mathcal{S}_B). \tag{24}$$

Again, by Lemma 4, with probability at least $1 - \delta/3$ over the selection of $\mathcal{S}_B \sim D_B^{m_2}$, we have:

$$\begin{aligned}\forall d \in \mathcal{C} : \ & \mathbb{E}_{x \sim D_B}[d(x)] - \frac{1}{m_2}\sum_{x \in \mathcal{S}_B} d(x) \\ & \leq 2\hat{\mathscr{R}}_{\mathcal{S}_B}(\mathcal{C}) + 3\sup_{d \in \mathcal{C}} \|d\|_{\infty, x_B}\sqrt{\frac{\log(6/\delta)}{2m_2}}.\end{aligned}$$

In particular, by taking $\sup_{d \in \mathcal{C}}$ in both sides of the inequality, we have:

$$\begin{aligned}\rho_{\mathcal{C}}(D_B, \mathcal{S}_B) &= \sup_{d \in \mathcal{C}}\left\{\mathbb{E}_{x \sim D_B}[d(x)] - \frac{1}{m_2}\sum_{x \in \mathcal{S}_B} d(x)\right\} \\ &\leq 2\hat{\mathscr{R}}_{\mathcal{S}_B}(\mathcal{C}) + 3\sup_{d \in \mathcal{C}} \|d\|_{\infty, x_B}\sqrt{\frac{\log(6/\delta)}{2m_2}} \\ &\lesssim \hat{\mathscr{R}}_{\mathcal{S}_B}(\mathcal{C}) + \sqrt{\frac{\log(1/\delta)}{m_2}},\end{aligned} \tag{25}$$

where the last inequality follows from the fact that $\sup_{d\in\mathcal{C}} \|d\|_{\infty,\mathcal{X}_B}$ is a constant. Similarly, by Lemma 4, with probability at least $1-\delta/3$ over the selection of $\mathcal{S}_A \sim D_A^{m_1}$, for all $d \in \mathcal{C}$ and $h \in \mathcal{H}$, we have:

$$\begin{aligned} \forall d \in \mathcal{C}, \forall h \in \mathcal{H} :\ & \mathbb{E}_{x\sim D_A}[d(h(x))] - \frac{1}{m_1}\sum_{x\in\mathcal{S}_A} d(h(x)) \\ & \leq 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{C}\circ\mathcal{H}) + 3\sup_{d\in\mathcal{C}}\|d\|_{\infty,\mathcal{X}_B}\sqrt{\frac{\log(6/\delta)}{2m_1}}. \end{aligned}$$

By taking $\sup_{d\in\mathcal{C}}$ in both sides of the inequality, we have:

$$\begin{aligned} \forall h \in \mathcal{H} : \rho_{\mathcal{C}}(h\circ D_A, h\circ\mathcal{S}_A) &= \sup_{d\in\mathcal{C}}\left\{\mathbb{E}_{x\sim D_A}[d(h(x))] - \frac{1}{m_1}\sum_{x\in\mathcal{S}_A} d(h(x))\right\} \\ &\leq 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{C}\circ\mathcal{H}) + 3\sup_{d\in\mathcal{C}}\|d\|_{\infty,\mathcal{X}_B}\sqrt{\frac{\log(6/\delta)}{2m_1}} \qquad (26) \\ &\lesssim \hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{C}\circ\mathcal{H}) + \sqrt{\frac{\log(1/\delta)}{m_1}}, \end{aligned}$$

where the last inequality follows from the fact that $\sup_{d\in\mathcal{C}} \|d\|_{\infty,\mathcal{X}_B}$ is a constant. Therefore, by Equation (24) and union bound over Equations (25) and (26), with probability at least $1-2\delta/3$ over the selection of both $\mathcal{S}_A$ and $\mathcal{S}_B$, for every $h \in \mathcal{H}$, we have:

$$\begin{aligned} \rho_{\mathcal{C}}(h\circ D_A, D_B) \leq & \rho_{\mathcal{C}}(h\circ\mathcal{S}_A, \mathcal{S}_B) + 2\hat{\mathscr{R}}_{\mathcal{S}_B}(\mathcal{C}) + 2\hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{C}\circ\mathcal{H}) \\ & + 3\sup_{d\in\mathcal{C}}\|d\|_{\infty,\mathcal{X}_B}\left(\sqrt{\frac{\log(6/\delta)}{2m_1}} + \sqrt{\frac{\log(6/\delta)}{2m_2}}\right). \end{aligned} \tag{27}$$

Finally, by Equation (22) and union bound over Equations (23) and (27), with probability at least $1-\delta$ over the selection of both $\mathcal{S}_A$ and $\mathcal{S}_B$, for every $h_1, h_2 \in \mathcal{H}$ and $d \in \mathcal{C}$, such that, $\beta(d) \leq 1+\alpha$, we have:

$$\begin{aligned} \inf_{y\in\mathcal{T}} R_{D_A}[h_1, y] \lesssim & \sup_{h_2\in\mathcal{P}_\omega(\mathcal{S}_A,\mathcal{S}_B)} R_{D_A}[h_1,h_2] + \frac{1}{1-\alpha}\inf_{h,d}\left\{\rho_{\mathcal{C}}(h\circ D_A, D_B) + \inf_{y\in\mathcal{T}}\mathcal{K}(h,d;y)\right\} \\ \lesssim & \sup_{h_2\in\mathcal{P}_\omega(\mathcal{S}_A,\mathcal{S}_B)} R_{\mathcal{S}_A}[h_1,h_2] + \frac{1}{1-\alpha}\inf_{h,d}\left\{\rho_{\mathcal{C}}(h\circ\mathcal{S}_A, \mathcal{S}_B) + \inf_{y\in\mathcal{T}}\mathcal{K}(h,d;y)\right\} \\ & + \hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{H}) + \frac{1}{1-\alpha}\left(\hat{\mathscr{R}}_{\mathcal{S}_A}(\mathcal{C}\circ\mathcal{H}) + \hat{\mathscr{R}}_{\mathcal{S}_B}(\mathcal{C}) + \sqrt{\frac{\log(1/\delta)}{\min(m_1,m_2)}}\right). \end{aligned}$$

■

Theorem 1 follows immediately from the above lemma, by taking $\alpha = 0$ and $\mathcal{T} = \{y\}$.

Appendix B. Additional Experiments

In this section, we provide additional figures, tables and plots for demonstrating the results in Section 8.

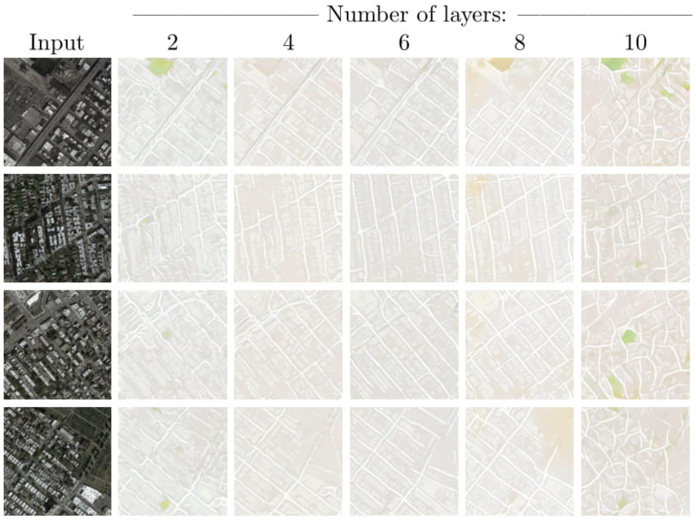


Figure 10: **Varying the number of layers of the generator.** Results of CycleGAN on Aerial View Images to Maps transfer. The best results are obtained for 4 or 6 layers. For more than 6 layers, the target alignment is lost.

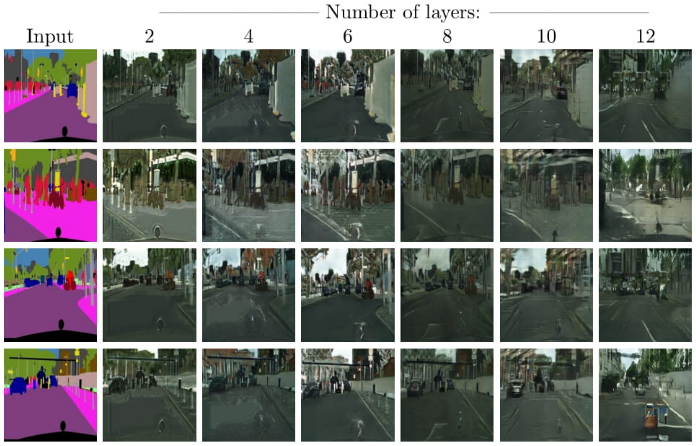


Figure 11: **Varying the number of layers of the generator.** Results of CycleGAN on Segmentation to Image transfer.

		$k=4$	$k=6$	$k=8$	$k=10$	$k=12$	$k=14$
Male to Female	Discrepancy (w circ)	0.521	0.203	0.091	0.094	0.080	0.084
	VGG similarity (w circ)	0.301	0.269	0.103	0.106	0.096	0.110
	Discrepancy (w.o circ)	0.501	0.213	0.102	0.091	0.079	0.82
	VGG similarity (w.o circ)	0.332	0.292	0.110	0.115	0.132	0.117
Female to Male	Discrepancy (w circ)	0.872	0.122	0.155	0.075	0.074	0.091
	VGG similarity (w circ)	0.313	0.287	0.118	0.109	0.095	0.104
	Discrepancy (w.o circ)	0.807	0.132	0.163	0.095	0.072	0.102
	VGG similarity (w.o circ)	0.298	0.283	0.117	0.115	0.090	0.094
Blond to Black Hair	Discrepancy (w circ)	0.447	0.204	0.092	0.082	0.084	0.081
	VGG similarity (w circ)	0.395	0.293	0.260	0.136	0.101	0.097
	Discrepancy (w.o circ)	0.431	0.212	0.087	0.092	0.098	0.078
	VGG similarity (w.o circ)	0.415	0.313	0.254	0.113	0.121	0.109
Black to Blond Hair	Discrepancy (w circ)	0.663	0.264	0.071	0.068	0.074	0.082
	VGG similarity (w circ)	0.347	0.285	0.245	0.113	0.093	0.097
	Discrepancy (w.o circ)	0.693	0.271	0.062	0.081	0.097	0.059
	VGG similarity (w.o circ)	0.361	0.273	0.258	0.121	0.071	0.078
Eyeglasses to Non-Eyeglasses	Discrepancy (w circ)	0.311	0.144	0.065	0.062	0.058	0.051
	VGG similarity (w circ)	0.493	0.402	0.377	0.173	0.153	0.148
	Discrepancy (w.o circ)	0.303	0.122	0.061	0.052	0.054	0.067
	VGG similarity (w.o circ)	0.531	0.433	0.353	0.151	0.122	0.141
Non Eyeglasses to Eyeglasses	Discrepancy (w circ)	0.542	0.528	0.226	0.243	0.097	0.085
	VGG similarity (w circ)	0.481	0.382	0.377	0.131	0.138	0.137
	Discrepancy (w.o circ)	0.512	0.502	0.193	0.186	0.084	0.065
	VGG similarity (w.o circ)	0.499	0.363	0.341	0.195	0.171	0.146

Table 4: **Comparing the performance of DiscoGAN with and without the circularity losses.** We compare the averaged VGG input-output descriptor similarity $\mathbb{E}_{x\sim D_A}[cs(f(x), f(h(x)))]$ and mapping discrepancy $\rho_C(h \circ D_A, D_B)$ of a generator h, when varying its number of layers k. "w/w.o circ" are short-hands that specify whether the generator was trained with or without circularity losses. The expectations $\mathbb{E}_{x\sim D_A}$ are estimated using the test data.

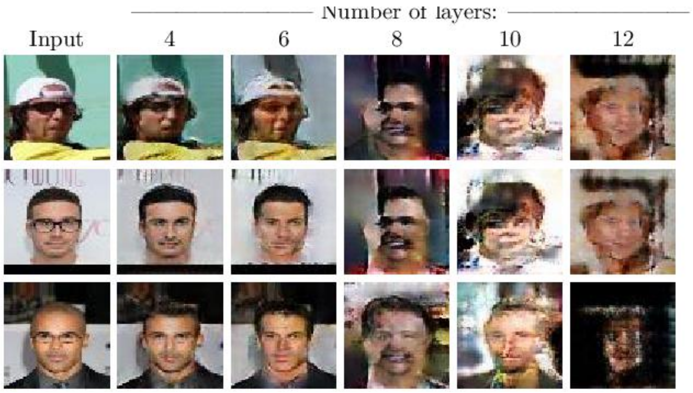


Figure 12: **Varying the number of layers of the generator.** Results of WGAN on eyeglasses removal.

		Number of layers				
	Norm	4	6	8	10	12
$A \rightarrow B$	L1 norm	6382	23530	36920	44670	71930
	Average L1 norm per layer	1064	2353	2637	2482	3270
	L2 norm	18.25	29.24	28.44	31.72	36.57
	Average L2 norm per layer	7.084	8.353	7.154	6.708	7.009
$B \rightarrow A$	L1 norm	6311	21240	31090	37380	64500
	Average L1 norm per layer	1052	2124	2221	2077	2932
	L2 norm	18.36	26.79	25.85	28.36	34.99
	Average L2 norm per layer	7.161	7.757	6.552	6.058	6.771

(a)

		Number of layers				
	Norm	4	6	8	10	12
$A \rightarrow B$	L1 norm	317200	228700	356500	247200	164200
	Average L1 norm per layer	9329	6726	10485	7271	4829
	L2 norm	528.1	401.7	559.6	410.1	346.8
	Average L2 norm per layer	3.031	2.284	3.242	2.257	1.890
$B \rightarrow A$	L1 norm	316900	194500	353900	171500	228900
	Average L1 norm per layer	9323	5719	10410	5045	6733
	L2 norm	523.2	375.9	555.7	346.5	373.3
	Average L2 norm per layer	3.003	2.029	3.210	1.921	2.289

(b)

Table 5: **(a)** Norms of the various mappings h for mapping Males to Females using the DiscoGAN architecture. **(b)** Norms of 18-layer networks that approximate the mappings obtained with a varying number of layers.

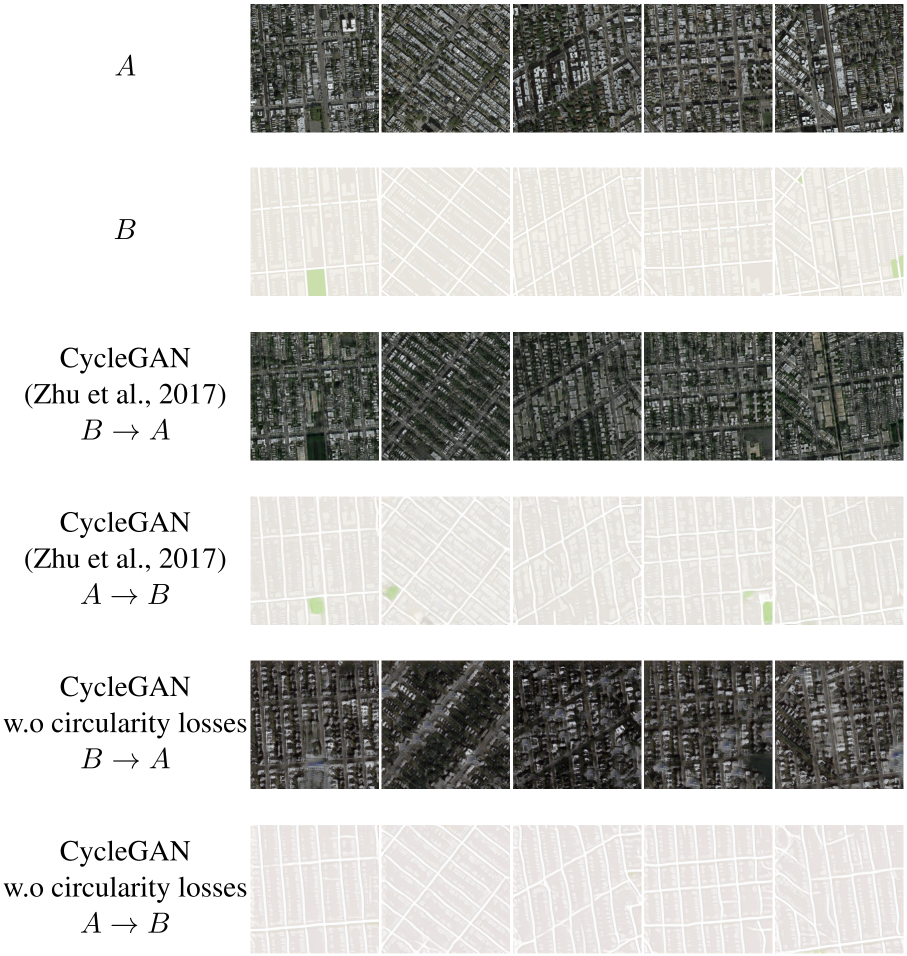


Figure 13: **Results of CycleGAN with and without circularity losses on the Maps data set.** The first and second row stand for the ground-truth pairs from the data set. The third (/fourth) row presents the results of CycleGAN for mapping the samples in the second (/first) row to samples into A (/B). The fifth and sixth rows are the same as the third and fourth rows, but for CycleGAN without the circularity losses.

References

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In *International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research. PMLR, 2017.

Sanjeev Arora, Rong Ge, Yingyu Liang, Tengyu Ma, and Yi Zhang. Generalization and equilibrium in generative adversarial nets (GANs). In *International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research. PMLR, 2017.

Peter L. Bartlett, Dylan J. Foster, and Matus Telgarsky. Spectrally-normalized margin bounds for neural networks. In *Advances in Neural Information Processing Systems*. Curran Associates Inc., 2017.

Sagie Benaim and Lior Wolf. One-sided unsupervised domain mapping. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017.

Sagie Benaim, Tomer Galanti, and Lior Wolf. Estimating the success of unsupervised image to image translation. In *European Conference on Computer Vision (ECCV)*, 2018.

David Berthelot, Tom Schumm, and Luke Metz. Began: Boundary equilibrium generative adversarial networks. *CoRR*, 2017.

Piotr Bojanowski, Armand Joulin, David Lopez-Pas, and Arthur Szlam. Optimizing the latent space of generative networks. In *International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research. PMLR, 2018.

Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

Tomer Galanti, Lior Wolf, and Sagie Benaim. The role of minimal complexity functions in unsupervised learning of semantic mappings. In *International Conference on Learning Representations (ICLR)*, 2018.

Noah Golowich, Alexander Rakhlin, and Ohad Shamir. Size-independent sample complexity of neural networks. In *Conference On Learning Theory (COLT)*, Proceedings of Machine Learning Research. PMLR, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2014.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2016.

Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jung Kwon Lee, and Jiwon Kim. Learning to discover cross-domain relations with generative adversarial networks. In *International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research. PMLR, 2017.

Vladimir Koltchinskii and Dmitriy Panchenko. Rademacher processes and bounding the risk of function learning. In Evarist Giné, David M. Mason, and Jon A. Wellner, editors, *High Dimensional Probability II*, pages 443–457, Boston, MA, 2000. Birkhäuser Boston.

Chun-Liang Li, Wei-Cheng Chang, Yu Cheng, Yiming Yang, and Barnabas Poczos. Mmd gan: Towards deeper understanding of moment matching network. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017.

L Li, K Jamieson, Giulia DeSalvo, A Rostamizadeh, and A Talwalkar. Hyperband: A novel bandit-based approach to hyperparameter optimization. *Journal of Machine Learning Research*, 18:1–52, 04 2018.

Yujia Li, Kevin Swersky, and Richard Zemel. Generative moment matching networks. In *International Conference on Machine Learning (ICML)*, Proceedings of Machine Learning Research. PMLR, 2015.

Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017.

Mario Lucic, Karol Kurach, Marcin Michalski, Olivier Bousquet, and Sylvain Gelly. Are gans created equal? a large-scale study. In *Advances in Neural Information Processing Systems*. Curran Associates Inc., 2018.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. MIT Press, Cambridge, MA, 2 edition, 2018. ISBN 978-0-262-03940-6.

Youssef Mroueh and Tom Sercu. Fisher gan. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2017.

Youssef Mroueh, Chun-Liang Li, Tom Sercu, Anant Raj, and Yu Cheng. Sobolev GAN. In *International Conference on Learning Representations (ICLR)*, 2018.

Alfred Müller. Integral probability metrics and their generating classes of functions advances in applied probability. In *Advances in Applied Probability*, pages 429—443, 1997.

Radim Šára Radim Tyleček. Spatial pattern templates for recognition of objects with regular structure. In *Proc. GCPR*, 2013.

Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115 (3):211–252, 2015.

Sathamangalam Ranga Iyengar Srinivasa Varadhan. Lecture notes on limit theorems, 2002. URL `https://math.nyu.edu/~varadhan/limittheorems.html`.

Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *International Conference on Computer Vision (ICCV)*. IEEE Computer Society, 2015.

Zili Yi, Hao Zhang, Ping Tan, and Minglun Gong. Dualgan: Unsupervised dual learning for image-to-image translation. In *International Conference on Computer Vision (ICCV)*. IEEE Computer Society, 2017.

A. Yu and K. Grauman. Fine-grained visual comparisons with local learning. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2014.

Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

Junbo Jake Zhao, Michaël Mathieu, and Yann LeCun. Energy-based generative adversarial network. In *International Conference on Learning Representations (ICLR)*, 2017.

Jun-Yan Zhu, Philipp Krähenbühl, Eli Shechtman, and Alexei A. Efros. Generative visual manipulation on the natural image manifold. In *European Conference on Computer Vision (ECCV)*, 2016.

Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *International Conference on Computer Vision (ICCV)*. IEEE Computer Society, 2017.