Langevin Cooling for Unsupervised Domain Translation

Vignesh Srinivasan, Klaus-Robert Müller*, *Member, IEEE*,
Wojciech Samek*, *Member, IEEE*, and Shinichi Nakajima*

***Abstract*—Domain translation is the task of finding correspondence between two domains. Several deep neural network (DNN) models, e.g., CycleGAN and cross-lingual language models, have shown remarkable successes on this task under the unsupervised setting—the mappings between the domains are learned from two independent sets of training data in both domains (without paired samples). However, those methods typically do not perform well on a significant proportion of test samples. In this paper, we hypothesize that many of such unsuccessful samples lie at the *fringe*—relatively low-density areas—of data distribution, where the DNN was not trained very well, and propose to perform Langevin dynamics to bring such fringe samples towards high density areas. We demonstrate qualitatively and quantitatively that our strategy, called *Langevin cooling* (L-Cool), enhances state-of-the-art methods in image translation and language translation tasks.**

***Index Terms*—Domain translation, Langevin dynamics, generative models, image to image translation, language translation.**

I. Introduction

Recently, deep neural networks (DNNs) have broadly contributed across various application domains in the sciences [1, 2, 3, 4, 5, 6, 7, 8] and the industry [9, 10, 11, 12, 13, 14, 15]. One of the notable successes is in unsupervised domain translation (DT), on which this paper focuses. DT is the task of translating data from a source domain to a target domain, which has applications in super-resolution [16], language translation [17, 18, 19], image translation [20, 21, 22, 23], text-image translation [24, 25], and data augmentation [26, 27, 28, 29] among others.

Corresponding authors: K.-R. Müller, W. Samek and S. Nakajima.
V. Srinivasan is with the Department of Artificial Intelligence, Fraunhofer Heinrich Hertz Institute, 10587 Berlin, Germany. (e-mail: vignesh.srinivasan@hhi.fraunhofer.de).
W. Samek is with the Department of Artificial Intelligence, Fraunhofer Heinrich Hertz Institute, 10587 Berlin, Germany and also with BIFOLD - Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany. (e-mail: wojciech.samek@hhi.fraunhofer.de).
K.-R. Müller is with the Machine Learning Group, Technische Universität Berlin, 10587 Berlin, Germany, and also with BIFOLD - Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany, the Department of Artificial Intelligence, Korea University, Seoul 136-713, South Korea and the Max Planck Institute for Informatics, 66123 Saarbrücken, Germany. (e-mail: klaus-robert.mueller@tu-berlin.de).
S. Nakajima is with the Machine Learning Group, Technische Universität Berlin, 10587 Berlin, Germany, with BiFOLD - Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany, and RIKEN AIP, 1-4-1 Nihonbashi, Chuo-ku, Tokyo 103-0027, Japan. (e-mail: nakajima@tu-berlin.de)

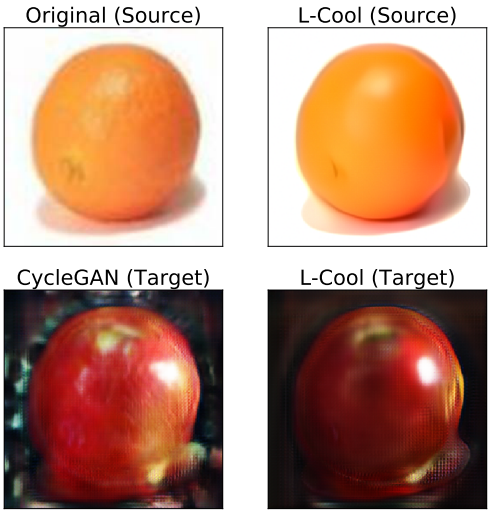


Fig. 1: An example of orange2apple task. The baseline CycleGAN transfers an orange image to an apple image (left column). Our proposed L-Cool makes a slight change in the original orange image, which significantly improves the quality of the transferred apple image (right column): the green artifacts surrounding the apple were removed almost completely, and the texture and the color on the apple were improved, although slight blurry along the edges of the apple was introduced.

In some DT applications, labeled samples, i.e., paired samples in the two domains, can be collected cheaply. For example, in the super-resolution, a paired low resolution image can be created by artificially blurring and down-sampling a high resolution image. However, in many other applications including image translation and language translation, collecting paired samples require significant human effort, and thus only a limited amount of paired data are available.

Unsupervised DT methods eliminate the necessity of paired data for supervision, and only require independent sets of training samples in both domains. In computer vision, CycleGAN, an extension of generative adversarial networks (GAN) [30], showed its capability of unsupervised DT with impressive results in image translation tasks [31, 32, 33]. It learns the mappings between the two domains by matching the source training distribution transferred to the target domain and the target training distribution, under the cycle consistency constraint. Similar ideas were applied

TABLE I: Examples of French-English translation by XLM [18] and L-Cool. L-Cool makes the translation closer to the ground-truth.

Original sentence	Le prix du pétrole continue à baisser et se rapproche de 96 $ le baril
Ground-truth translation	**Oil extends drop toward $ 96 a barrel**
XLM [18] (baseline)	Oil price continues to drop and moves past $ 96 a barrel
L-Cool (Ours)	**Oil price continues to drop and moves closer to $ 96 a barrel**
Original sentence	" Au milieu de XXe siècle , on appelait cela une urgence psychiatrique " , a indiqué Drescher
Ground-truth translation	**" Back in the middle of the 20th century , it was called a ' psychiatric emergency ' " said Drescher.**
XLM [18] (baseline)	" In the late 20th century , we called this a psychiatric emergency , " Drescher said
L-Cool (Ours)	**" In the middle of the 20th century , we called this a psychiatric emergency , " Drescher said .**

to natural language processing (NLP): dual learning [17, 34] and cross-lingual language models (XLM) [18], which are trained on unpaired monolingual data, achieved high performance in language translation.

Despite their remarkable successes, existing unsupervised DT methods are known to fail on a significant proportion of test samples [31, 35, 36, 37]. In this paper, we hypothesize that some of the unsuccessful samples are at the *fringe* of the data distribution, i.e., they lie slightly off the data manifold, and therefore the DNN was not trained very well for translating those samples. This hypothesis leads to our proposal to bring fringe samples towards the high density data manifold, where the DNN is well-trained, by *cooling down* the test distribution. Specifically, our proposed method, called L-Cool, performs the Metropolis-adjusted Langevin algorithm (MALA) to lower the temperature of test samples before applying the base DT method. The gradient of the log-probability, which MALA requires, is estimated by the denoising autoencoder (DAE) [38].

L-Cool is generic and can be used for enhancing any DT method. We demonstrate its effectiveness in image translation and language translation tasks, where L-Cool exhibits consistent performance gain. Figure 1 and Table I show a few intuitive exemplar results. The main contributions of this paper include:[1].

1) Proposal of a novel Langevin cooling (L-Cool) method that enhances DT performance by cooling down test samples towards the high density data manifold.
2) Qualitative evaluation of L-Cool in comparison with state-of-the-art methods as well as image processing techniques (as baseline projection methods) on image translation tasks including horse2zebra, zebra2horse, apple2orange and orange2apple which visualizes the effectiveness of L-Cool.
3) Quantitative evaluation on image translation tasks (horse2zebra and sat2map) based on classification accuracy by pretrained classifiers as well as paired data. Experiments with fringe detection support our hypotheses and show significant gains by L-Cool when applied to fringe samples.
4) Comparison between the gradient estimator by DAE and that by the cycle structure of CycleGAN (L-Cool-Cycle) on a synthetic toy dataset. Our investigation reveals drawbacks of L-Cool-Cycle.
5) Evaluation in language translation (English $\leftrightarrow$ French and English $\leftrightarrow$ German) on the NewsCrawl dataset[2], which revealed quantitative performance gain by L-Cool in terms of the BLEU score [40].
6) Identification of the feature space (L-Cool-Feature) as a more reliable place for applying Langevin dynamics than the input space (L-Cool-Input) for language translation models.
7) Analysis of hyperparameter dependence on image as well as language translation tasks.

[1]This paper is an extended version of our preliminary conference publication [39] with additional contributions and extended analyses. The conference publication [39] contains the first three contributions listed below, and the last four contributions have been newly added in this journal version. However, the first three contributions have also been refined with additional baselines and analyses. Specifically, all experimental results, which were obtained with L-Cool-Cycle in the conference version, have been replaced with the results obtained with L-Cool (with DAE), since drawbacks of L-Cool-cycle have been found.

A. Related Work

1) Unsupervised Image Translation: CycleGAN [31] and its concurrent works [32, 33] have eliminated the necessity of supervision for image translation [22, 41] by using the loss inspired by GAN [30] along with the cycle-consistency loss. The consistency requirement forces translation to retain the contents of source images so that they can be translated back. [42] proposed a variant that shares the latent space between the two domains, which works as additional regularization for alleviating the highly ill-posed nature of unsupervised domain translation.

[43] and [44] tackled the general issue of unimodality in sample generation by splitting the latent space into two—a content space and a style space. The content space is shared between the two domains but the style space is unique to each domain. The style space is modeled with a Gaussian prior, which helps in generating diverse images at test time. [36, 45] showed that attention maps can boost the performance by making the model focus on relevant regions in the image. Alternatives to cycle consistency include geometry-consistent generative adversarial networks (GcGAN) [46] and contrastive unpaired translation (CUT) [47]. GcGAN tries to maintain the distance between the inputs in the output space, while CUT employs patch based contrastive learning for improving DT performance. Despite a lot of new ideas proposed for improving the image translation performance, CycleGAN [31] is still considered to be the state-of-the-art in many transformation tasks.

[2]http://www.statmt.org/wmt14/index.html

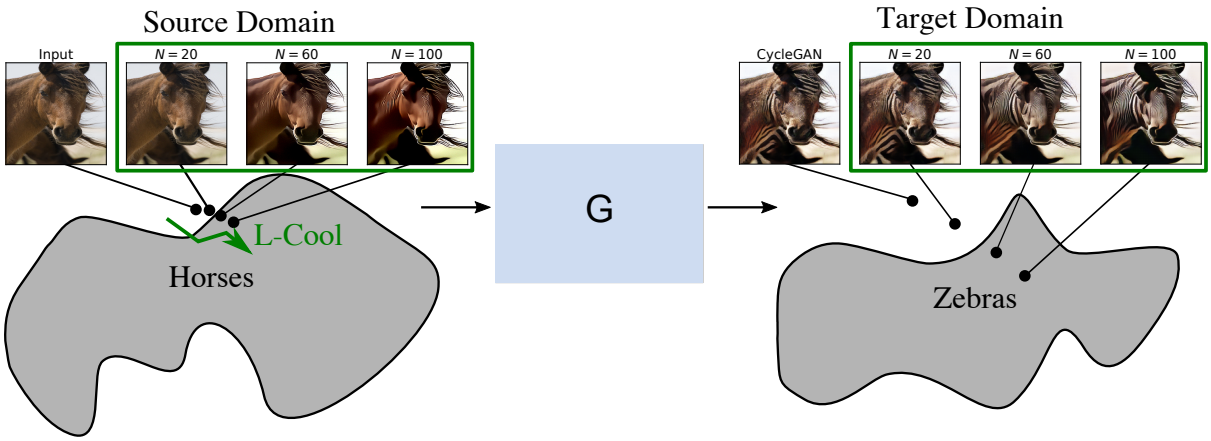


Fig. 2: L-Cool drives the test sample in the source (horse) domain slightly towards the center of data manifold, which gives a significant impact on the translated sample in the target (zebra) domain.

2) Unsupervised Language Translation: Language translation has been tackled with DNNs with encoder-decoder architectures, where text in the source language is fed to the encoder and the decoder generates its translation in the target language [48]. Unsupervised language translation methods have enabled learning from a large pool of monolingual data [17, 49], which can be cheaply collected through the internet without any human labeling effort.

Transformers [34] with attention mechanisms have shown their excellent performance in unsupervised language translation, as well as many other NLP tasks including language modeling, understanding, and sentence classification. It was shown that generative pretraining strategies like masked language modeling (MLM) (which masks a portion of the words in the input sentence and forces the model to predict the masked words) is effective in making transformers better at language understanding [50, 51, 52, 53]. Back translation has also enhanced performance by being a source of data augmentation while maintaining the cycle consistency constraint [19, 54, 55]. XLM [18] have shown state-of-the-art results in unsupervised language translation, outperforming generative pre-trained transformer (GPT) [50], bidirectional encoder representations from transformers (BERT) [52], and other previous methods [54, 56].

3) Temperature Control: Changing distributions by controlling the temperature has been used in Bayesian learning and sample generation. [57] and [58] reported that sampling weights from its cooled posterior distribution improves the predictive performance in Bayesian learning. Higher quality images were generated from a reduced-temperature model in [59, 60, 61]. [60] used a tempered softmax for super resolution. In contrast to previous works that cool down estimated distributions (Bayes posterior or predictive distributions), our approach cools down the input test distribution to make fringe samples more typical for unsupervised domain translation.

II. Theoretical Background

Here we introduce two basic tools, on which our proposed method relies.

A. Metropolis-Adjusted Langevin Algorithm

The Metropolis-adjusted Langevin algorithm (MALA) is an efficient Markov chain Monte Carlo (MCMC) sampling method that uses the gradient of the energy (negative log-probability $E(\boldsymbol{x}) = -\log p(\boldsymbol{x})$). Sampling is performed sequentially by

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \alpha \boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}_t) + \boldsymbol{\nu}, \qquad (1)$$

where α is the step size, and $\boldsymbol{\nu}$ is a random perturbation subject to $\mathcal{N}_L(\mathbf{0}, \delta^2 \boldsymbol{I}_L)$. Here $\mathcal{N}_L(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ denotes the L-dimensional Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$, and $\boldsymbol{I}_L$ denotes the $L \times L$ identity matrix. By appropriately controlling the step size α and the noise variance δ^2, the sequence is known to converge to the distribution $p(\boldsymbol{x})$.[3] [62] successfully generated high-resolution, realistic, and diverse artificial images by MALA.

B. Denoising Autoencoders (DAE)

A denoising autoencoder (DAE) [63, 64] is trained so that data samples contaminated with artificial noise are cleaned. Specifically, (an estimator) for the following reconstruction error is minimized:

$$L(\boldsymbol{r}) = \mathbb{E}_{p(\boldsymbol{x})p(\boldsymbol{\varepsilon})} \left[\|\boldsymbol{r}(\boldsymbol{x} + \boldsymbol{\varepsilon}) - \boldsymbol{x}\|^2 \right], \qquad (2)$$

where $\mathbb{E}_p[\cdot]$ denotes the expectation over the distribution p, $\mathbb{R}^L \ni \boldsymbol{x} \sim p(\boldsymbol{x})$ is a data sample, and $\boldsymbol{\varepsilon} \sim p(\boldsymbol{\varepsilon}) = \mathcal{N}_L(\mathbf{0}, \sigma^2 \boldsymbol{I})$ is an artificial Gaussian noise. [38] discussed the relation between DAEs and contractive autoencoders (CAE), and proved the following useful property of DAEs:

[3]For convergence, a rejection step after applying Eq.(1) is required. However, it was observed that a variant, called MALA-approx [62], without the rejection step gives reasonable sequence for moderate step sizes. We use MALA-approx in our proposed method.

Proposition 1: [38] Under the assumption that $\boldsymbol{r}(\boldsymbol{x}) = \boldsymbol{x} + o(1)$,[4] the minimizer of the DAE objective Eq.(2) satisfies

$$\boldsymbol{r}(\boldsymbol{x}) - \boldsymbol{x} = \sigma^2 \boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}) + o(\sigma^2), \tag{3}$$

as $\sigma^2 \to 0$.

Proposition 1 states that a DAE trained with a small σ^2 can be used to estimate the gradient of the log probability, i.e.,

$$\boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}) \approx \widehat{\boldsymbol{g}}(\boldsymbol{x}) \equiv \frac{\boldsymbol{r}(\boldsymbol{x}) - \boldsymbol{x}}{\sigma^2}. \tag{4}$$

III. Langevin Cooling (L-Cool)

A. Langevin Dynamics with Lower Temperature

As discussed in Section I, we hypothesize that domain translation (DT) methods can work poorly on test samples lying at the *fringe* of the data distribution. We therefore propose to drive such fringe samples towards the high density area, where the DNN is better trained. Specifically, we apply MALA Eq.(1) to each test sample with the step size α and the variance δ^2 of the random perturbation satisfying the following inequality:

$$2\alpha > \delta^2. \tag{5}$$

If $2\alpha = \delta^2$, MALA can be seen as a discrete approximation to the (continuous) Langevin dynamics,

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}) + \sqrt{2}\frac{d\boldsymbol{w}}{dt}, \tag{6}$$

where $\boldsymbol{w}$ is the standard Brownian motion.[5] The dynamics Eq.(6) is known to converge to $p(\boldsymbol{x})$ as the equilibrium distribution [65, 66]. By setting the step size and the perturbation variance so that Inequality (5) holds, we can approximately draw samples from the distribution with *lower temperature*, as shown below.

By seeing the negative log probability as the energy $E(\boldsymbol{x}) = -\log p(\boldsymbol{x})$, we can see $p(\boldsymbol{x})$ as the Boltzmann distribution with the inverse temperature equal to $\beta = 1$:

$$p_\beta(\boldsymbol{x}) = \frac{1}{Z_\beta} \exp\left(-\beta E(\boldsymbol{x})\right), \tag{7}$$

where $Z_\beta = \int \exp\left(-\beta E(\boldsymbol{x})\right) d\boldsymbol{x}$ is the partition function. The following theorem holds:

[4] $o(\cdot)$ is the "small o" of asymptotic notation, i.e., $o(f(\sigma^2))$ is a function such that $\lim_{\sigma^2 \to 0} o(f(\sigma^2))/f(\sigma^2) = 0$.

[5] Intuitively, Eq.(6) can be derived by letting $\Delta t = \alpha = \delta^2/2$, and computing the velocity from Eq.(1):

$$\begin{aligned}\frac{d\boldsymbol{x}}{dt} &= \lim_{\delta \to 0} \frac{\boldsymbol{x}_{t+\Delta t} - \boldsymbol{x}_t}{\Delta t} \\ &= \lim_{\Delta t \to 0} \left(\boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}) + \sqrt{2\Delta t} \frac{\boldsymbol{\varepsilon}_t}{\Delta t} \right) \\ &= \boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}) + \sqrt{2}\frac{d\boldsymbol{w}}{dt},\end{aligned}$$

where $\boldsymbol{\varepsilon}_t \sim \mathcal{N}_L(\boldsymbol{0}, \boldsymbol{I})$.

Theorem 1: In the limit where $\alpha, \delta^2 \to 0$ with their ratio α/δ^2 kept constant, the sequence of MALA Eq.(1) converges to $p_\beta(\boldsymbol{x})$ for

$$\beta = \frac{2\alpha}{\delta^2}. \tag{8}$$

(Proof) As α and δ^2 go to 0, MALA Eq.(1) converges to the following dynamics:

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}) + \frac{\delta}{\sqrt{\alpha}} \frac{d\boldsymbol{w}}{dt},$$

which is equivalent to

$$\frac{d\boldsymbol{x}}{dt} = \frac{2\alpha}{\delta^2} \boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}) + \sqrt{2}\frac{d\boldsymbol{w}}{dt}. \tag{9}$$

Eq.(9) can be rewritten with the Boltzmann distribution Eq.(7) with the inverse temperature specified by Eq.(8):

$$\frac{d\boldsymbol{x}}{dt} = \boldsymbol{\nabla}_{\boldsymbol{x}} \log p_\beta(\boldsymbol{x}) + \sqrt{2}\frac{d\boldsymbol{w}}{dt}.$$

Comparing this equation with Eq.(6), we find that this dynamics converges to the equilibrium distribution $p_\beta(\boldsymbol{x})$. □

Theorem 1 states that the ratio between α and δ^2 effectively controls the temperature. Specifically, we can see MALA Eq.(1) as a discrete approximation to the Langevin dynamics converging to the distribution given by

$$p_{2\alpha/\delta^2}(\boldsymbol{x}) = \frac{p^{2\alpha/\delta^2}(\boldsymbol{x})}{\int p^{2\alpha/\delta^2}(\boldsymbol{x}) d\boldsymbol{x}},$$

of which the probability mass is more concentrated than $p(\boldsymbol{x})$ if Inequality (5) holds.

Our proposed *Langevin cooling* (L-Cool) strategy uses DAE for estimating the gradient, and applies MALA for $\beta > 1$ to cool down test samples before DT is performed. As illustrated in Figure 2, this yields a small move of the test sample towards high density areas in the source domain. Since the DNN for DT is expected to be well trained on the high density areas, such a small move can result in a significant improvement of the translated image in the target domain, and thus enhances the DT performance. We show qualitative and quantitative performance gain by L-Cool in the subsequent sections.

B. Extensions

We can choose two options for L-Cool, depending on the application and computational resources.

1) Fringe Detection: We can apply fringe detection, in the same way as adversary detection [67]. Namely, assuming that the gradient of $\log p(\boldsymbol{x})$ is large at the fringe of the data distribution, we identify samples as fringe if

$$\left\| \boldsymbol{\nabla}_{\boldsymbol{x}} \log p(\boldsymbol{x}) \right\|_2 > \xi \tag{10}$$

for a threshold $\xi > 0$, and apply MALA only to those samples. This prevents non-fringe samples already lying in high density areas from being perturbed by Langevin dynamics.

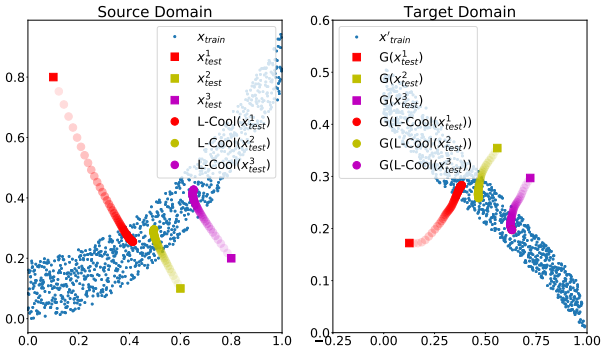


Fig. 3: Toy data demonstration of L-Cool, which drives test samples, $x^1_{test}, x^2_{test}, x^3_{test}$, towards the data manifold in the source domain (left). This makes the translated samples $\boldsymbol{G}(x^1_{test}), \boldsymbol{G}(x^2_{test}), \boldsymbol{G}(x^3_{test})$ by CycleGAN more typical in the target domain (right).

2) Gradient Estimation by Cycle: Another option is to omit to train DAE, and estimate the gradient by a cycle structure that the DNN for DT already possesses. This idea follows the argument in [62], where MALA is successfully used to generate high-resolution, realistic, and diverse artificial images. The authors argued that DAE for estimating the gradient can be replaced with any cycle (autoencoding) structure in their application. In our image translation experiment, we use CycleGAN as the base method, and therefore, we can estimate the gradient by

$$\nabla_{\boldsymbol{x}} \log p(\boldsymbol{x}) \approx \widehat{\boldsymbol{g}}_{\mathrm{Cycle}}(\boldsymbol{x}) \equiv \gamma \left(\boldsymbol{F}(\boldsymbol{G}(\boldsymbol{x})) - \boldsymbol{x}\right) \quad (11)$$

for some $\gamma > 0$, where $\boldsymbol{G}$ corresponds to the mapping of the CycleGAN from the source domain to the target domain and $\boldsymbol{F}$ to its inversion. We call this option L-Cool-Cycle, which eliminates the necessity of training DAE. However, one should use this option with care: we found that L-Cool-Cycle tends to exacerbate artifacts created by CycleGAN, which will be discussed in detail in Section V-E.

IV. Demonstration with Toy Data

We first show the basic behavior of L-Cool on toy data. We generated $1,000$ training samples each in the source and the target domains by

$$\boldsymbol{x} = (t, 0.75 \times t^2 + \epsilon), \quad \boldsymbol{x}' = (t', 0.4 \times t' + \epsilon'),$$

respectively, where $t, t' \sim \mathrm{Uniform}(0, 1)$, $\epsilon \sim \mathrm{Uniform}(0, 0.2)$, and $\epsilon' \sim \mathrm{Uniform}(0, 0.1)$. Then, a CycleGAN [31] with two-layer feed forward networks, $\boldsymbol{G}(\boldsymbol{x}) \to \widehat{\boldsymbol{x}}'$ and $\boldsymbol{F}(\boldsymbol{x}') \to \widehat{\boldsymbol{x}}$, were trained to learn the forward and the inverse mappings between the two domains. A DAE having the same architecture as $\boldsymbol{G}$ with two-layer feed forward network was also trained on the samples in the source domain.

Blue dots in Figure 3 show training samples, from which we can see the high density areas both in the source (right) and the target (left) domains. Now we feed three off-manifold test samples $x^1_{test}, x^2_{test}, x^3_{test}$, shown as red, yellow, and magenta squares in the left graph, to the forward (source to target) translator $\boldsymbol{G}$. As expected, the translated samples $\boldsymbol{G}(x^1_{test}), \boldsymbol{G}(x^2_{test}), \boldsymbol{G}(x^3_{test})$, shown as red, yellow, and magenta squares in the right graph, are not in the high density area (not typical target samples), because $\boldsymbol{G}$ was not trained for those off-manifold samples. As shown as trails of circles, L-Cool drives the off-manifold samples into the data manifold in the source domain, which also drives the translated samples into the data manifold in the target domain. This way, L-Cool helps CycleGAN generate typical samples in the target domain by making source samples more typical.

V. Image Translation Experiments

Next, we demonstrate the performance of L-Cool in several image translation tasks. We use CycleGAN as the base translation method, and L-Cool is performed in the source image space before translation (Figure 4).

A. Translation Tasks and Model Architectures

We used pretrained CycleGAN models, along with the training and the test datasets, publicly available in the official github repository[6] of CycleGAN [31]. Experiments were conducted on the following tasks.

horse2zebra Translation from horse images to zebra images and vice versa. The training set consists of 1067 horse images and 1334 zebra images, subsampled from ImageNet. Dividing the test set, we prepared 60 and 70 validation images and 60 and 70 test images for horse and zebra, respectively.

apple2orange Translation from apple images to orange images and vice versa. The training set consists of 995 apple images and 1019 orange images, subsampled from ImageNet. Dividing the test set, we prepared 133 and 133 validation images and 133 and 133 test images for apple and orange, respectively.

sat2map Translation from satellite images to map images. The training set consists of 1096 satellite images and 1096 map images, subsampled from Google Maps. 1098 and 1098 images each are provided for test. Dividing the test set, we prepared 250 validation images and 848 test images. Although CycleGAN was pretrained in the unsupervised setting, the dataset is actually paired, i.e., the ground truth map image for each satellite image is available, which allows quantitative evaluation.

For the first two tasks, we also conducted experiments on the inverse tasks, i.e., zebra2horse and orange2apple. The validation images were used for hyperparameter tuning for L-Cool (see Section V-D).

The CycleGAN model consists of a forward mapping $\boldsymbol{G}$ and a reverse mapping $\boldsymbol{F}$. Both $\boldsymbol{G}$ and $\boldsymbol{F}$

[6] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

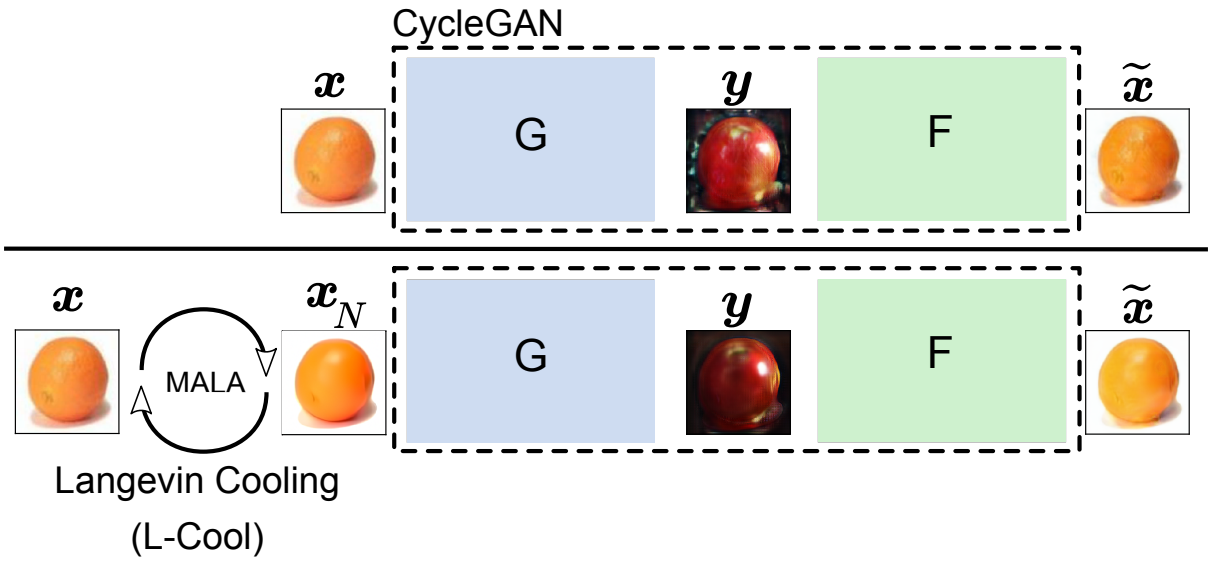


Fig. 4: Schematics of (the plain) CycleGAN (top) and L-Cool (bottom). In CycleGAN, an encoder, $\boldsymbol{y} = \boldsymbol{G}(\boldsymbol{x})$, translates a source sample to a target sample, while a decoder, $\widetilde{\boldsymbol{x}} = \boldsymbol{F}(\boldsymbol{y})$, translates the target sample back to the source sample. In L-Cool, a source sample is cooled down by MALA, before being translated by CycleGAN.

have the same architecture including 2 downsampling layers followed by 9 resnet generator blocks and 2 upsampling layers. Each resnet generator block consists of convolution, instance normalization [68] and relu layers with residual connections added between every block. While training with a batch size of 1 using instance normalization is equivalent to using batch normalization [69], instance normalization has been shown to help improve the results for image stylization [68] as well as for image translation [31, 46, 47]. For reproducing the results of CycleGAN, we utilize the pretrained models provided by the authors on their official github repository. The network architecture and training strategies of CycleGAN are also shared by CUT and GcGAN. For CUT and GcGAN, we use the code provided in their respective official github repositories[7][8] for training the models.

For DAE, we adapted a tiramisu model [70] consisting 67 layers in total. The PyTorch [71] code for tiramisu was obtained from a publicly available github repository[9]. The tiramisu consists of 5 downsampling layers followed by a bottleneck layer and 5 upsampling layers. Each downsampling as well as upsampling layer consists of dense blocks with a growth rate of 16. Each dense block consists of batch normalization [69], relu and convolution layers with dense connections [72]. We trained the DAE on the training images in the source domain for 200 epochs by the Adam optimizer with the learning rate set to 0.0002.

B. Qualitative Evaluation

Figure 5 shows some example results of horse2zebra, zebra2horse, apple2orange, and orange2apple tasks. For each example, we compare L-Cool with (the plain) CycleGAN [31], CUT [47], and GcGAN [46], which are state-of-the-art in these tasks. As other baselines, we also evaluated two edge-preserving smoothing techniques, median filter [73] and total variation denoising [74] which are applied before translation by CycleGAN. These baselines are supposed to move test samples towards the high density areas to some extent. Below each input image (in the first column), we report as a fringeness measure the percentile ρ of the norm of the score function (see Eq.(10)) among the whole test set. Higher values of ρ implies more fringeness.

We see in Figure 5 that, while in some cases median filter and total variation denoising show improvement in target domain attributes (e.g. increased zebra stripes for the task of horse2zebra), in many other cases it gives no improvement (e.g. for the task of apple2orange) or worsens the output image quality (e.g. for the task of zebra2horse). Although smoothing methods can be broadly considered as a projection towards the high density region, the destination can still be outside the training data manifold. L-Cool on the other hand is multi-step projection towards the data distribution, whose dynamics can be controlled by tuning the hyperparameters—the number of steps N, the step size α and the temperature β^{-1}. Thus, in our experiments, we find that the projection by L-Cool results in better translation performance than the edge-preserving smoothing methods.

[7]https://github.com/taesungp/contrastive-unpaired-translation
[8]https://github.com/hufu6371/GcGAN
[9]https://github.com/bfortuner/pytorch_tiramisu

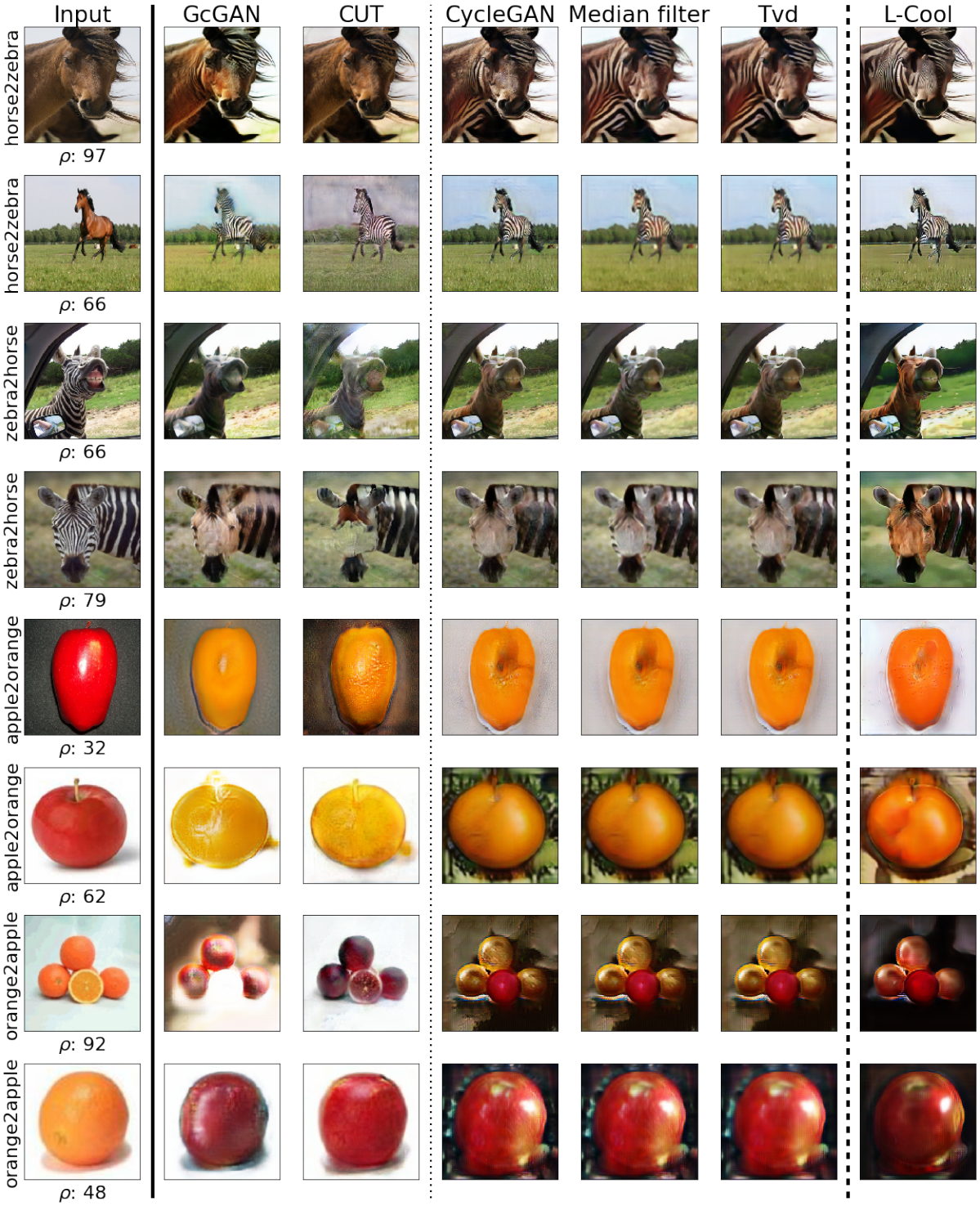


Fig. 5: Example results of image translation tasks. For each example (row) the left most figure shows the input test image with its fringeness ρ—the percentile of the norm of the score function as shown in Eq.(10) (higher ρ indicates that the image is further from the manifold). The six right columns show translated images in the target domain. Median filter, TVD, and L-Cool are applied to the input image before translation by CycleGAN. In each task and in each image, we find that the translations provided by L-Cool better represent the target domain attributes.

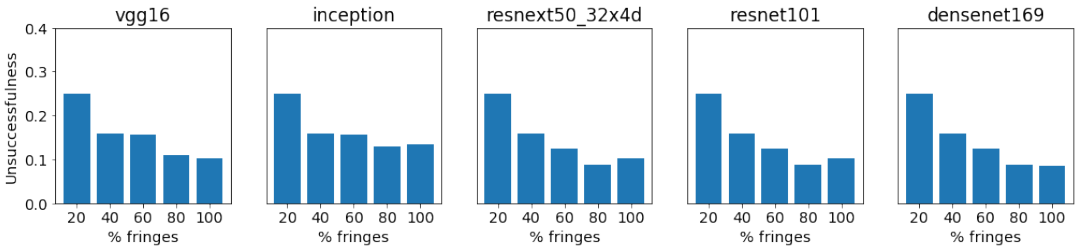


Fig. 6: Fringeness vs. unsuccessfulness of image translation by plain CycleGAN in the horse2zebra task. Here, %fringe indicates the proportion of the samples identified as fringe by the detector (10) (and therefore lower %fringe indicates higher fringeness of the evaluated set of samples). The unsuccessfulness is measured by the proportion of the samples for which the probability output $p(\boldsymbol{y} = \text{zebra}|\boldsymbol{x})$ of the classifier for the translated image is smaller than 0.1. With all 5 different pre-trained classifiers, we consistently observe clear correlation between the fringeness and the unsuccessfulness of image translation by CycleGAN.

We also find in Figure 5 that for the task of horse2zebra, L-Cool provides for an increase in the zebra stripes while reducing the artifacts, while CycleGAN, CUT and GcGAN produce fewer or no zebra stripes or suffer from severe artifacts. This pattern is also observed for the task of zebra2horse where the results of L-Cool show increased brown color of the horse compared to all other methods. L-Cool significantly increases the orange color of the output image for the task of apple2orange. Similarly, the apple images in the results for the task of orange2apple show reduced artifacts and significantly better target domain attributes by L-Cool than any other method.

In general, our qualitative experiment shows that L-Cool performs better than or comparable to the baseline methods. However, we also observe its side effect—the translated image can be over-saturated in some cases. This is because of imperfect training of the sore function estimator—Langevin dynamics with normally trained DAE does not converge to the training distribution when the number of steps is large [75]. Recent study has overcome this issue by *convergent learning* applied to energy-based models (EBM), with which Langevin dynamics converge to the target distribution and generated samples are not oversaturated [76]. Unfortunately, the current techniques for convergent learning of EBM are applicable only to small networks. We expect that scaling convergent learning will remedy this side effect of L-Cool in the near future.

C. Quantitative Evaluation

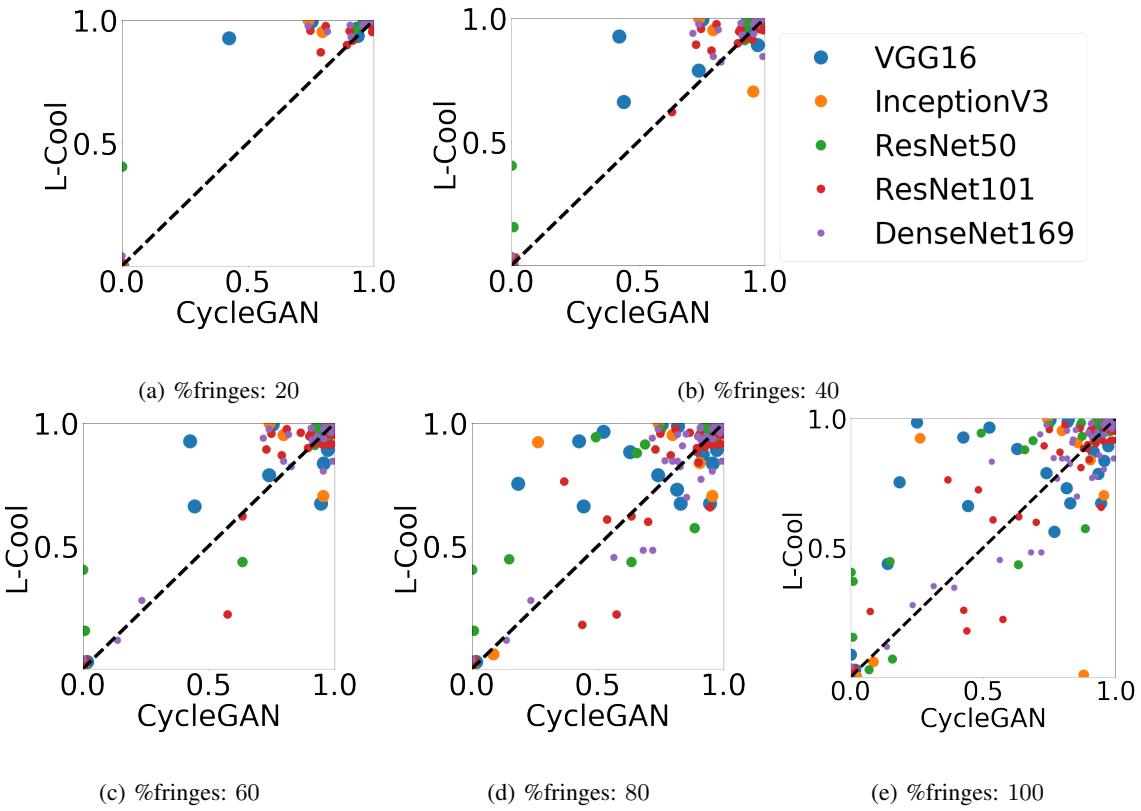
In order to confirm that L-Cool generally improves the image translation performance, we conducted two experiments that quantitatively evaluate the performance.

1) Likeness Evaluation by Pretrained Classifiers: Focusing on horse2zebra, we evaluated the likeness of the translated images to zebra images by using state-of-the-art classifiers, including VGG16 [77], InceptionV3 [78], Resnet50 [79], Resnet101 [80], and Densenet169 [72] pretrained on the ImageNet dataset [81]. Specifically, we evaluated and compared the probability outputs (i.e., after softmax) of the classifiers for the translated images by plain CycleGAN and those by L-Cool. We applied fringe detection, Eq.(10), with the threshold ξ adjusted so that specified proportions (20%, 40%, 60%, 80%, and 100%) of the test samples are identified as fringe. Note that 100% fringe samples correspond to the whole test samples, while 20% fringe samples correspond to the 20% samples that are farthest from the data manifold identified by the fringe detector (10). Our strategy of L-Cool with the fringe detector is to apply L-Cool only to the fringe samples.

Let us first check if one of our hypotheses — unsuccessful translation tends to happen for fringe samples — holds. Figure 6 shows the relation between the fringeness and the unsuccessfulness of image translation in the horse2zebra task. Here the unsuccessfulness is measured by the proportion of the samples for which the probability output $p(\boldsymbol{y} = \text{zebra}|\boldsymbol{x})$ of the classifier for the translated image is smaller than 0.1. We consistently observe clear correlation between the fringeness and the unsuccessfulness of image translation by the plain CycleGAN, which supports our first hypothesis.

Next we show that the translation performance can be improved by applying L-Cool to fringe samples — our second hypothesis. Figure 7 shows scatter plots of likeness to zebra images, i.e., the probability $p(\boldsymbol{y} = \text{zebra}|\boldsymbol{x})$ evaluated by pretrained classifiers. The five panels respectively plot the $20, 40, 60,$ 80 and 100% fringe samples. In each plot, the horizontal axis corresponds to the output probabilities of the transferred images by CycleGAN, while the vertical axis corresponds to the output probabilities of the transferred images by L-Cool. The dashed line indicates the equal-probability, i.e., the points above the dashed line imply the improvement by L-Cool.

We observe that all classifiers tend to give higher probability to the images translated after L-Cool is applied. We emphasize that L-Cool uses no information on the target domain—DAE is trained purely on the samples in the source domain, and MALA drives samples towards high density areas in the source



(a) %fringes: 20 (b) %fringes: 40

(c) %fringes: 60 (d) %fringes: 80 (e) %fringes: 100

Fig. 7: Likeness to zebra images evaluated by the probability output $p(\boldsymbol{y} = \text{zebra}|\boldsymbol{x})$ of pretrained classifiers for the translated images by CycleGAN (horizontal axis) and by L-Cool (vertical axis). Each panel plots the fringe samples detected by Eq.(10) with the threshold ξ controlling the proportion of the test samples identified as fringe. Points above the equal-likeness dashed line implies improvement by L-Cool compared to CycleGAN. We can see that, consistently for all classifiers (shown in different colors), points tend to be above the equal-likeness dashed line implying improvement by L-Cool.

TABLE II: Average likeness to zebra images over the fringe samples and the classifiers (shown in the legend in Figure 7). The fringe samples are detected by Eq.(10) with the threshold ξ controlling %fringes. For each row, we mark in bold the best method and the methods that are not significantly outperformed by the best, according to the Wilcoxon signed rank test for $p = 0.05$.

% fringes	CycleGAN	L-Cool
20	0.6910	**0.7385**
40	0.7872	**0.8145**
60	**0.8023**	**0.8167**
80	**0.8138**	**0.8331**
100	**0.8022**	**0.8211**

TABLE III: Average pixel-wise accuracy in the sat2map task. The fringe samples are detected by Eq.(10) with the threshold ξ controlling %fringes For each row, we mark in bold the best method and the methods that are not significantly outperformed by the best, according to the Wilcoxon signed rank test for $p = 0.05$.

%fringes	CycleGAN	L-Cool
20	61.83	**62.76**
40	65.95	**66.37**
60	66.37	**67.54**
80	68.56	**68.76**
100	68.83	**69.05**

domain, independently from the translation task. The hyperparameters for the Langevin dynamics were set to $\alpha = 0.005$, $\beta^{-1} = 0.001$ and $N = 40$, which were found optimal on the validation set (see Section V-D).

Table II shows the average of the output probabilities over the fringe samples and over the five classifiers for plain CycleGAN (second column) and for L-Cool (third column). Here again, we use the fringe detector (10) to identify a proportion, indicated by %fringes, of the test samples as fringe. We observe in Table II that, for smaller proportions of fringe samples (i.e. most outlying samples), the performance of the plain CycleGAN is worse, and the performance gain, i.e., the differences between L-Cool and CycleGAN, is larger. These observations empirically support our hypothesis that CycleGAN does not perform well on fringe samples, and cooling down those samples can improve the translation performance.

2) Evaluation on Paired-data: As mentioned in Section VI-A, sat2map dataset consists of pairs of

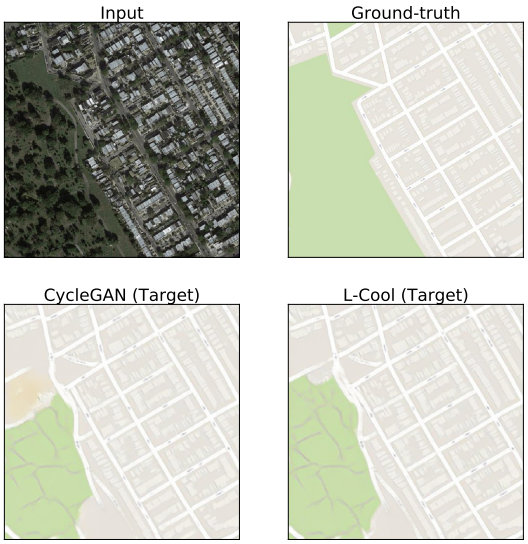


Fig. 8: An example of sat2map image translation result. The green regions are increased in the output of L-Cool (bottom right) compared to that of CycleGAN (bottom left). As a result, the output of L-Cool is closer to the ground-truth map (top right).

satellite images and the corresponding map images, and therefore allows us to directly evaluate image translation performance. We applied the pretrained CycleGAN to the test satellite images with and without L-Cool, and compared the transferred map images with the corresponding ground-truth map images. Following the evaluation procedure in [42], we counted pixels as *correct* if the color mismatch (i.e., the Euclidean distance between the transferred map and the ground-truth map in the RGB color space) is below 16.

Table III shows the average pixel-wise accuracy, where we observed a similar tendency to the likeness evaluation in Section V-C1: for smaller proportions of fringe samples, the translation performance of the plain CycleGAN is worse, and the performance gain by L-Cool is larger. This again supports our hypothesis that driving the fringe samples towards the data manifold is beneficial for improving the the performance of the base domain translation method. Figure 8 shows an exemplar case where L-Cool improves translation performance.

D. Hyperparameter Setting

L-Cool has several hyperparameters. For DAE training, we set the training noise to $\sigma = 0.3$ for all tasks, which approximately follows the recommendation (10% of the mean pixel values) in [62]. We visually inspected the performance dependence on the remaining hyperparameters, i.e., temperature β^{-1}, step size α, and the number of steps N. Roughly speaking, the product of α and N determines how far the resulting image can reach from the original point, and similar results are obtained if $\alpha \cdot N$ has similar values, as long as the step size α is sufficiently small.

Figure 9 shows exemplarily translated images in the orange2apple task, where the dependence on the temperature β^{-1} and the step size α is shown for the number of steps fixed to $N = 100$. We observed that, as the step size α increases, the translated image gets more attributes—increased red color on the apple—of the target domain, and artifacts are reduced. However, if α is too large, the image gets blurred. We also observed that too high temperature β^{-1} gives noisy result. The visually best result was obtained when $\beta^{-1} = 0.001$, $\alpha = 0.005$ and $N = 100$ (marked with a green box and plotted on the right most in Figure 9). Similar tendency was observed in other test samples and other tasks.

For quantitative evaluations in Section V-C, we optimized the hyperparameters on the validation set. The reported results were obtained with the hyperparameters searched over $\beta^{-1} = 0.0001, 0.001, 0.005, 0.01$, $\alpha = 0.001, 0.005, 0.01$, and $N = 20, 40, 60, 80, 100$.

E. Investigation on the L-Cool-Cycle

L-Cool requires a trained DAE for gradient estimation. However, a variant, introduced in Section III-B2 as an option called L-Cool-Cycle, eliminates the necessity of DAE training, and estimate the gradient by using the autoencoding structure of CycleGAN. This option empirically showed good performance in image generation [62], as well as in our preliminary experiments in image translation [39].

Suboptimality of L-Cool-Cycle can already be seen in the toy data experiment. Figure 10 shows the same demonstration as in Figure 3, and compares trails by L-Cool and L-Cool-Cycle. We see that L-Cool (red) drives the off-manifold samples directly towards the closest points in the data manifold that are expected to be semantically similar than the farther points. On the other hand, L-Cool-Cycle (green) does not always do so. This implies that the cycle estimator Eq.(11) is not a very good gradient estimator. Although L-Cool-Cycle is an option when training DAE is hard or time-consuming, it should be used in care—resulting samples should be checked by human.

VI. Language Translation Experiments

In this section, we demonstrate the performance of our proposed L-Cool in language translation tasks with cross-lingual language model (XLM) [18, 54]—a state-of-the-art method for unsupervised language translation—as the base method.

A. Translation Tasks and Model Architectures

We conducted experiments on four language translation tasks, EN-FR, FR-EN, EN-DE, and DE-EN, based on NewsCrawl dataset[10] under the default setting defined in the github repository page:[11] for each pair of

[10] http://www.statmt.org/wmt14/
[11] https://github.com/facebookresearch/XLM

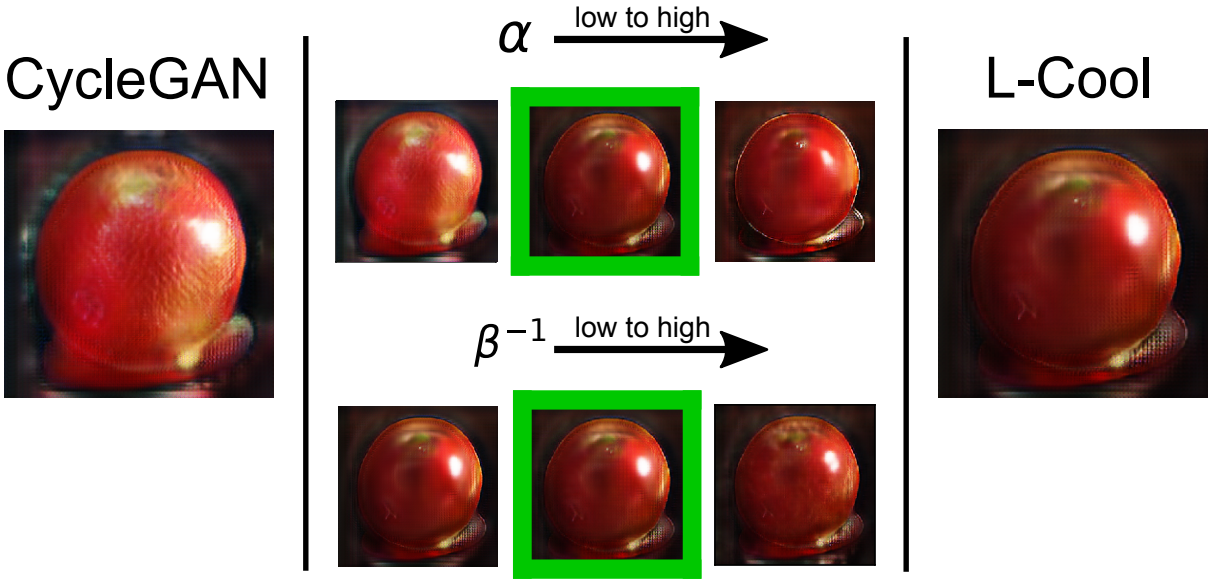


Fig. 9: Translated images by L-Cool with different hyperparameter settings. We found that the setting $\beta^{-1} = 0.001$, $\alpha = 0.005$, and $N = 100$ (marked with a *green* bounding box) best removes artifacts and increases the target domain attributes.

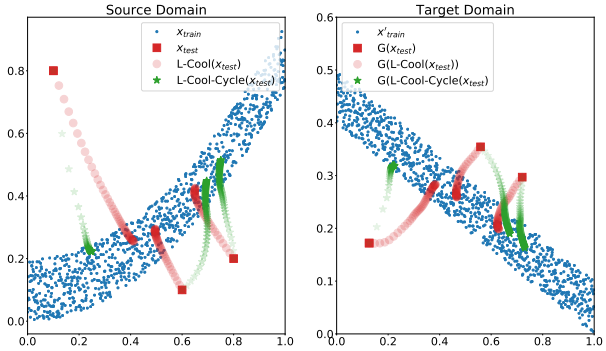


Fig. 10: The same toy data demonstration as Figure 3, comparing L-Cool (red) and L-Cool-Cycle (green). L-Cool moves samples towards the closest points that are expected to be semantically similar than the farther points. In contrast, L-Cool-Cycle does not move samples directly towards the high density region in the source domain, implying that the cycle gradient estimator is not a very good substitution for DAE gradient estimator.

languages, we used the first 5M sentences for training, 3000 sentences for validation, and 3000 sentences for test.

The main idea of XLM is to share sub-word vocabulary between the source and the target languages created through the byte pair encoding (BPE). Masked language modeling (MLM) is performed as pretraining, similarly to BERT [52]. 15% of the BPE from the text stream is masked 80% of the time, by a random token 10% of the time and they are kept unchanged 10% of the time. The encoder is pretrained with the MLM objective, whose weights are then used as initialization for both the encoder and the decoder. This pretraining strategy was shown to give the best results [18].

The transformer consists of 6 encoders and 6 decoders. The architectures of encoders and decoders are similar, and each consists of a multi-head attention layer followed by layer normalization [82], 2 fully connected layers with gelu activations [83] and another layer normalization. While the first fully connected layer projects the input with a dimensionality of 1024 to a latent dimension of 4096, the second fully connected layer projects it back to 1024. Each encoder and decoder layer also consists of a residual connection. For XLM implementation, we use the code publicly available at the github page. We train the model by using the ADAM optimizer along with linear warm-up and linear learning rates. We warm start with the model weights obtained after the MLM stage, and further train the weights on the training sentences.

B. L-Cool Variants

We tested two variants of L-Cool (see Figure 11).

L-Cool-Input: MALA is performed in the input word embedding space (the position embeddings are unaffected), similarly to the image translation experiments in Sections V-B and V-C.

L-Cool-Feature: MALA is performed in the intermediate feature (code) space.

In both the variants, DAE with the same architecture as the encoder of the transformer was trained in the corresponding space on the training sentences of NewsCrawl. Hyperparameters were tuned on the validation sentences (see Section VI-D).

L-Cool-Feature was motivated by our preliminary observation that L-Cool-Input rarely improves the

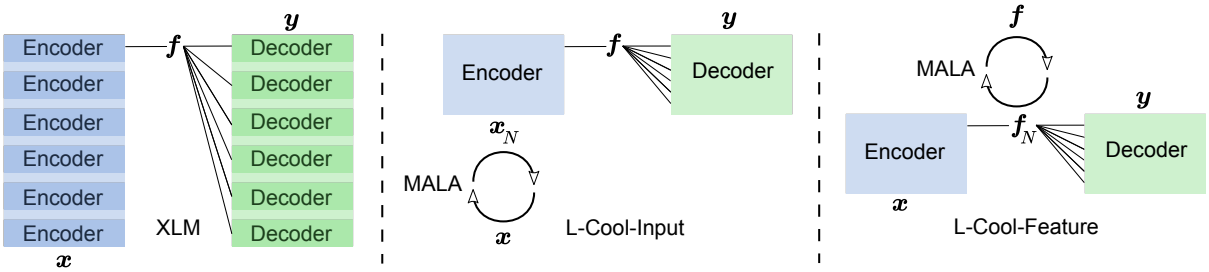


Fig. 11: Schematics of XLM (left), L-Cool-Input (middle), and L-Cool-Feature (right). L-Cool-Input performs MALA in the input space, while L-Cool-Feature performs MALA in the feature (code) space between the encoder and the decoder.

TABLE IV: BLEU scores in language translation tasks on the test set. For each task (column), we mark in bold the best method and the methods that are not significantly outperformed by the best, according to the Wilcoxon signed rank test for $p = 0.05$.

	EN-FR	FR-EN	EN-DE	DE-EN
XLM (Baseline)	33.46	31.62	25.51	30.89
L-Cool-Input	31.59	**31.90**	25.66	30.93
L-Cool-Feature	**33.91**	**31.93**	**25.73**	**31.17**

language translation performance, as will be shown in subsequent sections. We hypothesized that this is because of the discrete nature of the input space—the input is the word embedding that depends only on discrete occurrences of words, and therefore, a single step of MALA to any direction can bring the sample to a point where the base transformer is less trained than the original point. This issue might be remedied by applying Langevin dynamics in the feature space where the mapped distribution is already smoothed. Note that L-Cool-Feature would be not suitable when paired data is not available for hyperparameter tuning (like in our image translation experiments except Section V-C2). This is because driving samples in the feature space can drastically change the corresponding input, and thus the translated result can become unrelated to the original input, unless the hyperparameters are tuned with paired-data.

C. Quantitative Evaluation

Table IV shows the BLEU scores [40] by plain XLM, L-Cool-Input, and L-Cool-Feature. We find that L-Cool-Feature shows consistent performance gain and outperforms XLM on all the four language translation tasks. On the other hand, L-Cool-Input does not improve the performance over XLM, except for the task of FR-EN.

Focusing on L-Cool-Feature in the EN-FR task, we evaluated the translation performance with the fringe detector. Similarly to the image translation experiments in Section V-C, we applied the fringe detector (10) with the threshold ξ controlling the proportion of the fringe samples (%fringes) in the test set. Table VI-C shows the BLEU scores in the EN-FR translation task by XLM and by L-Cool-Feature on the fringe samples with different %fringes. We observe a similar tendency to the image translation experiments: for smaller %fringes (hence for sample sets with higher fringeness), the translation performance by the plain XLM is worse, and the performance gain, i.e., the difference between L-Cool-Feature and XLM is larger, which empirically supports our hypothesis also in this language application.

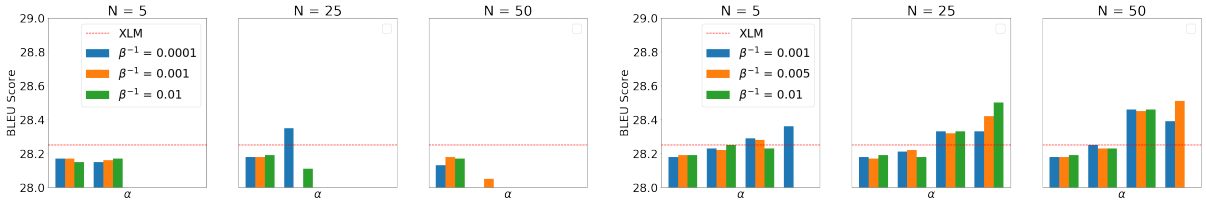
TABLE V: BLEU scores in the EN-FR translation task with fringe detection. Different proportions of fringe samples are identified by the fringe detector (10) with adjusted threshold ξ. In each row, we mark in bold the best method and the methods that are not significantly outperformed by the best, according to the Wilcoxon signed rank test for $p = 0.05$.

% fringes	XLM	L-Cool-Feature
20	32.71	**33.62**
40	32.86	**33.63**
60	32.59	**33.77**
80	33.50	**33.90**
100	33.46	**33.91**

D. Hyperparameter Setting

Similarly to Section V-D, we set the DAE training noise to $\sigma^2 = 0.1$ for L-Cool-Input and $\sigma^2 = 1.0$ for L-Cool-Feature, which approximately follow the recommendation in [62]. The remaining hyperparameters, i.e., the temperature β^{-1}, the step size α, and the number of steps N, were tuned by maximizing the BLEU score on the validation sentences. The search ranges were $\beta^{-1} = 0.0001, 0.0005, 0.001, 0.005, 0.01$, $\alpha = 0.001, 0.005, 0.01,\ 0.05,\ 0.1$ and $N = 5, 25, 50$, respectively.

The destination by L-Cool must be close enough to the original point, in order not to change the semantics of the original sentence. This is achieved by tuning the hyperparameters effectively on the validation set. Additionally, a user also has the possibility to stop the sequence generation (e.g. a rejection step) if the L_p



(a) L-Cool-Input (b) L-Cool-Feature

Fig. 12: Language translation performance (BLEU score) dependence on hyperparameters in the EN-FR task with L-Cool-Input (left) and L-Cool-Feature (right) on the validation set. The dashed line in each graph indicates the baseline performance by plain XLM.

distance between the input and a sample from the step of MALA is beyond a predetermined threshold.

Figure 12 shows performance dependence on the hyperparameters for L-Cool-Input (left) and L-Cool-Feature (right) in the EN-FR translation task, where the best performance was obtained when $\beta^{-1} = 10^{-4}, \alpha = 10^{-5}, N = 25$ for L-Cool-Input, and when $\beta^{-1} = 10^{-3}, \alpha = 10^{-2}, N = 25$ for L-Cool-Feature.

VII. Computation Time

L-Cool requires additional computation cost both in training and test. Training the DAE can typically be done much faster than training the base DNN for the domain translation. In our experiment for the horse2zebra image translation task, training the DAE took ~ 12800 seconds or 3.55 hours, while training the CycleGAN typically takes ~ 42320 seconds or 11.75 hours (we did not train it because we used a pretrained network provided by the authors of CycleGAN). Note that this additional training is not necessary for L-Cool-Cycle, which substitutes the cycle structure of the base DNN for gradient estimation. In the test time, L-Cool requires 10 to 100 times more computation time, depending on the number of MALA steps. This is because DAE should have a similar structure and complexity to the base DNN. In our image translation experiment, L-Cool and CycleGAN took ~ 5.3 seconds and ~ 0.5 seconds per test image, respectively, while in the language translation experiment, L-Cool and XLM took ~ 0.047 seconds and ~ 0.013 per test sentence, respectively.

VIII. Conclusion

Developing unsupervised, as well as self-supervised, learning methods, is one of the recent hot topics in the machine learning community for computer vision [84, 85, 86, 87, 88] and natural language processing [51, 52, 89, 90, 91]. It is challenging but highly attractive since eliminating the necessity of labeled data may enable us to keep improving learning machines from data stream automatically without any human intervention. The successes of deep learning in the unsupervised domain translation (DT) was a milestone in this exciting research area.

Our work contributes to this area with a simple idea. Namely, Langevin cooling (L-Cool) performs Metropolis-adjusted Langevin algorithm (MALA) to test samples in the source domain, and drives them towards high density manifold, where the base deep neural network is well-trained. Our qualitative and quantitative evaluations showed improvements by L-Cool in image and language translation tasks, and the evaluations of L-Cool with fringe detection, i.e., applying L-Cool only to the detected fringe samples, supported our hypothesis that a proportion of test samples are failed to be translated because they lie at the fringe of data distribution, and therefore can be improved by L-Cool.

L-Cool is generic and can be used to improve any DT method. Future work is therefore to apply L-Cool to other base DT methods and other DT tasks. We will also try to improve the gradient estimator for L-Cool by using other types of generative models such as normalizing flows [92]. Explanation methods, such as layer-wise relevance propagation (e.g. [93, 94, 95]), might help identify the reasons for successes and failures [96] of DT, suggesting possible ways to improve the performance.

IX. Acknowledgements

The authors acknowledge financial support by the German Ministry for Education and Research (BMBF) for the Berlin Center for Machine Learning (01IS18037A), Berlin Big Data Center (01IS18025A) and under the Grants 01IS14013A-E, 01GQ1115 and 01GQ0850; Deutsche Forschungsgemeinschaft (DFG) under Grant Math+, EXC 2046/1,Project ID 390685689 and by the Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea Government (No. 2019-0-00079, Artificial Intelligence Graduate School Program, Korea University). Correspondence to WS, SN and KRM.

References

[1] R. Biswas, M. K. Sen, V. Das, and T. Mukerji, "Prestack and poststack inversion using a physics-guided convolutional neural network," *Interpretation*, vol. 7, no. 3, pp. SE161–SE174, 2019.

[2] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, "Neural message passing for quantum chemistry," in *International Conference on Machine Learning*, 2017, pp. 1263–1272.

[3] K. Schütt, P.-J. Kindermans, H. E. S. Felix, S. Chmiela, A. Tkatchenko, and K.-R. Müller, "Schnet: A continuous-filter convolutional neural network for modeling quantum interactions," in *Advances in neural information processing systems*, 2017, pp. 991–1001.

[4] K. Schütt, M. Gastegger, A. Tkatchenko, K.-R. Müller, and R. J. Maurer, "Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions," *Nature Communications*, vol. 10, no. 1, pp. 1–10, 2019.

[5] G. Zhang, Z. Wang, and Y. Chen, "Deep learning for seismic lithology prediction," *Geophysical Journal International*, vol. 215, no. 2, pp. 1368–1387, 2018.

[6] J. Schmidt, M. R. Marques, S. Botti, and M. A. Marques, "Recent advances and applications of machine learning in solid-state materials science," *npj Computational Materials*, vol. 5, no. 1, pp. 1–36, 2019.

[7] S. Arridge, P. Maass, O. Öktem, and C.-B. Schönlieb, "Solving inverse problems using data-driven models," *Acta Numerica*, vol. 28, pp. 1–174, 2019.

[8] T. A. Bubba, G. Kutyniok, M. Lassas, M. März, W. Samek, S. Siltanen, and V. Srinivasan, "Learning the invisible: A hybrid deep learning-shearlet framework for limited angle computed tomography," *Inverse Problems*, vol. 35, no. 6, p. 064002, 2019.

[9] F. Codevilla, M. Müller, A. López, V. Koltun, and A. Dosovitskiy, "End-to-end driving via conditional imitation learning," in *International Conference on Robotics and Automation*. IEEE, 2018, pp. 1–9.

[10] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," in *Conference on Robot Learning*, 2017, pp. 1–16.

[11] Unknown, "Developers, start your engines," 2020. [Online]. Available: https://aws.amazon.com/deepracer/

[12] D. Gray, "Introducing voyage deepdrive," 2019. [Online]. Available: https://news.voyage.auto/introducing-voyage-deepdrive-69b3cf0f0be6

[13] B. McMahan and D. Ramage, "Federated learning: Collaborative machine learning without centralized training data," 2017. [Online]. Available: https://ai.googleblog.com/2017/04/federated-learning-collaborative.html-g/

[14] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey *et al.*, "Google's neural machine translation system: Bridging the gap between human and machine translation," *arXiv preprint arXiv:1609.08144*, 2016.

[15] Unknown, "Game intelligence," 2020. [Online]. Available: https://www.microsoft.com/en-us/research/theme/game-intelligence/

[16] J. Johnson, A. Alahi, and L. Fei-Fei, "Perceptual losses for real-time style transfer and super-resolution," in *European Conference on Computer Vision*. Springer, 2016, pp. 694–711.

[17] D. He, Y. Xia, T. Qin, L. Wang, N. Yu, T.-Y. Liu, and W.-Y. Ma, "Dual learning for machine translation," in *Advances in neural information processing systems*, 2016, pp. 820–828.

[18] A. Conneau and G. Lample, "Cross-lingual language model pretraining," in *Advances in Neural Information Processing Systems*, 2019, pp. 7059–7069.

[19] S. Edunov, M. Ott, M. Auli, and D. Grangier, "Understanding back-translation at scale," *Empirical Methods in Natural Language Processing*, pp. 489–500, 2018.

[20] L. A. Gatys, A. S. Ecker, and M. Bethge, "A neural algorithm of artistic style," *arXiv preprint arXiv:1508.06576*, 2015.

[21] C. Dong, C. C. Loy, K. He, and X. Tang, "Image super-resolution using deep convolutional networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 2, pp. 295–307, 2015.

[22] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Computer Vision and Pattern Recognition*, 2017, pp. 1125–1134.

[23] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Deep image prior," in *Computer Vision and Pattern Recognition*, 2018, pp. 9446–9454.

[24] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," *arXiv preprint arXiv:1605.05396*, 2016.

[25] H. Zhang, T. Xu, H. Li, S. Zhang, X. Wang, X. Huang, and D. N. Metaxas, "Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks," in *International Conference on Computer Vision*, 2017, pp. 5907–5915.

[26] V. Sandfort, K. Yan, P. J. Pickhardt, and R. M. Summers, "Data augmentation using generative adversarial networks (cyclegan) to improve generalizability in ct segmentation tasks," *Scientific reports*, vol. 9, no. 1, pp. 1–9, 2019.

[27] E. Wu, K. Wu, D. Cox, and W. Lotter, "Conditional infilling gans for data augmentation in mammogram classification," in *Image Analysis for Moving Organ, Breast, and Thoracic Images*. Springer, 2018, pp. 98–106.

[28] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Synthetic data augmentation

using gan for improved liver lesion classification," in *International Symposium on Biomedical Imaging*. IEEE, 2018, pp. 289–293.

[29] C. Bowles, L. Chen, R. Guerrero, P. Bentley, R. Gunn, A. Hammers, D. A. Dickie, M. V. Hernández, J. Wardlaw, and D. Rueckert, "Gan augmentation: Augmenting training data using generative adversarial networks," *arXiv preprint arXiv:1810.10863*, 2018.

[30] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[31] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *International Conference on Computer Vision*, 2017, pp. 2223–2232.

[32] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to discover cross-domain relations with generative adversarial networks," in *International Conference on Machine Learning*. JMLR. org, 2017, pp. 1857–1865.

[33] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," in *International Conference on Computer Vision*, 2017, pp. 2849–2857.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems*, 2017, pp. 5998–6008.

[35] "Cyclegan," https://github.com/junyanz/CycleGAN#failure-cases.

[36] Y. A. Mejjati, C. Richardt, J. Tompkin, D. Cosker, and K. I. Kim, "Unsupervised attention-guided image-to-image translation," in *Advances in Neural Information Processing Systems*, 2018, pp. 3693–3703.

[37] Y. Kim, M. Graça, and H. Ney, "When and why is unsupervised neural machine translation useless?" in *European Association for Machine Translation*, 2020, pp. 35–44.

[38] G. Alain and Y. Bengio, "What regularized auto-encoders learn from the data-generating distribution," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 3563–3593, 2014.

[39] V. Srinivasan, K.-R. Müller, W. Samek, and S. Nakajima, "Benign examples: Imperceptible changes can enhance image translation performance," in *AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 5842–5850.

[40] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, "Bleu: a method for automatic evaluation of machine translation," in *Association for Computational Linguistics*, 2002, pp. 311–318.

[41] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Computer Vision and Pattern Recognition*, 2018, pp. 8798–8807.

[42] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised image-to-image translation networks," in *Advances in neural information processing systems*, 2017, pp. 700–708.

[43] X. Huang, M.-Y. Liu, S. Belongie, and J. Kautz, "Multimodal unsupervised image-to-image translation," in *European Conference on Computer Vision*, 2018, pp. 172–189.

[44] H.-Y. Lee, H.-Y. Tseng, J.-B. Huang, M. Singh, and M.-H. Yang, "Diverse image-to-image translation via disentangled representations," in *European Conference on Computer Vision*, 2018, pp. 35–51.

[45] J. Kim, M. Kim, H. Kang, and K. Lee, "U-GAT-IT: unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation," *CoRR*, vol. abs/1907.10830, 2019. [Online]. Available: http://arxiv.org/abs/1907.10830

[46] H. Fu, M. Gong, C. Wang, K. Batmanghelich, K. Zhang, and D. Tao, "Geometry-consistent generative adversarial networks for one-sided unsupervised domain mapping," in *Computer Vision and Pattern Recognition*, 2019, pp. 2427–2436.

[47] T. Park, A. A. Efros, R. Zhang, and J.-Y. Zhu, "Contrastive learning for unpaired image-to-image translation," in *European Conference on Computer Vision*. Springer, 2020, pp. 319–345.

[48] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[49] M. Artetxe, G. Labaka, E. Agirre, and K. Cho, "Unsupervised neural machine translation," *arXiv preprint arXiv:1710.11041*, 2017.

[50] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[51] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," *OpenAI Blog*, vol. 1, no. 8, p. 9, 2019.

[52] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Association for Computational Linguistics: Human Language Technologies, Volume 1 and Short Papers)*, 2019, pp. 4171–4186.

[53] "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[54] G. Lample, A. Conneau, L. Denoyer, and M. Ranzato, "Unsupervised machine translation using monolingual corpora only," *arXiv preprint arXiv:1711.00043*, 2017.

[55] A. Poncelas, D. Shterionov, A. Way, G. M. d. B. Wenniger, and P. Passban, "Investigating backtranslation in neural machine translation," *arXiv preprint arXiv:1804.06189*, 2018.

[56] G. Lample, M. Ott, A. Conneau, L. Denoyer, and M. Ranzato, "Phrase-based & neural unsupervised machine translation," *arXiv preprint arXiv:1804.07755*, 2018.

[57] J. Heek and N. Kalchbrenner, "Bayesian inference for large scale image classification," *arXiv preprint arXiv:1908.03491*, 2019.

[58] F. Wenzel, K. Roth, B. Veeling, J. Swiatkowski, L. Tran, S. Mandt, J. Snoek, T. Salimans, R. Jenatton, and S. Nowozin, "How good is the Bayes posterior in deep neural networks really?" in *International Conference on Machine Learning*, 2020, pp. 10 248–10 259.

[59] N. Parmar, A. Vaswani, J. Uszkoreit, Ł. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," *International Conference on Machine Learning*, pp. 4055–4064, 2018.

[60] R. Dahl, M. Norouzi, and J. Shlens, "Pixel recursive super resolution," in *International Conference on Computer Vision*, 2017, pp. 5439–5448.

[61] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in neural information processing systems*, 2018, pp. 10 215–10 224.

[62] A. Nguyen, J. Clune, Y. Bengio, A. Dosovitskiy, and J. Yosinski, "Plug & play generative networks: Conditional iterative generation of images in latent space," in *Computer Vision and Pattern Recognition*, 2017, pp. 4467–4477.

[63] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *International Conference on Machine Learning*. ACM, 2008, pp. 1096–1103.

[64] Y. Bengio, L. Yao, G. Alain, and P. Vincent, "Generalized denoising auto-encoders as generative models," in *Advances in Neural Information Processing Systems*, 2013, pp. 899–907.

[65] G. O. Roberts and R. L. Tweedie, "Exponential convergence of Langevin distributions and their discrete approximations," *Bernoulli*, vol. 2, pp. 341–363, 1996.

[66] G. O. Roberts and J. S. Rosenthal, "Optimal scaling of discrete approximations to Langevin diffusions," *Journal of the Royal Statistical Society, Series B*, vol. 60, pp. 255–268, 1998.

[67] V. Srinivasan, C. Rohrer, A. Marban, K.-R. Müller, W. Samek, and S. Nakajima, "Robustifying models against adversarial attacks by langevin dynamics," *Neural Networks*, vol. 137, pp. 1–17, 2021.

[68] D. Ulyanov, A. Vedaldi, and V. S. Lempitsky, "Instance normalization: The missing ingredient for fast stylization," *CoRR*, vol. abs/1607.08022, 2016. [Online]. Available: http://arxiv.org/abs/1607.08022

[69] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.

[70] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *Computer Vision and Pattern Recognition Workshops*, 2017, pp. 11–19.

[71] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*.

[72] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Computer Vision and Pattern Recognition*, 2017, pp. 4700–4708.

[73] G. R. Arce, *Nonlinear signal processing: a statistical approach*. John Wiley & Sons, 2005.

[74] A. Chambolle, "An algorithm for total variation minimization and applications," *Journal of Mathematical imaging and vision*, vol. 20, no. 1, pp. 89–97, 2004.

[75] M. Hill, J. C. Mitchell, and S.-C. Zhu, "Stochastic security: Adversarial defense using long-run dynamics of energy-based models," in *International Conference on Learning Representations*, 2021.

[76] E. Nijkamp, M. Hill, T. Han, S.-C. Zhu, and Y. N. Wu, "On the anatomy of mcmc-based maximum likelihood learning of energy-based models," in *AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 5272–5280.

[77] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *International Conference on Learning Representations*, 2015.

[78] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Computer Vision and Pattern Recognition*, 2016, pp. 2818–2826.

[79] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Computer Vision and Pattern Recognition*, 2017, pp. 1492–1500.

[80] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[81] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical

image database," in *Computer Vision and Pattern Recognition*, 2009, pp. 248–255.

[82] J. L. Ba, J. R. Kiros, and G. E. Hinton, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[83] D. Hendrycks and K. Gimpel, "Bridging nonlinearities and stochastic regularizers with gaussian error linear units," 2016.

[84] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International Conference on Machine Learning*, vol. 119, 2020, pp. 1597–1607.

[85] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," in *Computer Vision and Pattern Recognition*, 2020, pp. 9729–9738.

[86] J.-B. Grill, F. Strub, F. Altché, C. Tallec, P. H. Richemond, E. Buchatskaya, C. Doersch, B. A. Pires, Z. D. Guo, M. G. Azar *et al.*, "Bootstrap your own latent: A new approach to self-supervised learning," *arXiv preprint arXiv:2006.07733*, 2020.

[87] M. Patrick, Y. M. Asano, R. Fong, J. F. Henriques, G. Zweig, and A. Vedaldi, "Multi-modal self-supervision from generalized data transformations," *arXiv preprint arXiv:2003.04298*, 2020.

[88] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," *arXiv preprint arXiv:2003.04297*, 2020.

[89] Q. Xie, Z. Dai, E. Hovy, M.-T. Luong, and Q. V. Le, "Unsupervised data augmentation for consistency training," *arXiv preprint arXiv:1904.12848*, 2019.

[90] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pretraining for language understanding," in *Advances in neural information processing systems*, 2019, pp. 5753–5763.

[91] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro, "Megatron-lm: Training multi-billion parameter language models using gpu model parallelism," *arXiv preprint arXiv:1909.08053*, 2019.

[92] G. Papamakarios, E. Nalisnick, D. J. Rezende, S. Mohamed, and B. Lakshminarayanan, "Normalizing flows for probabilistic modeling and inference," *arXiv preprint arXiv:1912.02762*, 2019.

[93] S. Bach, A. Binder, G. Montavon, F. Klauschen, K.-R. Müller, and W. Samek, "On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation," *PLoS ONE*, vol. 10, no. 7, p. e0130140, 2015.

[94] G. Montavon, W. Samek, and K.-R. Müller, "Methods for interpreting and understanding deep neural networks," *Digital Signal Processing*, vol. 73, pp. 1–15, 2018.

[95] W. Samek, G. Montavon, S. Lapuschkin, C. J. Anders, and K.-R. Müller, "Explaining deep neural networks and beyond: A review of methods and applications," *Proceedings of the IEEE*, vol. 109, no. 3, pp. 247–278, 2021.

[96] S. Lapuschkin, S. Wäldchen, A. Binder, G. Montavon, W. Samek, and K.-R. Müller, "Unmasking clever hans predictors and assessing what machines really learn," *Nature Communications*, vol. 10, no. 1, p. 1096, 2019.