Generative methods for some inverse problems in imaging

Coloma Ballester

Universitat Pompeu Fabra

LMS/ICMS Symposium: Analytic and Geometric Approaches to Machine Leaning, July 26-30, 2021

Collaborators in this work: Patricia Vitoria, Pierrick Chatillon, Lara Raad, Joan Sintes

Outline

1. Motivation and introduction to Generative models
2. HistoryAD: An adversarial approach to anomaly detection
3. Image colorization using adversarial learning and semantic information
4. Semantic image inpainting through improved Wasserstein generative adversarial networks

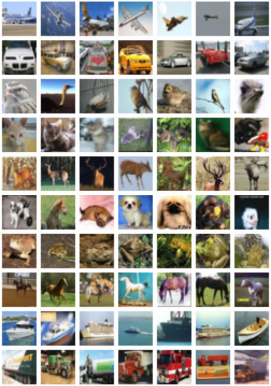
Generative models

- Generative methods aim to estimate the probability distribution of a large set of data $\mathcal{X}$.

- Example: $\mathcal{X}$ is a set of images.

- Theoretically, any $\mathcal{X}$. In practice, $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$.

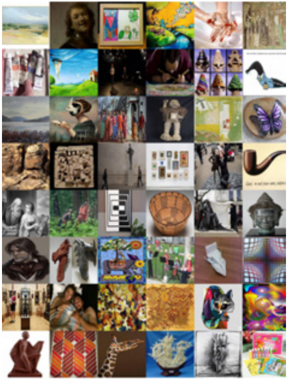
- Assumed: any $x \in \mathcal{X}$ comes from a probability distribution $\mathbb{P}_{\mathcal{X}}$ and the **goal** is to learn it from the data in $\mathcal{X}$.



CIFAR-10

CelebA

SVHN



Tiny ImageNet

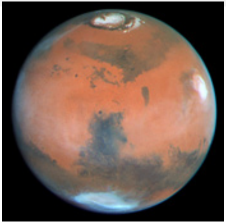
Example: Astronomical images

Images credit: ESA/Hubble & NASA

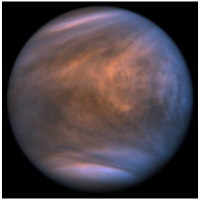
Example: Astronomical images

Images credit: ESA/Hubble & NASA

Anomaly detection. Anomalous?

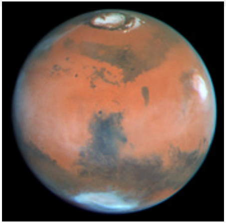




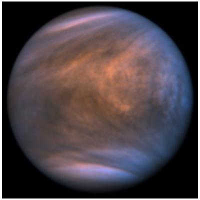



Anomaly detection.

Anomalous?






Out-of-distribution

Anomalous?





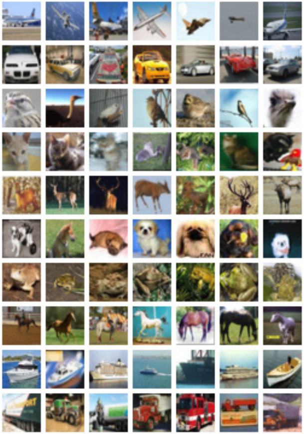


Out-of-distribution

Anomalous? Out-of-distribution

MNIST

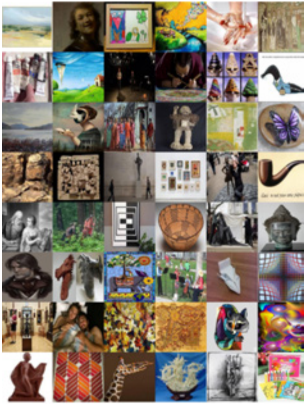
KMNIST



CIFAR-10

CelebA

SVHN



Tiny ImageNet

Generative methods

Generative methods

Generative models learn the probability distribution $\mathbb{P}_{\mathcal{X}}$ of the given data by learning to generate new samples.

Some of the most prominent approaches are

- Normalizing Flows (e.g., L. Dinh et al., 2014)
- Variational Autoencoders (VAE) (e.g., Kingma & Welling, 2013)
- Generative Adversarial Networks (GAN) (e.g., Goodfellow et al., 2014)
- Autoregressive models (e.g., Van den Oord et al., 2016)

Generative approaches

• Although natural images belong to high dimensional spaces, they contain geometric and semantic structure.

• Thus, following [1,2], rather than estimating the density of $\mathbb{P}_{\mathcal{X}}$ (or $\mathbb{P}_{\text{real}}$), which may not exist, we can define a random variable Z with a fixed distribution $\mathbb{P}_z$ and pass it through a **parametric function** $G_\theta : \mathcal{Z} \to \mathcal{X}$ (typically a neural network) that directly generates samples following a certain distribution $\mathbb{P}_{\text{real}}$.

• $\mathbb{P}_G = G_\# \mathbb{P}_Z$, the **pushforward measure** of $\mathbb{P}_Z$ **through** G (parametric density $G_\# \mathbb{P}_Z$ through the neural network G).

• By varying θ, we can change this distribution $\mathbb{P}_{G_\theta}$ and make it close to (converge, if possible) the real data distribution $\mathbb{P}_{\text{real}}$.

[1] Arjovsky, Chintala, and Bottou. Wasserstein GAN. 2017.

[2] Many authors: Peyré, Genevay, Cuturi, Brenier, Dieng, Lunz, Delon, Willett, Dumoulin, Schönlieb, Berthelot, Bengio, and many more.

First of the many GAN's papers (2014):

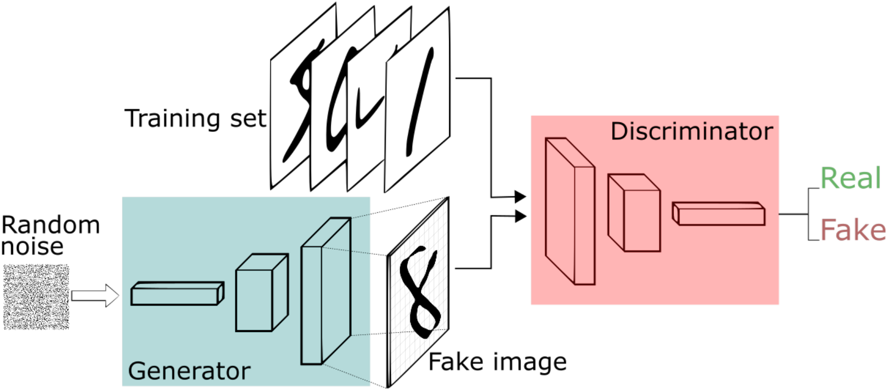
Generative Adversarial Nets

Ian J. Goodfellow,* Jean Pouget-Abadie,† Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair,‡ Aaron Courville, Yoshua Bengio§
Département d'informatique et de recherche opérationnelle
Université de Montréal
Montréal, QC H3C 3J7

Abstract

We propose a new framework for estimating generative models via an adversarial process, in which we simultaneously train two models: a generative model G that captures the data distribution, and a discriminative model D that estimates the probability that a sample came from the training data rather than G. The training procedure for G is to maximize the probability of D making a mistake. This framework corresponds to a minimax two-player game. In the space of arbitrary

GAN Framework



Training GAN

Original GAN objective:

$$\min_G \max_D V(G,D) = \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\text{real}}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_{\boldsymbol{z}}}[\log(1-D(G(\boldsymbol{z}))]$$

Min max iterations: iterate the "two steps" until convergence (which may not happen)

- Updating the discriminator should make it better at discriminating between real images and generated ones (discriminator improves).
- Updating the generator makes it better at fooling the current discriminator (generator improves).

Eventually (we hope) that the generator gets so good that it is impossible for the discriminator to tell the difference between real and generated images. Discriminator guess = 0.5.

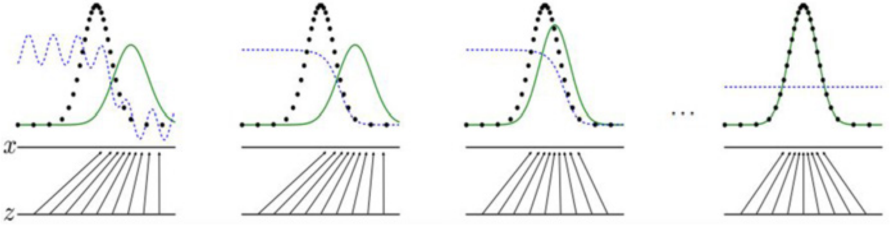


Image credits: Santiago Pascual, 2018

Distances and divergences between probability distributions

- Vanila-GAN training objective:

$$\min_G \max_D V(G,D) = \mathbb{E}_{\boldsymbol{x}\sim\mathbb{P}_{\textsf{real}}}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim\mathbb{P}_{\boldsymbol{z}}}[\log(1 - D(G(\boldsymbol{z})))]$$

- **Under optimal discriminator** $D_G^*(\boldsymbol{x}) = \frac{p_{\textsf{real}}(\boldsymbol{x})}{p_{\textsf{real}}(\boldsymbol{x}) + p_G(\boldsymbol{x})}$,

$$\min_G V(G, D_G^*) = -\log(4) + 2\cdot\delta_{\mathrm{JS}}\left(\mathbb{P}_{\textsf{real}}, \mathbb{P}_G\right)$$

(where $p_{\textsf{real}}, p_G$ densities)

- **Jensen-Shannon Divergence**:

$$\delta_{\mathrm{JS}}(\mathbb{P}_1, \mathbb{P}_2) \triangleq \frac{1}{2}\left[D_{\mathrm{KL}}\left(\mathbb{P}_1 || \frac{\mathbb{P}_1 + \mathbb{P}_2}{2}\right) + D_{\mathrm{KL}}\left(\mathbb{P}_2 || \frac{\mathbb{P}_1 + \mathbb{P}_2}{2}\right)\right]$$

where $\mathbb{P}_1, \mathbb{P}_2 \in \text{Prob}(\mathcal{X})$, space of probability distributions defined on $\mathcal{X}$, $\mathcal{X}$ a compact metric set (e.g., the space of images)

Wasserstein-1 Distance:

$$\mathbb{W}_1(\mathbb{P}_1, \mathbb{P}_2) = \inf_{\pi \in \Pi(\mathbb{P}_1, \mathbb{P}_2)} \mathbb{E}_{x,y\sim\pi}(\|x - y\|).$$

By Kantorovitch-Rubenstein duality:

$$\mathbb{W}_1(\mathbb{P}_1, \mathbb{P}_2) = \sup_{D \in \mathcal{D}} \left(\mathbb{E}_{x\sim\mathbb{P}_1}[D(x)] - \mathbb{E}_{y\sim\mathbb{P}_2}[D(y)]\right),$$

where $\mathcal{D}$ denotes the set of 1-Lipschitz functions (i.e., [1], the set of c-convex functions for the cost function $c(x, y) =| x - y |$).

In practice, the dual variable D is parametrized with some NN D_w

In these articles [2,3], the training objectives are adapted to minimize $\mathbb{W}_1(\mathbb{P}_{\text{real}}, \mathbb{P}_G)$

[1] Villani. Optimal transport: old and new. 2008
[2] Arjovsky, et al. Wasserstein GAN. 2017.

[3] Gulrajani, et al. Improved training of Wasserstein GANs. 2017.

$$\delta(\mathbb{P}_1, \mathbb{P}_2) = \sup_{A \in \mathcal{F}} |\mathbb{P}_1(A) - \mathbb{P}_2(A)|$$

which represents the choice $c(x, y) = \mathbb{1}_{x \neq y}$ in the optimal transport problem [1].
$\delta(\mathbb{P}_1, \mathbb{P}_2) = \frac{1}{2}\|\mathbb{P}_1 - \mathbb{P}_2\|_{TV}$.

Kantorovitch-Rubenstein duality:

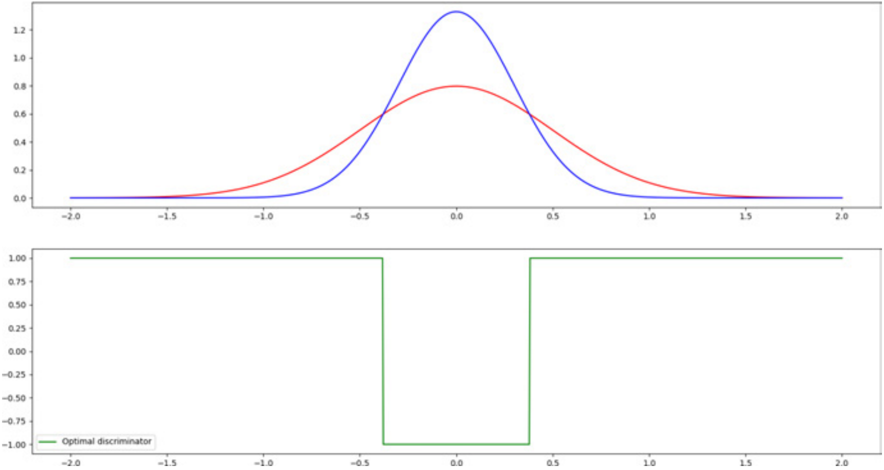
$$\delta(\mathbb{P}_1, \mathbb{P}_2) = \sup_{-1 \leq D \leq 1} \left(\mathbb{E}_{x \sim \mathbb{P}_1}[D(x)] - \mathbb{E}_{y \sim \mathbb{P}_2}[D(y)]\right)$$

Taking $\mu = \mathbb{P}_1 - \mathbb{P}_2$, a signed measure, and (P, N) its Hahn decomposition $(P = \{\mathbb{P}_1 > \mathbb{P}_2\})$, we can define the
optimal dual variable $D^* := \mathbb{1}_P - \mathbb{1}_N$

[1] Villani. Optimal transport: old and new. 2008

Distances. Optimal dual variable

$\mathbb{P}_1$, $\mathbb{P}_2$, and the optimal $D^* = \mathbb{1}_P - \mathbb{1}_N$

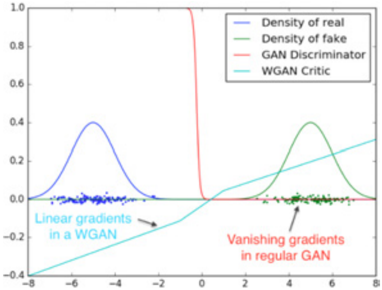


Wasserstein Generative Adversarial Network

• Main vanilla-GANs problems: vanishing gradients, mode collapse [1], non-continuity.

• In these articles [2,3], the training objectives are adapted to minimize $\mathbb{W}_1(\mathbb{P}_{\text{real}}, \mathbb{P}_G)$.

• Wasserstein GAN (WGAN) uses an approximation of the Wasserstein distance. It is continuous everywhere and differentiable almost everywhere.



[1] Arjovsky, and Bottou. Towards principled methods for training generative adversarial networks. 2017
[2] Arjovsky, et al., Wasserstein GAN. 2017.
[3] Gulrajani, et al. Improved training of Wasserstein GANs. 2017.

Theorem. Let $\mathbb{P}_{\text{real}}$ a fixed distribution over $\mathcal{X}$. Let Z be a random variable (e.g Gaussian) over another space $\mathcal{Z}$. Let $G : \mathcal{Z} \times \mathbb{R}^d \to \mathcal{X}$ be a function, that will be denoted $G_\theta(z)$ with z the first coordinate and θ the second. Let $\mathbb{P}_\theta$ denote the distribution of $G_\theta(Z)$. Then,

1. If G is continuous in θ, so is $W(\mathbb{P}_{\text{real}}, \mathbb{P}_\theta)$.
2. If G is locally Lipschitz and satisfies the regularity assumption $\mathbb{E}_{z\sim p}[L(\theta, z)] < +\infty$ on the local Lipschitz constants $L(\theta, z)$, then $W(\mathbb{P}_{\text{real}}, \mathbb{P}_\theta)$ is continuous everywhere, and differentiable almost everywhere.
3. Statements 1-2 are false for the Jensen-Shannon divergence $JS(\mathbb{P}_{\text{real}}, \mathbb{P}_\theta)$ and all the KLs.

The authors show that

- The assumption in 2 is satisfied for any feedforward neural network G_θ, and thus $W(\mathbb{P}_{\text{real}}, \mathbb{P}_\theta)$ is continuous everywhere and differentiable almost everywhere.
- $\nabla_\theta W(P_r, P_\theta) = -\,\mathbb{E}_{z\sim p(z)}[\nabla_\theta f_w(g_\theta(z))]$, when both terms are well defined.

[1] Arjovsky, et al. Wasserstein GAN. 2017.

How to ensure to have a 1-Lipschitz discriminator?

- **WGAN: Weight clipping:**[1] clipping the parameters of the discriminators
  - Problems, including that it reduces the capacity of the discriminator.

- **WGAN-GP: Gradient penalty:**[2] penalizing the norm of discriminator gradients with respect to data samples during training to be less than 1.

$$\min_{G} \max_{D \in \mathcal{D}} \mathbb{E}_{\tilde{x} \sim \mathbb{P}_{real}} \left[D(\tilde{x})\right] - \mathbb{E}_{x \sim \mathbb{P}_G} \left[D(x)\right] - \lambda \mathbb{E}_{\tilde{x} \sim \mathbb{P}_{\tilde{x}}} \left[(\|\nabla_{\tilde{x}} D(\tilde{x})\|_2 - 1)^2\right]$$

where $\mathbb{P}_{\tilde{x}}$ is implicitly defined sampling uniformly along straight lines between pairs of point sampled from the data distribution $\mathbb{P}_{\textbf{real}}$ and the generator distribution $\mathbb{P}_G$.

- The dual variable D is expected to be positive on real data samples and negative on generated ones.

[1] Arjovsky, et al. Wasserstein GAN 2017.

[2] Gulrajani, et al. Improved Training of Wasserstein GANs. 2017.

How can this be used for anomaly detection?

GANs in anomaly detection, main strategies

- If we learned to generate normal data, only normal data can be reconstructed with such a generator [1]

- Use or create an auxiliary dataset of corrupted data (out of distribution) as negative data for a classifier (outlier exposure) [2,3]

- Corrupt the generator to provide anomalies [4]

[1] Schlegl, et al. AnoGAN. Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery. 2017.
[2] Hendrycks, et al. Deep Anomaly Detection with Outlier Exposure. 2019.
[3] Meinke& Hein: Towards neural networks which provably know when they don't know. 2020.

[4] Ngo, et al. Fence GAN: Towards Better Anomaly Detection. 2019.

Our approach

History-based anomaly detector: an adversarial approach to anomaly detection[1]

Joint work with Pierrick Chatillon

[1] P. Chatillon and C. Ballester. History-based anomaly detector: an adversarial approach to anomaly detection. Advances in Intelligent Systems and Computing. 2020.

Our approach

Method's idea: Oscillation during training provides anomalies

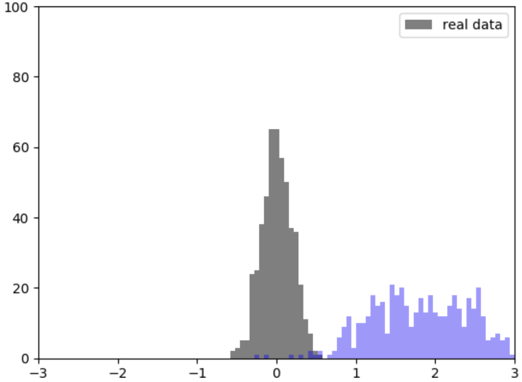


Figure: Generated distribution p_{G_t} oscillating around p_{real}

Method's idea: Oscillation during training provides anomalies

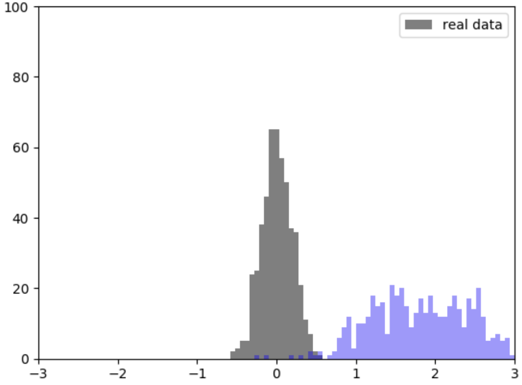


Figure: Generated distribution p_{G_t} oscillating around p_{real}

Our approach

Method's idea: Oscillation during training provides anomalies

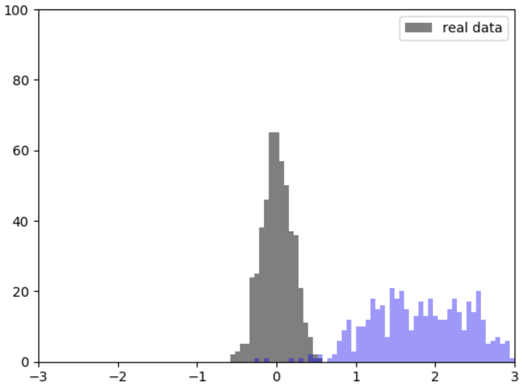


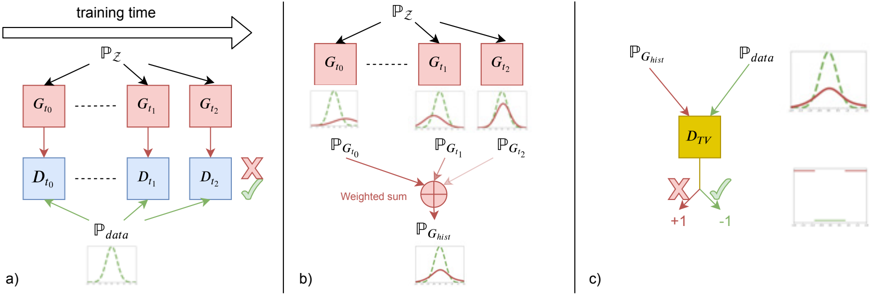
Figure: Generated distribution p_{G_t} oscillating around p_{real}

(a) Train a W-GAN on normal data while saving states

(b) Craft an 'anomalous' distribution

(c) Classify normal and anomalous data with the total variation framework

- We use the obtained classifier D_{TV} as anomaly detector



Which anomalous distribution?

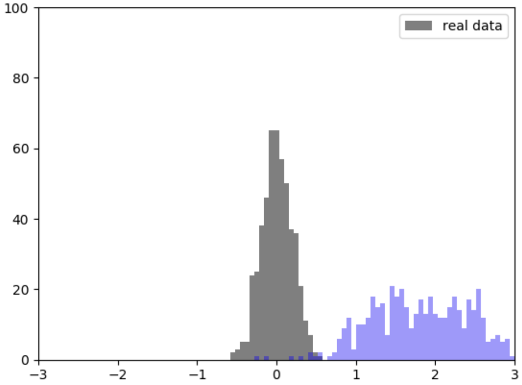


Figure: Generated distribution p_{G_t} oscillating around p_{real}

Hypothesis: $\operatorname{supp}\left(\mathbb{P}_{\mathsf{real}}\right) \subset \operatorname{supp}\left(\mathbb{P}_{G_{\mathsf{hist}}}\right).$

Anomalous distribution

$\mathbb{P}_{G_{\textbf{hist}}}$ is our anomalous distribution: it is a **weighted average** of $\mathbb{P}_{G_t}$ for the different states of G during training.

$$\mathbb{P}_{G_{\textbf{hist}}} \triangleq \int_1^{n_{epoch}} c \cdot G_t(\mathbb{P}_Z) \cdot e^{-\beta t} dt$$

To sample $\mathbb{P}_{G_{\textbf{hist}}}$:

- During W-GAN training, save the Generator's state at regular time steps
- When training D_{TV}, sample t along the exponential distribution, then sample z from $\mathbb{P}_Z$, and finally compute $G_t(z)$.
  i.e., in practice, we approximate $\mathbb{P}_{G_{\textbf{hist}}}$ by sampling data from $\mathbb{P}_{G_t}$ where t is a random variable of density of probability $c \cdot \mathbb{1}_{[\alpha, n_{\text{epochs}}]} \cdot e^{-\beta t}$

$$\textbf{Hypothesis:} \quad \text{supp}\left(\mathbb{P}_{\textsf{real}}\right) \subset \text{supp}\left(\mathbb{P}_{G_{\textbf{hist}}}\right).$$

Training our anomaly detector D_{TV}

D_{TV} training objective:

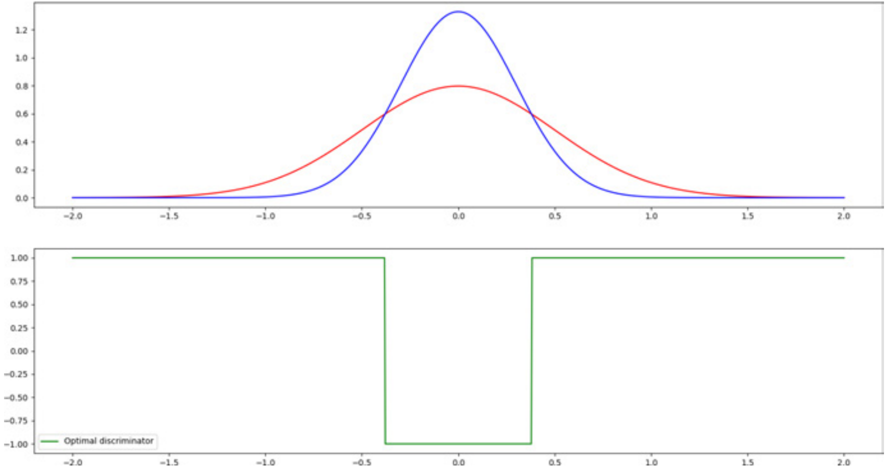
$$\sup_{-1 \leq D \leq 1} \left(\mathbb{E}_{x \sim \mathbb{P}_{\textbf{real}}}[D(x)] - \mathbb{E}_{y \sim \mathbb{P}_{G_{\textbf{hist}}}}[D(y)] \right)$$

For the optimal D_{TV}, the expression above is equal to $\delta(\mathbb{P}_{\textbf{real}}, \mathbb{P}_{G_{\textbf{hist}}})$, where δ is the **total variation distance**:

$$\delta(\mathbb{P}_1, \mathbb{P}_2) = \sup_{A \text{ measurable}} |\mathbb{P}_1(A) - \mathbb{P}_2(A)|$$

Optimal discriminator

$\mathbb{P}_{\text{real}}$, $\mathbb{P}_{G_{\text{hist}}}$, and the optimal D_{TV}



In a nutshell

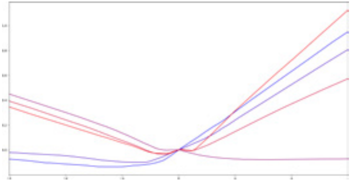
Our method does **not depend** on any specific **perturbation of the GAN objective**, but rather on an **intrinsic property** of GAN training: the **oscillation** of the generated distribution around real data.

Our method can be seen as an extension of the 'early stopping in GANs' [1]

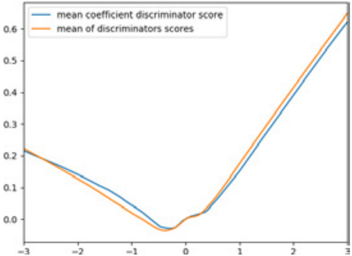
---

[1] Gu, et al. Semi-Supervised Outlier Detection Using a Generative and Adversary Framework. 2018.

- The mean of the different discriminator states is a good candidate
- The network with mean parameters ($W^0_{D_{TV}} = \int_1^{n_{epoch}} c \cdot W_{D_t} \cdot e^{-\beta t} dt$) is a good approximation [1]



(b) Discriminator score output during training (from blue to red)



(c) Comparison of the average outputs of saved discriminators during training and the output of a discriminator with average coefficients.

[1] Wang, et al. Deep Network Interpolation for Continuous Imagery Effect Transition. 2019.

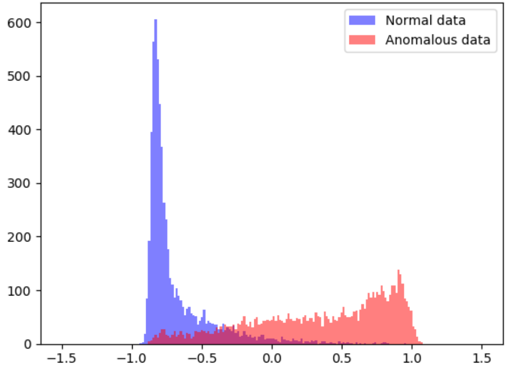
During D_{TV} training, we want $-1 \leq D_{TV} \leq 1$:

✗ non-linearity, bad gradient behavior, as the solution tends to -1 or 1 almost everywhere.

✔ $\lambda_{bounded} \cdot d(x, [-1, 1])^2$, smooth constraint loss term

Experimental results

One sample from each class of MNIST.



Histogram of Discriminator output over testing set (anomalous digit: 1).

- anomaly score of $G((1-t)\mathbf{z}_1 + t\mathbf{z}_2)$

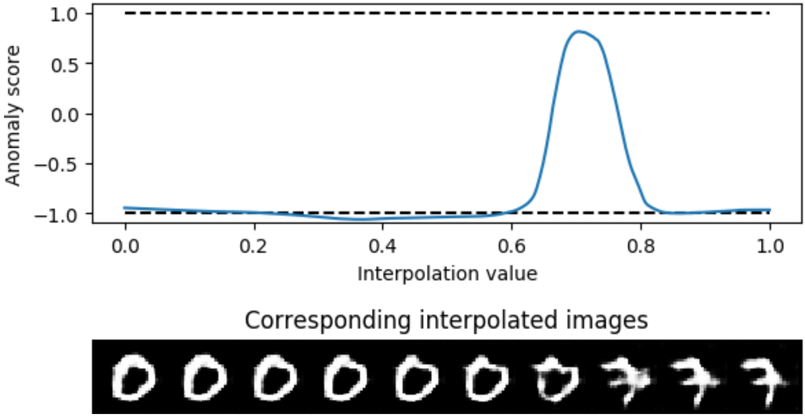


Figure: History-GAN trained on MNIST

Experiments. Latent space linear interpolation

- anomaly score of $G((1-t)\mathbf{z}_1 + t\mathbf{z}_2)$

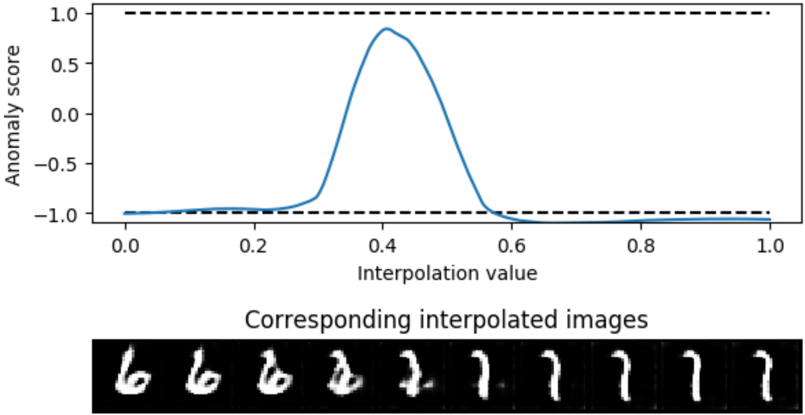


Figure: History-GAN trained on MNIST

Experiments. Gaussian Noise on normal data

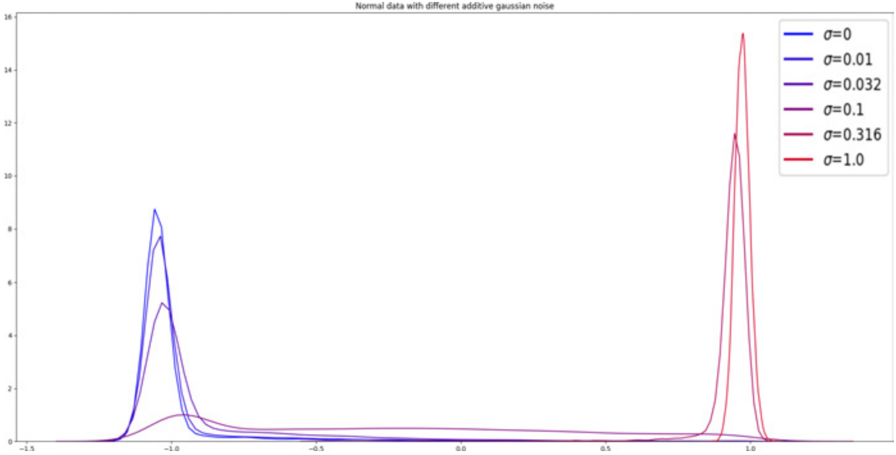


Figure: Density of distribution of anomaly scores

Results. Comparison to others on MNIST experiment

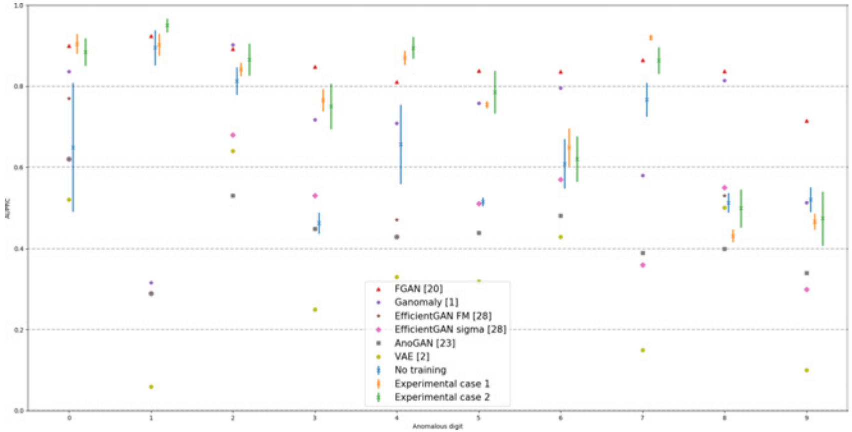
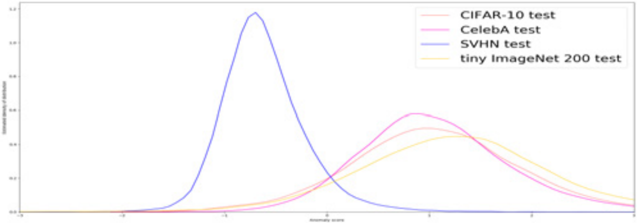


Figure: mean AUPRC for each digit of MNIST for experimental case 2, compared with other methods (performances of methods provided by the authors of Fence-GAN)

Method trained on SVHN and evaluated on several datasets



Approximate density of anomaly score distribution

test split	CIFAR-10	CelebA	Tiny ImageNet
AUPRC	0.941	0.976	0.949

Table AUPRC for SVHN compared to other datasets.

3. Image colorization using adversarial learning and semantic information

Joint work with Patricia Vitoria and Lara Raad

[1] P. Vitoria, L. Raad and C. Ballester. ChromaGAN: Adversarial Picture Colorization with Semantic Class Distribution. WACV. 2020.

Image and video colorization

Photograph: BBC/Wingnut Films/IWM

Problem statement

$L \in \mathbb{R}^{H \times W \times 1}$

$L \in \mathbb{R}^{H \times W \times 1}$

$(a, b) \in \mathbb{R}^{H \times W \times 2}$

$(L, a, b) \in \mathbb{R}^{H \times W \times 3}$

Image colorization

- Scribble-based
- Exemplar-based
- Automatic methods

- Scribble-based
- Exemplar-based
- Automatic methods

Automatic methods - deep learning

- Semantic information
  - Iizuka, Simo-Serra, and Ishikawaizuka, 2016 [ISSI16]
- Color distribution
  - Zhang, Isola, and Efros, 2016 [ZIE16]
  - Larsson, Maire, and Shakhnarovich, 2016 [LMS16]
- Adversarial training
  - Isola, Zhu, Zhou, and Efros, 2017 [IZZE17]
- Instance colorization
  - Su, Chu, and Huang, 2020 [SCH20]

Our approach: ChromaGAN

- Given a grayscale input image L, we learn a mapping $\mathcal{G} : L \longrightarrow (a, b)$ such that $I = (L, a, b)$ is a plausible color image and a and b are chrominance channel images in the CIE Lab color space. A plausible color image is one having geometric, perceptual and semantic photo-realism.

- the mapping (generator) $\mathcal{G}_\theta$ is learnt by means of an adversarial learning strategy.

- In parallel, a discriminator D_w evaluates how realistic is the proposed colorization $I = (L, a, b)$ of L.

- Our generator $\mathcal{G}_\theta$ will not only learn to generate color but also a class distribution vector, denoted by $y \in \mathbb{R}^m$, where m is the fixed number of classes. This provides information about the probability distribution of the semantic content and objects present in the image.

$\mathcal{G}_\theta = (\mathcal{G}^1_{\theta_1}, \mathcal{G}^2_{\theta_2})$, where $\theta = (\theta_1, \theta_2)$ stand for all the generator parameters, $\mathcal{G}^1_{\theta_1} : L \longrightarrow (a, b)$, and $\mathcal{G}^2_{\theta_2} : L \longrightarrow y$.

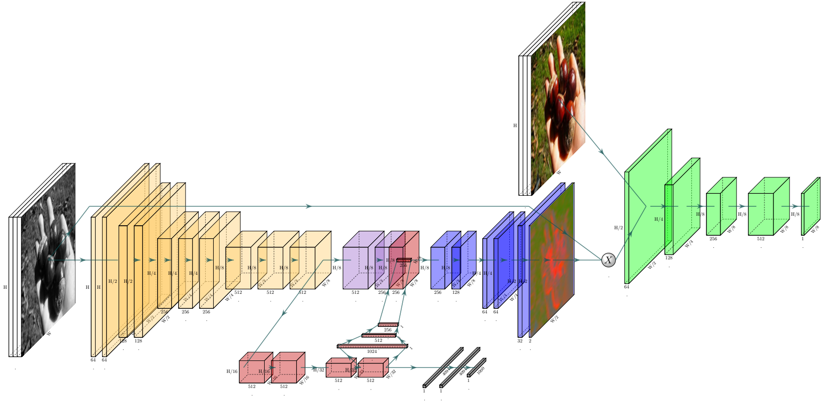


Figure: ChromaGAN network overview: Two outputs, $\mathcal{G}_\theta = (\mathcal{G}^1_{\theta_1}, \mathcal{G}^2_{\theta_2})$, where $\theta = (\theta_1, \theta_2)$ stand for all the generator parameters, $\mathcal{G}^1_{\theta_1} : L \longrightarrow (a, b)$, and $\mathcal{G}^2_{\theta_2} : L \longrightarrow y$.

Vitoria, Raad, and Ballester, 2020 [VRB20]

Cost function

The network is trained solving the min-max problem

$$\min_{\mathcal{G}_\theta} \max_{D_w \in \mathcal{D}} \mathcal{L}(\mathcal{G}_\theta, D_w),$$

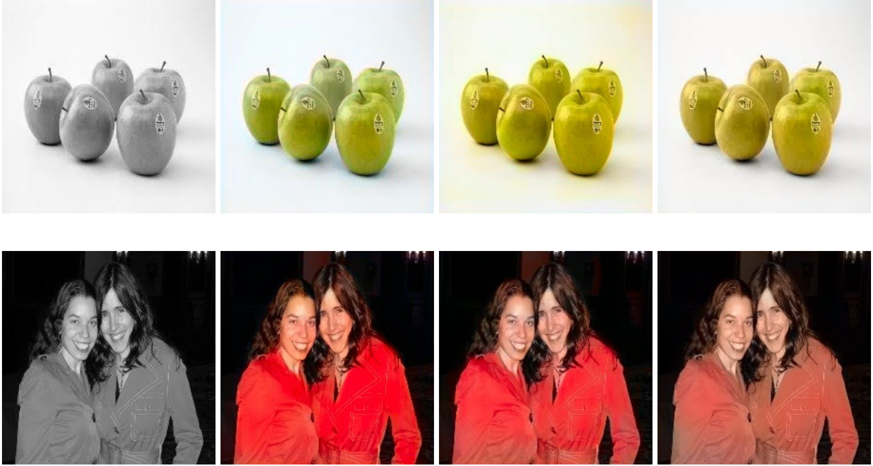
where the loss function is defined as

$$\mathcal{L}(\mathcal{G}_\theta, D_w) = \mathcal{L}_{\mathsf{e}}(\mathcal{G}^1_{\theta_1}) + \lambda_{\mathsf{p}} \mathcal{L}_{\mathsf{p}}(\mathcal{G}^1_{\theta_1}, D_w) + \lambda_{\mathsf{s}} \mathcal{L}_{\mathsf{s}}(\mathcal{G}^2_{\theta_2}).$$

where

- Reconstruction loss: $\mathcal{L}_{\mathsf{e}}(\mathcal{G}^1_{\theta_1}) = \mathbb{E}_{(L,a_r,b_r)\sim\mathbb{P}_r}\left[\|\mathcal{G}^1_{\theta_1}(L) - (a_r, b_r))\|_2^2\right]$
- Class distribution loss: $\mathcal{L}_{\mathsf{s}}(\mathcal{G}^2_{\theta_2}) = \mathbb{E}_{L\sim\mathbb{P}_{rg}}\left[\mathsf{KL}\left(y_{\mathsf{v}} \,\|\, \mathcal{G}^2_{\theta_2}(L)\right)\right]$
- WGAN-GP loss: $\mathcal{L}_{\mathsf{p}}(\mathcal{G}^1_{\theta_1}, D_w) =$
  $\mathbb{E}_{I_r\sim\mathbb{P}_r}[D_w(I_r)] - \mathbb{E}_{(a,b)\sim\mathbb{P}_{\mathcal{G}^1_{\theta_1}}}[D_w(L,a,b)] - \mathbb{E}_{\hat{I}\sim\mathbb{P}_{\hat{I}}}[(\|\nabla_{\hat{I}} D_w(\hat{I})\|_2 - 1)^2]$

Ablation study



input | ChromaGAN | without distribution term | without WGAN term

Qualitative results

Code: `https://github.com/pvitoria/ChromaGAN`

Perceptual evaluation

Colorization Tests

article | demo | archive

Please cite the reference article if you publish results obtained with this online demo.

Does this image has a natural colorization?
Try not to spend too much time looking at the details.
Five seconds per image should be enough.
You can use keys y and n or click the 'Yes' and 'No' buttons.

Yes No

Method	Naturalness	PSNR (dB)
Real images	87.1	
ChromaGAN [VRB20]	76.9	24.98
without class distr	70.9	25.04
without WGAN	61.4	**25.57**
Iizuka *et al.* [ISSI16]	53.9	23.69
Larsson *et al.* [LMS16]	53.6	24.93
Zhang *et al.* [ZIE16]	52.2	22.04
Isola *et al.* [IZZE17]	27.6	21.57

Table: Semantic information: [ISSI16], color distribution: [LMS16], [ZIE16], adversarial training: [IZZE17].

Qualitative comparison

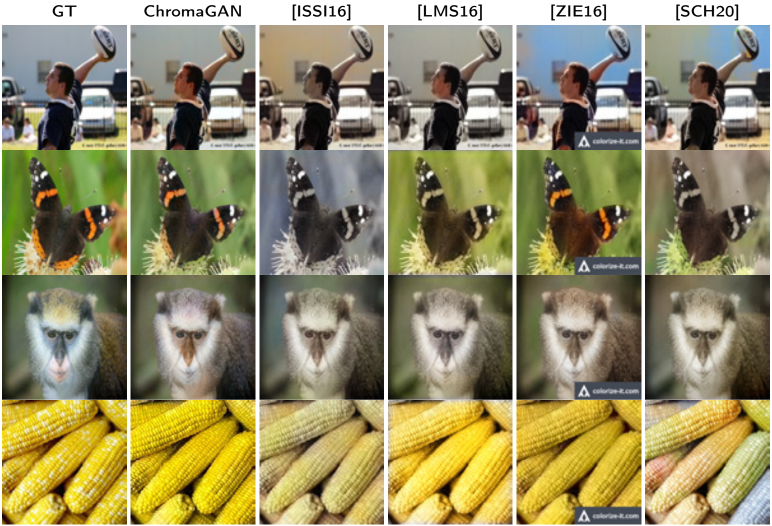


Figure: Results on Imagenet. Semantic information: [ISSI16], color distribution: [LMS16], [ZIE16], instance colorization: [SCH20].

Qualitative comparison

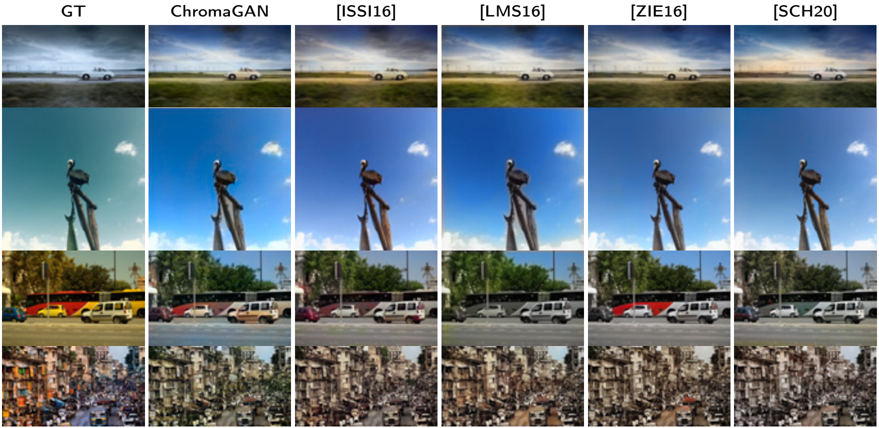


Figure: Results on random images. Semantic information: [ISSI16], color distribution: [LMS16], [ZIE16], instance colorization: [SCH20].

Failure cases

Color Bleeding

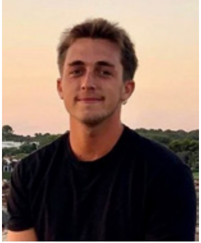
Desatured Results

Color Inconsistency

Old black and white photos

4. Image inpainting through an adversarial strategy

Joint work with Patricia Vitoria and Joan Sintes

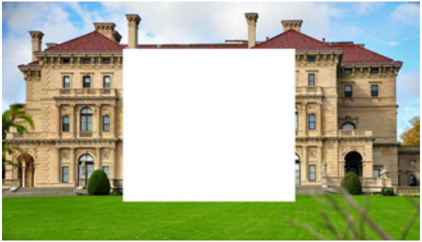
 

[1] P. Vitoria, J. Sintes and C. Ballester. Semantic Image Inpainting Through Improved Wasserstein Generative Adversarial Networks. 2019.

Image inpainting

- Image inpainting is also known as image completion, disocclusion or object removal. It aims to obtain a visually plausible completion of the image in a region in which data is missing due to damage or occlusion.

- <u>Problem:</u> When missing regions are large and moreover the missing information is unique in the sense that the information and redundancy available in the image is not useful to guide the completion, the task becomes even more challenging.



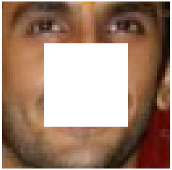


Image inpainting

- Image inpainting is also known as image completion, disocclusion or object removal. It aims to obtain a visually plausible completion of the image in a region in which data is missing due to damage or occlusion.

- Problem: When missing regions are large and moreover the missing information is unique in the sense that the information and redundancy available in the image is not useful to guide the completion, the task becomes even more challenging.

Semantic image inpainting

Method, based on an self-supervised adversarial strategy followed by an energy-based completion algorithm:

- 1st Step: given a **dataset of (non-corrupted) images**, the data latent space is learned via an improved version of the Wasserstein GAN

$$\min_{G} \max_{D \in \mathcal{D}} \mathbb{E}_{\tilde{x} \sim \mathbb{P}_{real}} \left[D(\tilde{x}) \right] - \mathbb{E}_{x \sim \mathbb{P}_G} \left[D(x) \right] - \lambda \mathbb{E}_{\tilde{x} \sim \mathbb{P}_{\hat{x}}} \left[(\|\nabla_{\hat{x}} D(\hat{x})\|_2 - 1)^2 \right]$$

- 2nd Step: given an **incomplete image** y and the converged generative adversarial G and D, a minimization procedure is performed to infer the missing content of the incomplete image by conditioning on the known regions

$$\hat{z} = \arg\min_{z} \{ \mathcal{L}_c(z|y, M) + \alpha \mathcal{L}_p(z) \}$$

where $\mathcal{L}_c$ is the **contextual loss** defined as

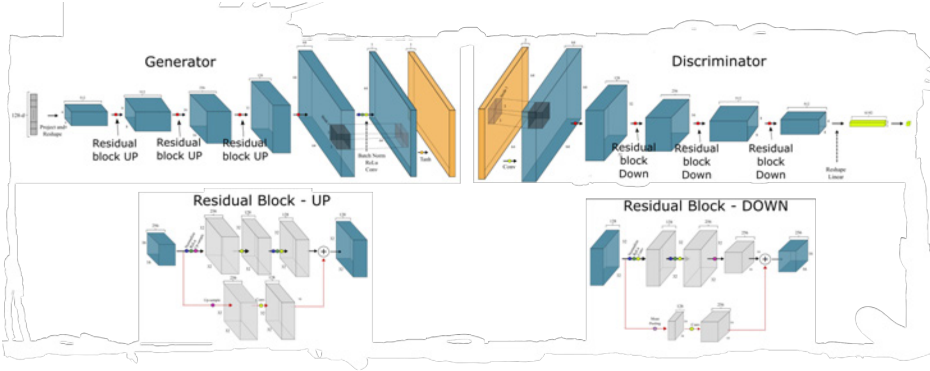
$$\mathcal{L}_c(z|y, M) = \|W(G(z) - y)\| + \beta \|W(\nabla G(z) - \nabla y)\|$$

with $\alpha, \beta > 0$, W a weight mask, $W(i) = \begin{cases} \sum_{j \in N_i} \dfrac{(1 - M(j))}{|N_i|} & \text{if } i \text{ is known} \\ 0 & \text{if } i \text{ is unknown} \end{cases}$

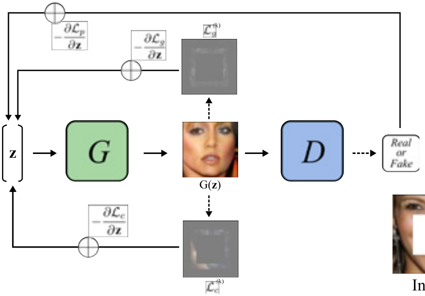
and $\mathcal{L}_p(z) = -D(G(z))$ is the **prior loss** that favours realistic images, similar to the samples that are used to train the generative model.

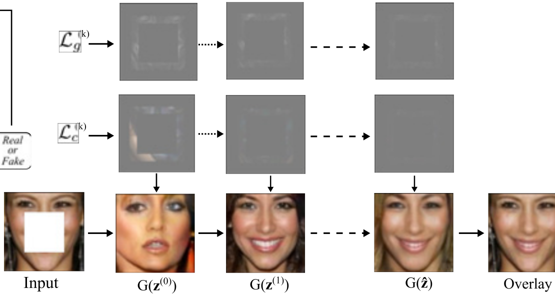
1st Step: Train our generative model

Architecture (based on the one of WGAN-GP plus some improvements)



2nd Step: Perform inpainting using an optimization method

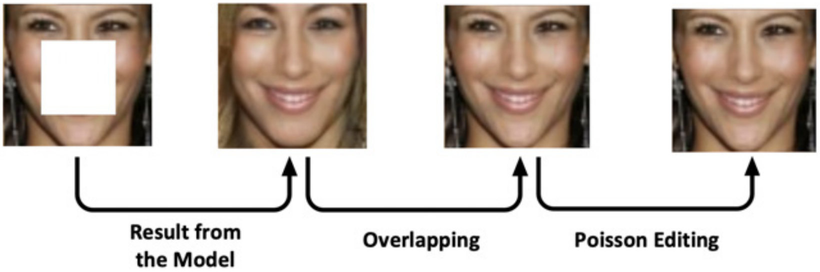




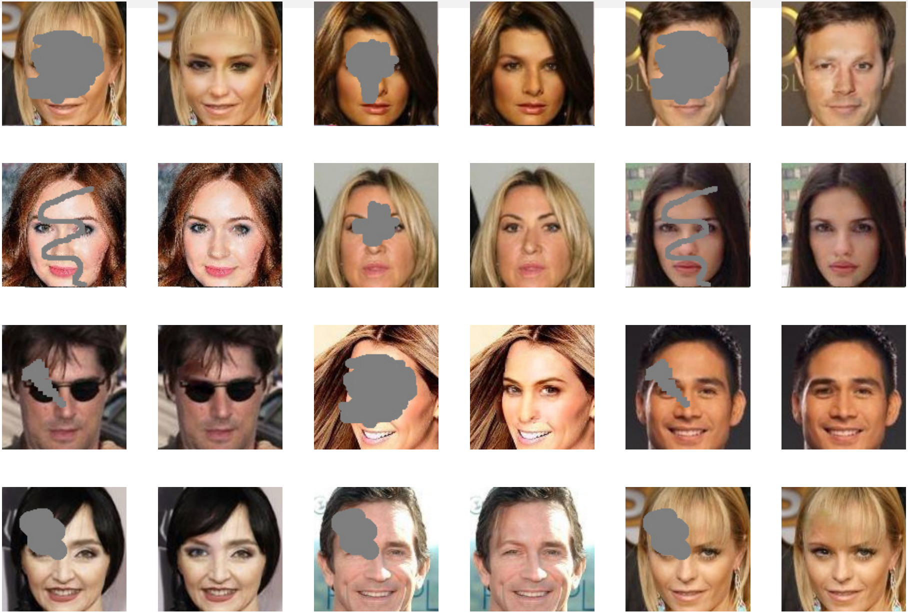
Given a GAN model trained on real images, we iteratively update z to find the closest mapping on the latent image manifold, based on the designed loss function.

Manifold traversing when iteratively update z using back-propagation. $z^{(0)}$ is random initialized; $z^{(k)}$ denotes the result in k-th iteration; and $\hat{z}$ denotes the final solution before the Poisson editing step is applied.

Optional:

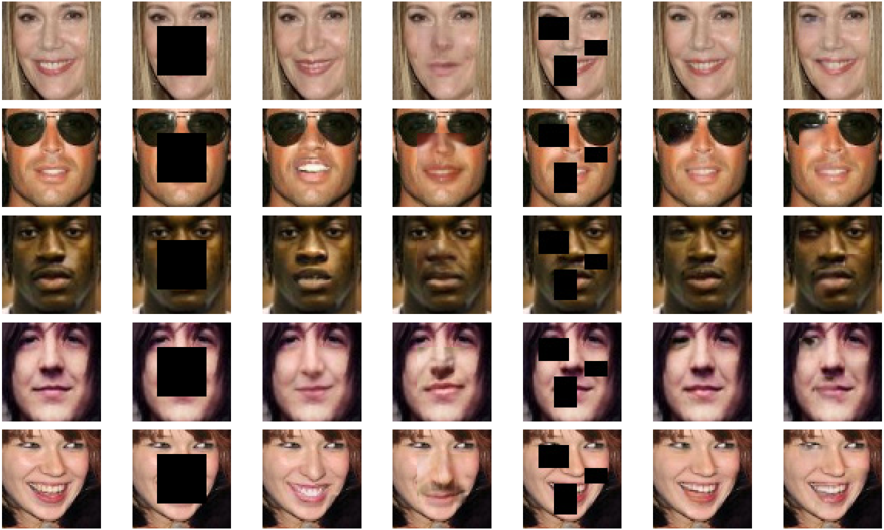


Experimental results



Masked	Ours	Masked	Ours	Masked	Ours

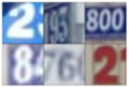
Experimental results



Original Masked Ours Yeh 2017 Masked Ours Yeh 2017

Street View House Numbers (SVHN)

Quantitative results



	CelebA dataset			SVHN dataset		
Loss formulation	MSE	PSNR	SSIM	MSE	PSNR	SSIM
(Yeh et al., 2017)	872.8672	18.7213	0.9071	1535.8693	16.2673	0.4925
(Yeh et al., 2017) adding gradient loss with $\alpha = 0.1$, $\beta = 0.9$ and $\eta = 1.0$	832.9295	18.9247	0.9087	1566.8592	16.1805	0.4775
(Yeh et al., 2017) adding gradient loss with $\alpha = 0.5$, $\beta = 0.5$ and $\eta = 1.0$	862.9393	18.7710	0.9117	1635.2378	15.9950	0.4931
(Yeh et al., 2017) adding gradient loss with $\alpha = 0.1$, $\beta = 0.9$ and $\eta = 0.5$	794.3374	19.1308	0.9130	1472.6770	16.4438	0.5041
(Yeh et al., 2017) adding gradient loss with $\alpha = 0.5$, $\beta = 0.5$ and $\eta = 0.5$	876.9104	18.7013	0.9063	1587.2998	16.1242	0.4818
Our proposed loss with $\alpha = 0.1$, $\beta = 0.9$ and $\eta = 1.0$	855.3476	18.8094	0.9158	631.0078	20.1305	**0.8169**
Our proposed loss with $\alpha = 0.5$, $\beta = 0.5$ and $\eta = 1.0$	**785.2562**	**19.1807**	**0.9196**	743.8718	19.4158	0.8030
Our proposed loss with $\alpha = 0.1$, $\beta = 0.9$ and $\eta = 0.5$	862.4890	18.7733	0.9135	**622.9391**	**20.1863**	0.8005
Our proposed loss with $\alpha = 0.5$, $\beta = 0.5$ and $\eta = 0.5$	833.9951	18.9192	0.9146	703.8026	19.6563	0.8000

	CelebA dataset			SVHN dataset		
Method	MSE	PSNR	SSIM	MSE	PSNR	SSIM
(Yeh et al., 2017)	622.1092	20.1921	0.9087	1531.4601	16.2797	0.4791
(Yeh et al., 2017) adding gradient loss with $\alpha = 0.1$, $\beta = 0.9$ and $\eta = 1.0$	584.3051	20.4644	0.9067	1413.7107	16.6272	0.4875
(Yeh et al., 2017) adding gradient loss with $\alpha = 0.5$, $\beta = 0.5$ and $\eta = 1.0$	600.9579	20.3424	0.9080	1427.5251	16.5850	0.4889
(Yeh et al., 2017) adding gradient loss with $\alpha = 0.1$, $\beta = 0.9$ and $\eta = 0.5$	580.8126	20.4904	0.9115	1446.3560	16.5281	0.5120
(Yeh et al., 2017) adding gradient loss with $\alpha = 0.5$, $\beta = 0.5$ and $\eta = 0.5$	563.4620	20.6222	0.9103	1329.8546	16.8928	0.4974
Our proposed loss with $\alpha = 0.1$, $\beta = 0.9$ and $\eta = 1.0$	424.7942	21.8490	0.9281	168.9121	25.8542	0.8960
Our proposed loss with $\alpha = 0.5$, $\beta = 0.5$ and $\eta = 1.0$	380.4035	22.3284	0.9314	221.7906	24.6714	**0.9018**
Our proposed loss with $\alpha = 0.1$, $\beta = 0.9$ and $\eta = 0.5$	**321.3023**	**23.0617**	**0.9341**	**154.5582**	**26.2399**	0.8969
Our proposed loss with $\alpha = 0.5$, $\beta = 0.5$ and $\eta = 0.5$	411.8664	21.9832	0.9292	171.7974	25.7806	0.8939

Thank you!

References I

[ISSI16] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa, *Let there be color!: joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification*, ACM Transactions on Graphics (TOG) **35** (2016), no. 4, 110.

[IZZE17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros, *Image-to-image translation with conditional adversarial networks*, Proceedings of the IEEE conference on computer vision and pattern recognition, 2017, pp. 1125–1134.

[LMS16] Gustav Larsson, Michael Maire, and Gregory Shakhnarovich, *Learning representations for automatic colorization*, European Conference on Computer Vision, Springer, 2016, pp. 577–593.

[SCH20] Jheng-Wei Su, Hung-Kuo Chu, and Jia-Bin Huang, *Instance-aware image colorization*, Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 7968–7977.

[VRB20] Patricia Vitoria, Lara Raad, and Coloma Ballester, *Chromagan: Adversarial picture colorization with semantic class distribution*, The IEEE Winter Conference on Applications of Computer Vision (WACV), March 2020.

References II

[ZIE16] Richard Zhang, Phillip Isola, and Alexei A Efros, *Colorful image colorization*, European conference on computer vision, Springer, 2016, pp. 649–666.