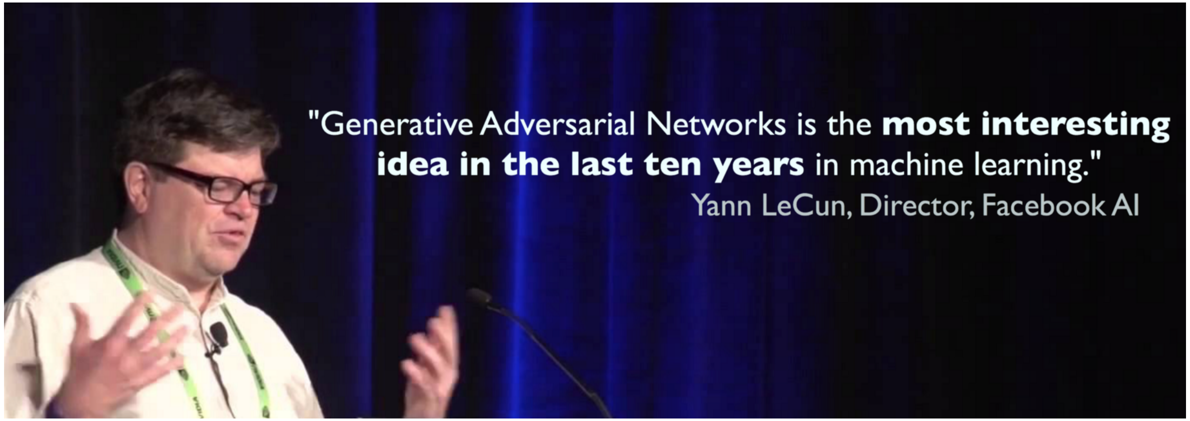
III. Physikalisches Institut A

RWTH AACHEN UNIVERSITY

Generative Adversarial Networks Advanced Techniques

Martin Erdmann, **Jonas Glombitza**



ErUM-Data IDT

SPONSORED BY THE Federal Ministry of Education and Research

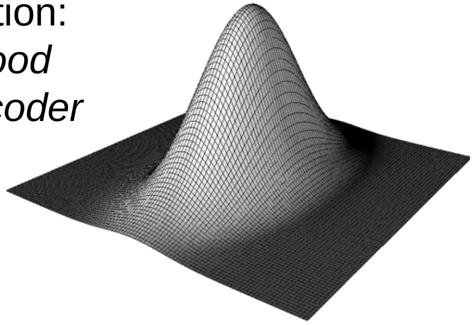
Generative Models

Approximate data distribution P_r with another distribution P_θ

θ = distribution parameters

P_r

Learn prior distribution:
Maximizing Likelihood
Variational Autoencoder



Learn to generate samples following P_θ
Without using directly P_θ
Train a generator only → GANs

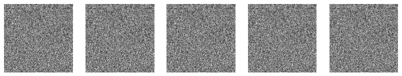
Train a Generator

- Objective: learn to generate new samples following P_θ

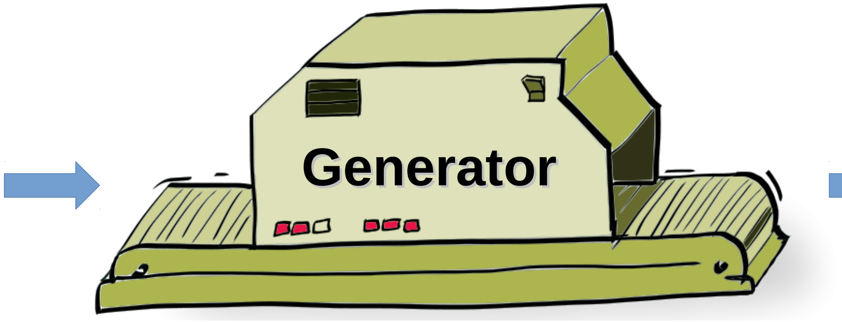
- Learn a function that transform a distribution $p(\mathbf{z})$ into P_θ using a generator G_θ
  $\mathbf{z} \in Z \rightarrow$ latent space

- Generator G_θ is implemented as neural network with weights θ

Noise input



$\mathbf{z} \sim p(\mathbf{z})$



Samples following P_θ

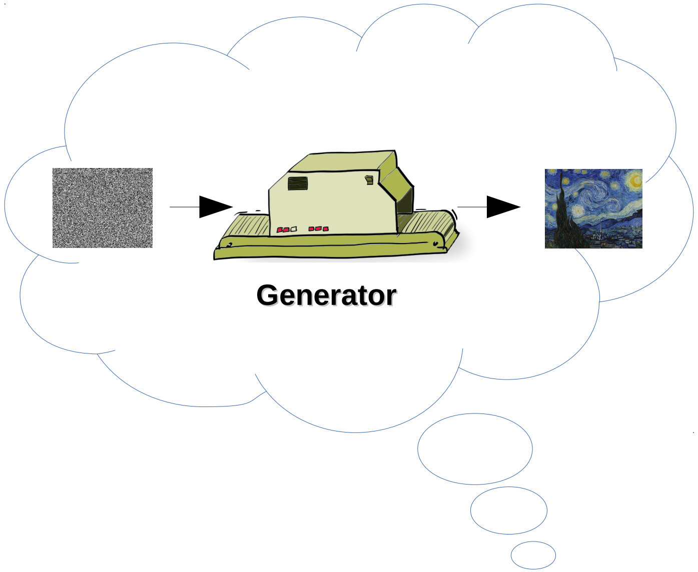
Generative Adversarial Networks

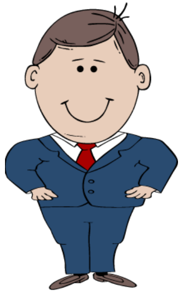
- Hard to formulate a supervised training loss

- Use **unsupervised training** to train the generator
  - Objective: $P_\theta \approx P_r$
  - Measure: given by **second neural network**

→ Generated samples of generator should be similar to real samples after training
  - without reproducing training data

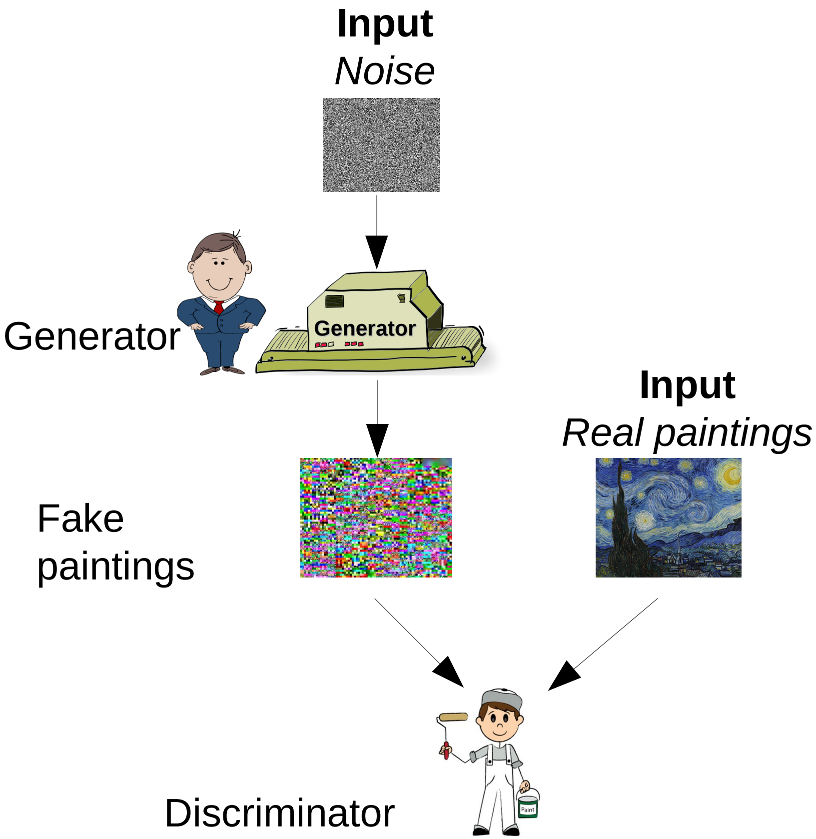
→ **Adversarial approach:**
Train 2 networks adversarial (against each other)



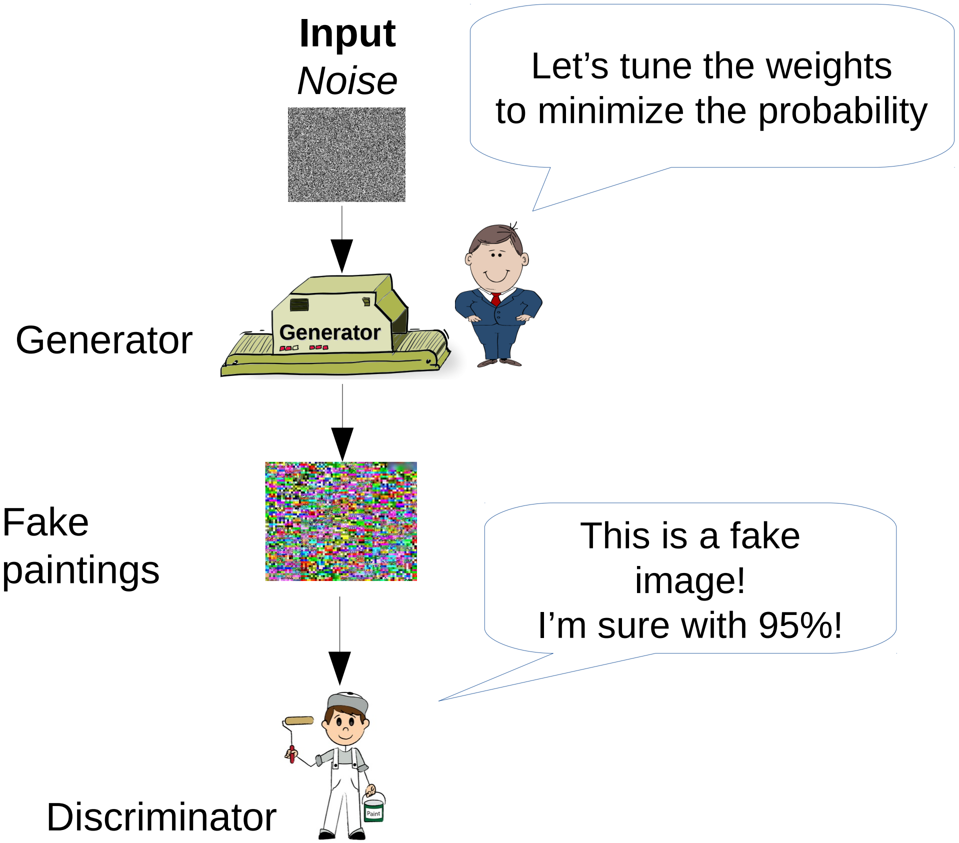


Art forger
Wants to create some fake images

Train Discriminator



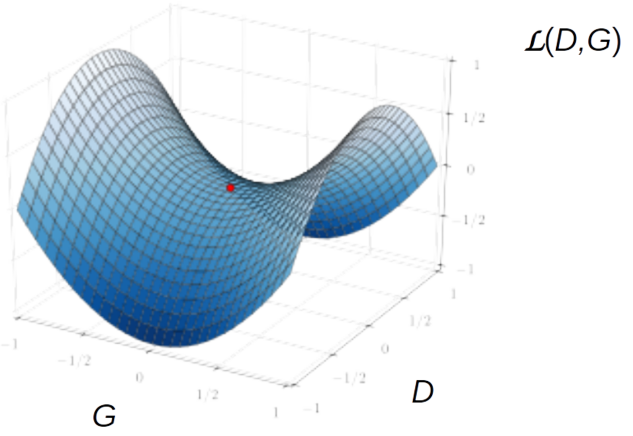
Train Generator



Adversarial Training

$$\min_G \max_D L(D,G) = \mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[log(D(\mathbf{x}))] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[log(1 - D(G(\mathbf{z})))]$$

Training 2 networks at the same time is challenging

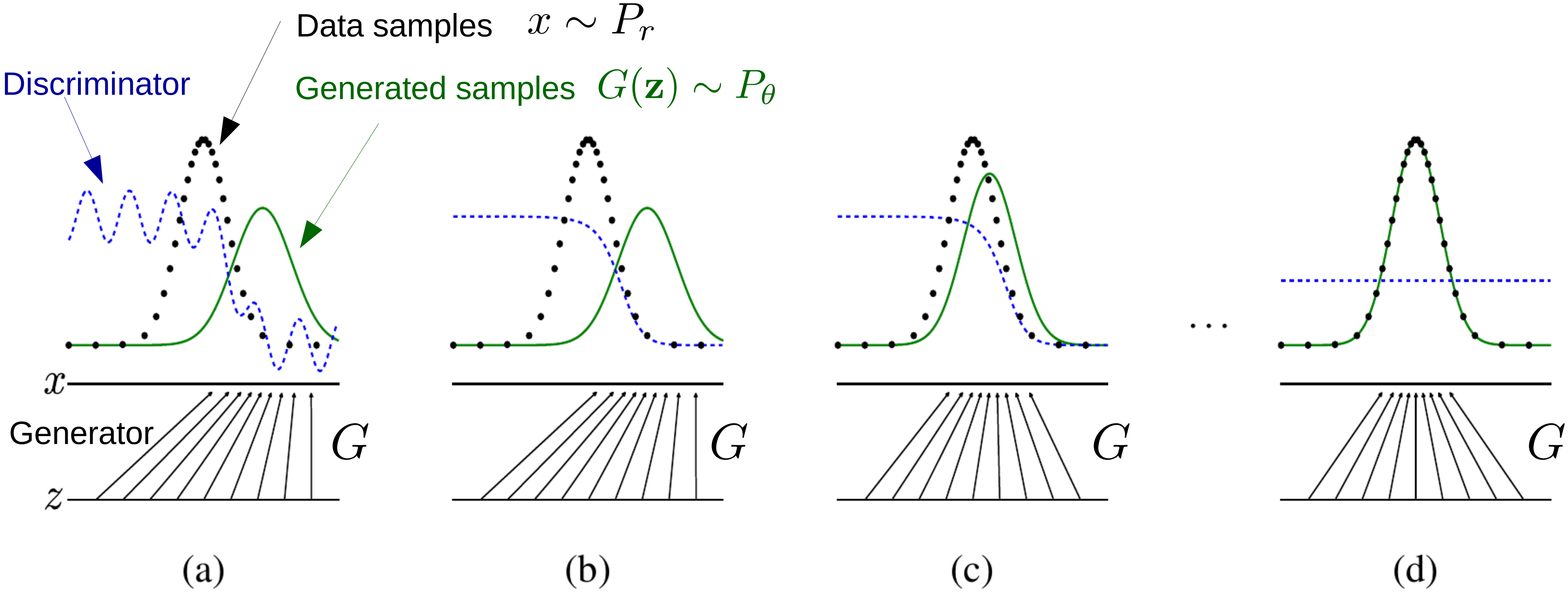


- Train generator and discriminator iteratively
  - Min/Max game
  - Sum of both players is zero

- Finding Nash equilibrium is hard
  - Discriminator and generator need to have same quality

- Minimize Jensen-Shannon divergence (assume optimal discriminator)

Optimal Evolution of GAN Training



Goodfellow et al. - arXiv:1406.2661

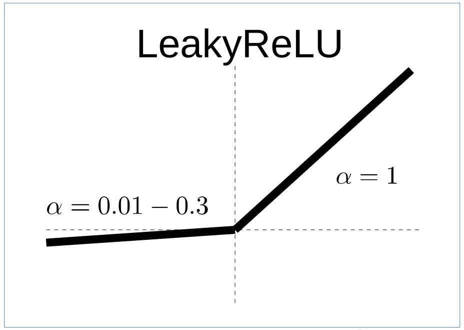
Gradient of discriminator guides generator
→ G generates samples which are more likely identified as data

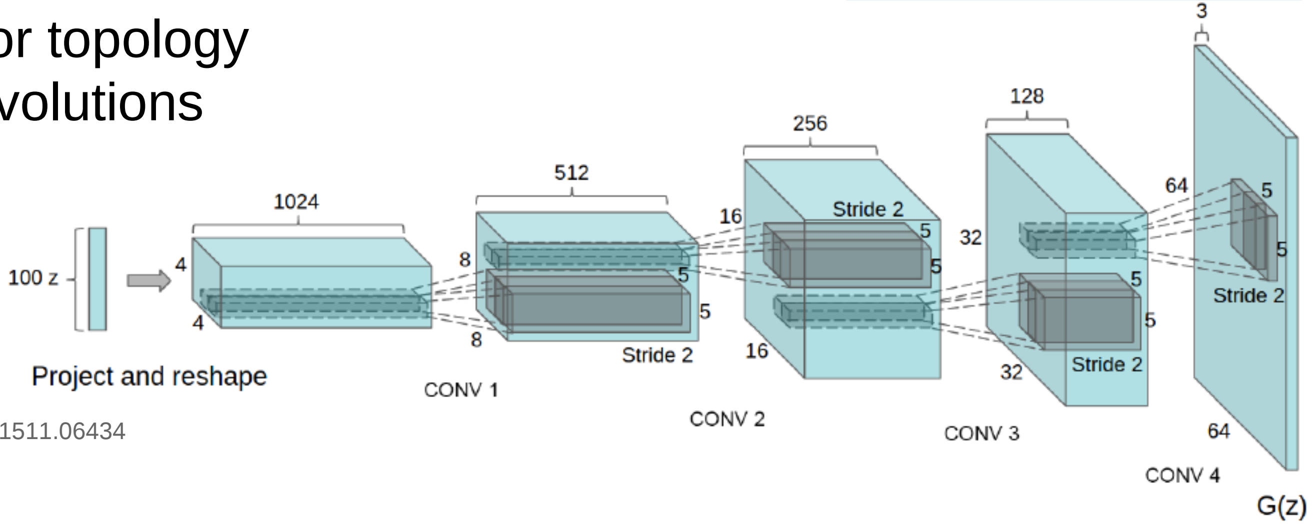
Epochs ►

Deep Convolutional GANs (DCGANs)

- DCGANs (Deep Convolutional GANs) show improved stability

- Use **deep** convolutional generator and discriminator:
  I. Use batch normalization
  II. Remove fully connected hidden layers
  III. Use ReLU in the generator
  IV. Use LeakyReLU in the discriminator
  V. Use special generator topology
    - Use transposed convolutions





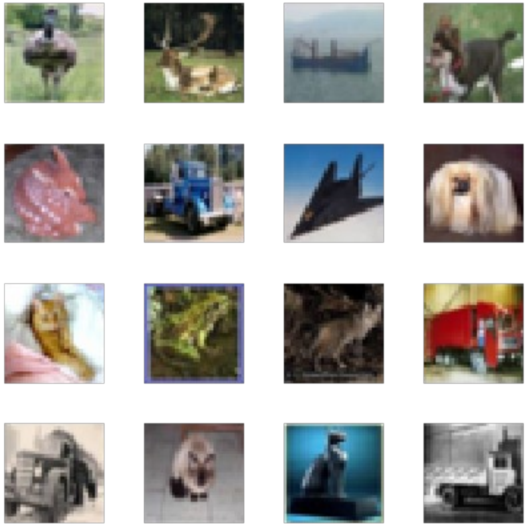
A. Radford, L. Metz, S. Chintala - https://arxiv.org/abs/1511.06434

HANDS ON I

- Train GAN on **CIFAR10** data set
  - Size: 32 x 32 x 3 (RGB)
  - Holding 10 classes

P_r

Generate new samples

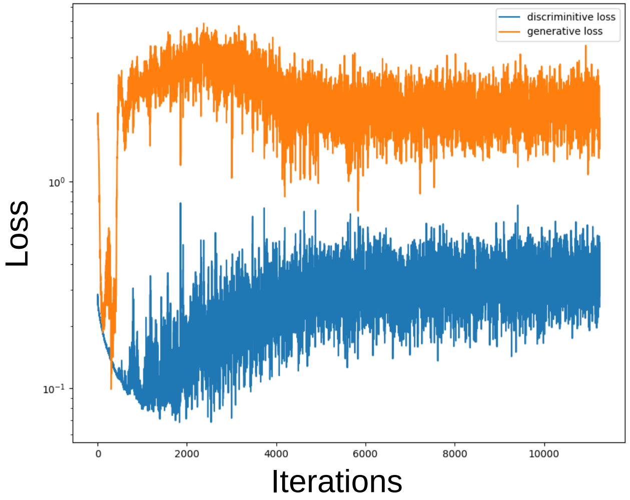


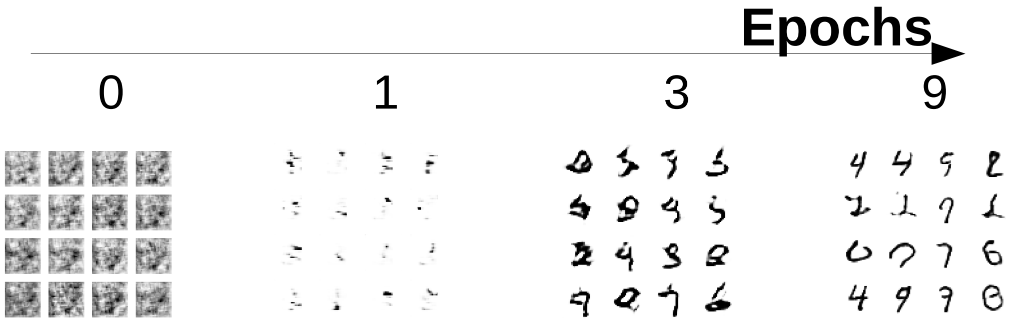
Latest developments & advanced techniques

- **Understanding GAN training**
  - Training issues
- Wasserstein GANs
- Spectral normalization

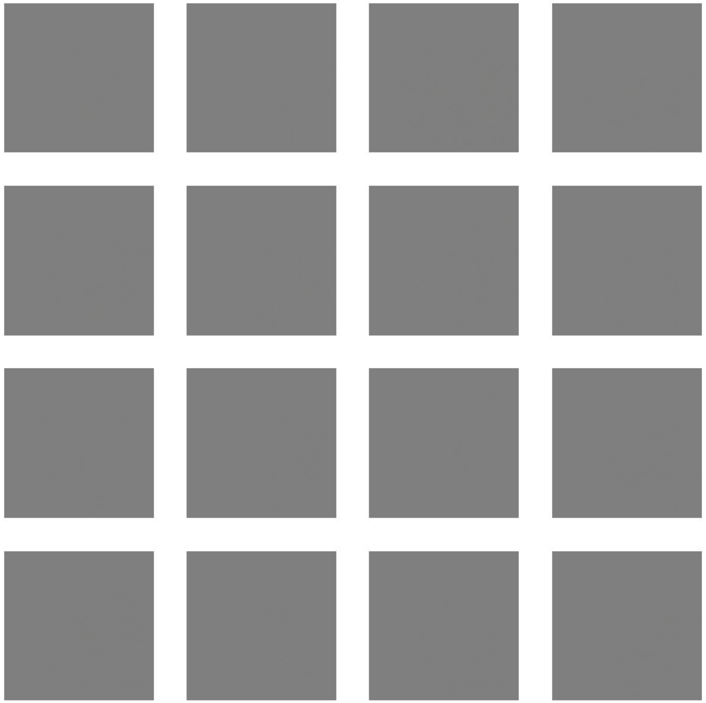
Results

MNIST





CIFAR 10



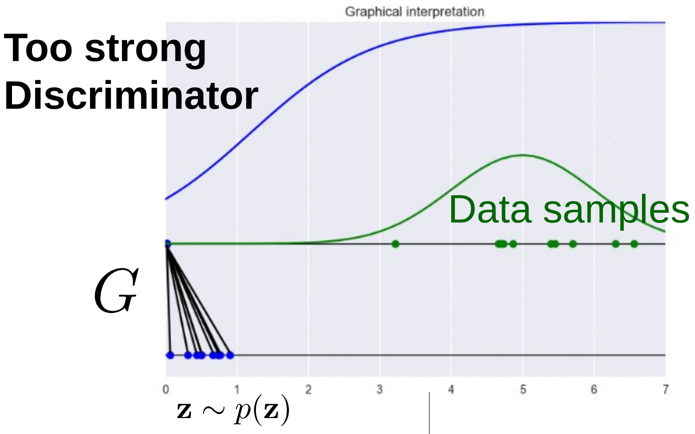
Interpreting the Adversarial Loss

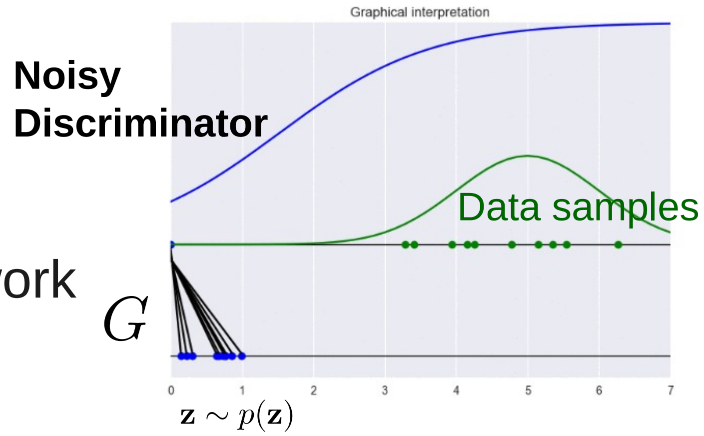
Emanuele Sansone: Tutorial on Generative Adversarial Networks (GANs) - GitHub

- GANs are hard to train → Nash equilibrium
  - generator $\longleftrightarrow$ discriminator

- Loss is hard to interpret (depends on discriminator)
  - no correlation with image quality

- **Strong discriminator → vanishing gradients**
- Best: generator and discriminator on same scale
  - Inexact noisy training → Rarely converging framework

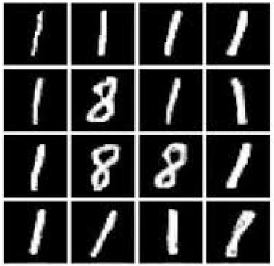




Mode Collapsing - Helvetica Scenario

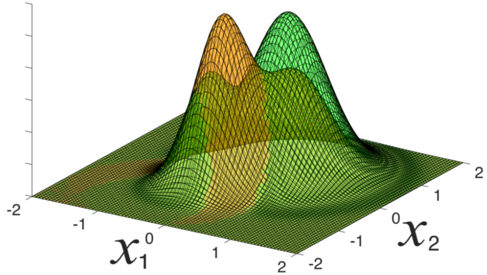
Problem: GANs often suffer from *mode collapsing*

- Many $\mathbf{z} \sim p(\mathbf{z})$ collapse towards restricted space in P_r
  - Generator produce samples of a limited phase space
  - Example: generate only digits 1 and 8



- Discriminator feedback is insensitive to complete phase-space
  - Will focus on point(s) of phase space the generator do not cover
- Discriminator will push generator to this mode → cycling behavior

- **Need different (softer) metric to address these issues!**



Distribution Similarity - Metrics

III. Physikalisches Institut A | RWTH AACHEN UNIVERSITY

- Kullback-Leibler divergence
  - ✗ Not finite, not symmetric

$$\mathcal{D}_{KL}(P_r||P_\theta) = \mathbb{E}_{\mathbf{x}\sim P_r} log\left(\frac{P_r}{P_\theta}\right)$$

- Jensen-Shannon divergence

$$\mathcal{D}_{JS}(P_r||P_\theta) = \mathcal{D}_{KL}(P_r||P_m) + \mathcal{D}_{KL}(P_\theta||P_m) \qquad P_m = \frac{1}{2}(P_r + P_\theta)$$

  - ✔ Symmetric

- Wasserstein distance
  - ✔ Symmetric
  - ✔ Meaningful distance measure for disjoint distributions

For disjoint distributions:

$$\mathcal{D}_{KL}(P_\theta||P_r) = \infty$$
$$\mathcal{D}_{KL}(P_r||P_\theta) = \infty$$
$$\mathcal{D}_{JS}(P_r||P_\theta) = \log(2)$$

In GAN training we are dealing with disjoint distributions!

Wasserstein Distance

- Earth Mover's distance (EMD) provides meaning full feedback for disjoint settings

Ensures smallest cost

Traveling distance

$$\mathcal{D}_W(P_r||P_\theta) = \inf_{\gamma\in\Pi(P_r,P_\theta)} \mathbb{E}_{(x,y)\sim\gamma}[||x-y||]$$

Transportation plans

- Describes **minimal cost** to move distribution P_θ on P_r and vice versa
  - Cost: mass * distance

- Wasserstein distance
  - ✔ Symmetric
  - ✔ Ensures meaningful distance for disjoint distributions

The WGAN Concept

- Use Kantorovich-Rubinstein duality to estimate Wasserstein distance

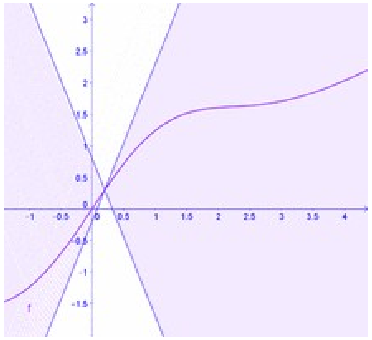
$$\mathcal{D}_W(P_r||P_\theta) = \sup_{f \in Lip_1} \mathbb{E}_{x \sim P_r}[f_w(x)] - \mathbb{E}_{\tilde{x} \sim P_\theta}[f_w(\tilde{x})]$$

Real samples

Generated samples $\tilde{x} = G_\theta(z)$

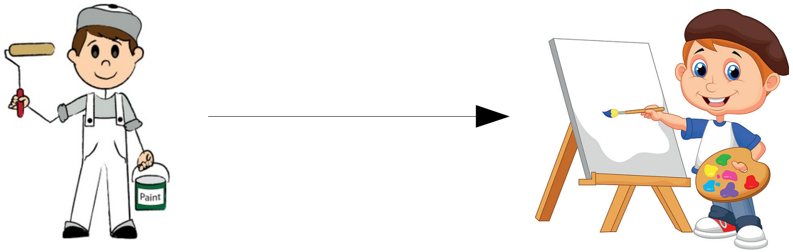
- f_w = neural network (discriminator → critic)
- Neural network carries the Lipschitz continuity constraint
  - Critic network estimate Wasserstein distance between generate and real samples

1-Lipschitz functions



Slope everywhere less equal 1!

Discriminator

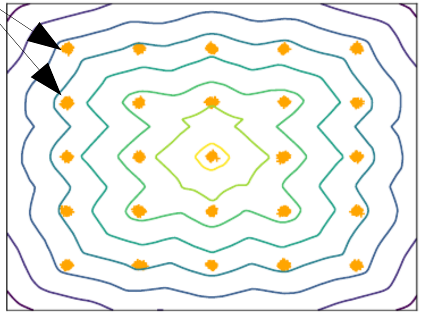


Critic

Gradient Penalty

- Implement Lipschitz constraint
  - Build up space for meaningful discriminator feedback
- Without Lipschitz constrain
  - Critic will not converge → **No Wasserstein!**

Targets



Value surfaces of critic

Extend objective with additional term:

- Penalize gradients being different from 1

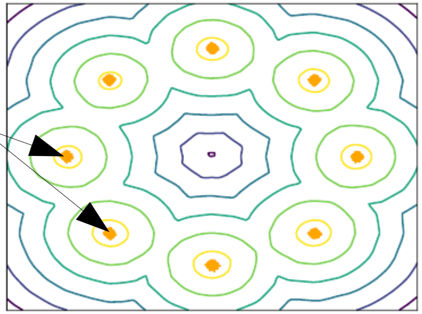
$$\mathcal{L}_{GP} = \lambda\, \mathbb{E}_{\hat{u}\sim P_{\hat{u}}}[(||\nabla_{\hat{u}} f_w(\hat{u})||_2 - 1)^2]$$

hyperparameter

- Sample gradients along line between event mixture $\hat{u}$

$$\hat{u} = \epsilon x + (1-\epsilon)\tilde{x} \qquad 0 \leq \epsilon \leq 1$$

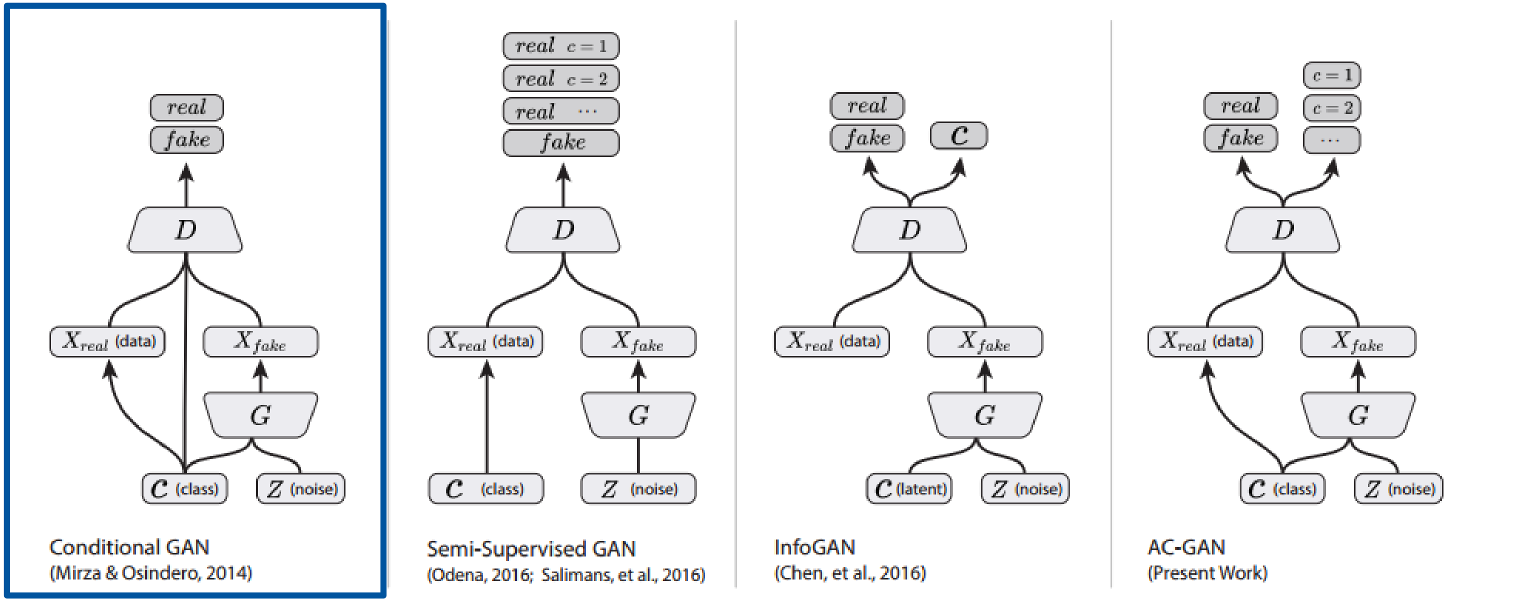
Targets



Value surfaces of critic

Conditioning of GANs

- Constrain generator to learn conditional probability distribution
  - Reduce complexity of latent space, allow for interpretations
- Feed generator and discriminator additional informations (e.g. class labels: dog)
  - Force generated samples show specific characteristics (label dependencies)

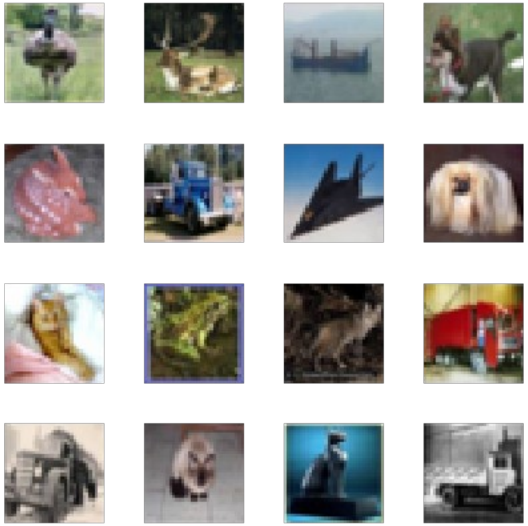


HANDS ON II

- Train conditioned WGAN on **CIFAR10** data set
  - Size: 32 x 32 x 3 (RGB)

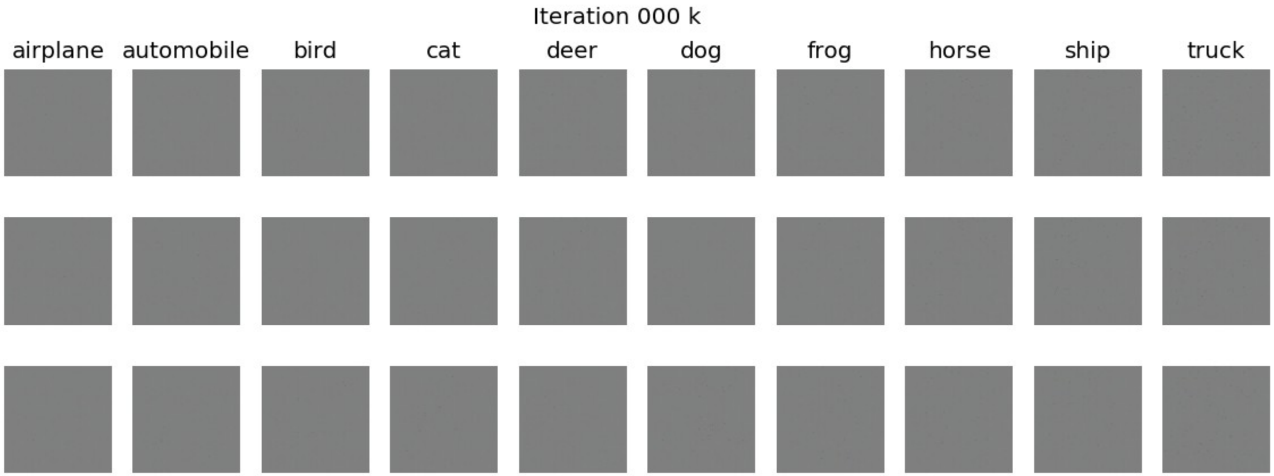
P_r

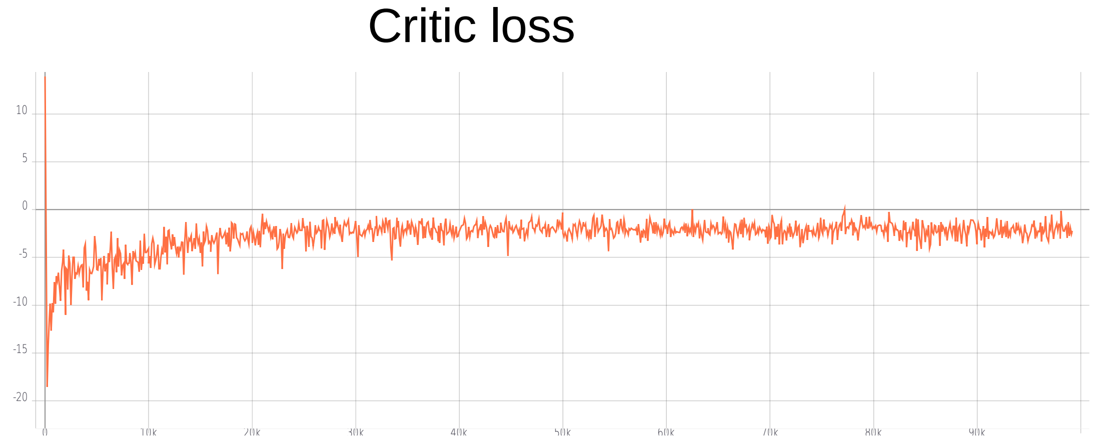
Generate new samples



Results

- WGAN generates images with much better quality
- Critic loss converges
- Loss correlates with images quality





Wasserstein GANs

- Allow stable training of GANs
  - Train critic to convergence
    - Precise feedback for generator
- Prevent mode collapsing
- Provide meaningful loss

Spectral Normalization for GANs

- **Gradient penalty / regularization is most important for training GANs!**
- WGAN-GP is state of the art → Gradient “normalization” (penalty)
  - Also *standard GAN* with gradient penalty performance well!
  - Training can be slow because of many critic iterations

- Adapt Lipschitz constraint using different normalization strategy
  - Normalize weights using *spectral norm* (fast approximation)

- GAN training:
  - Speed up
  - Increased stability (high learning rates, high momentum rates)

Spectral Normalization

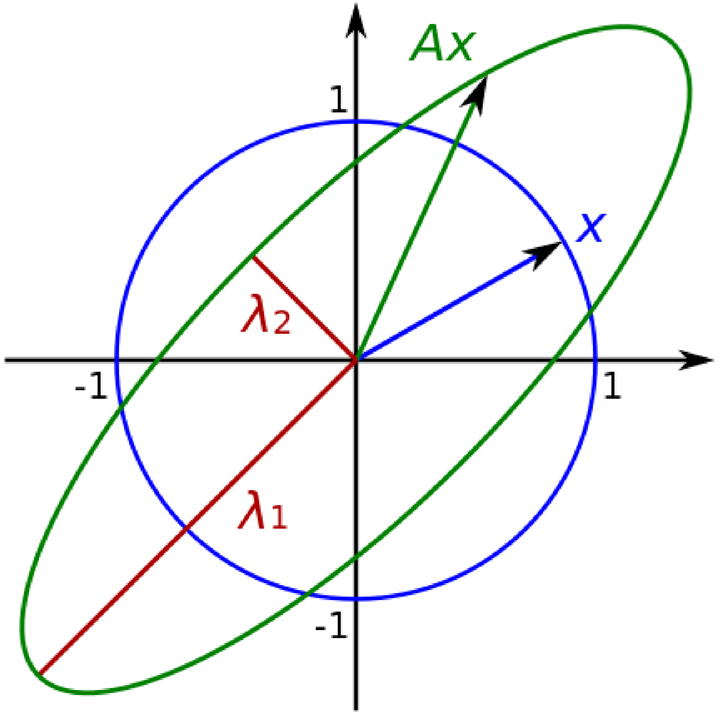
III. Physikalisches Institut A | RWTH AACHEN UNIVERSITY

- Spectral norm: “natürliche Matrixnorm”

$$\|A\|_2 := \max_{x \neq 0} \frac{\|Ax\|_2}{\|x\|_2} = \max_{\|x\|_2 = 1} \|Ax\|_2$$

- Maximum stretch factor of unit vector after multiplication with matrix

- λ_1 = highest singular value (“Singulärwert”) of the matrix



Spectral Normalization for GANs

- $D(x)$ = discriminator
- Adapt WGAN-GP constraint (gradient wrt. x real and fake samples)
  - Use **spectral normalization** in each layer!

- Basic idea:

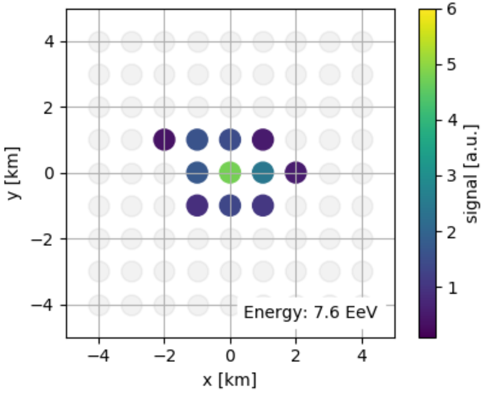
$$||D(x)||_{\mathrm{Lip}} = \sup_x \sigma(\nabla_x D(x)) = \sup_x \sigma(\nabla_x W x) = \sigma(W) \longrightarrow W_{\mathrm{norm}} = \frac{W}{\sigma(W)}$$

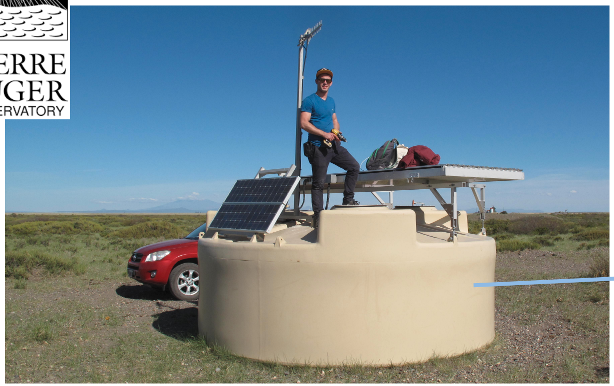
➢ Cover Lipschitz constraint by normalizing the weights

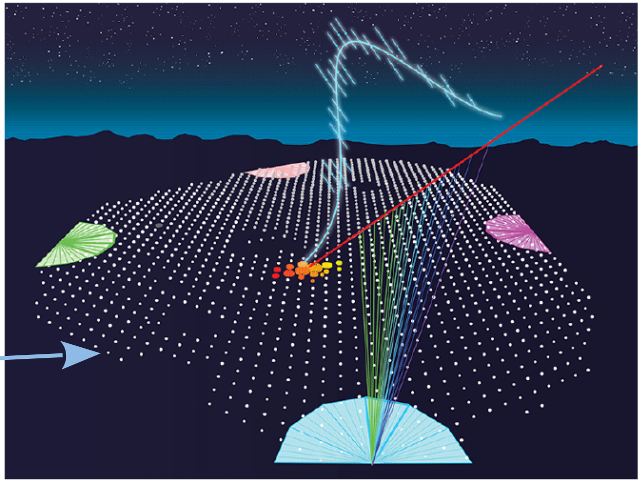
- **Gradient update:**
  - Gradient penalizes updates in direction of highest singular value (in each layer)

Generate Air Shower Footprints

- Measurement of cosmic ray induced air showers
- **Pierre Auger Observatory: Fluorescence (FD)** and **Surface Detector (SD)**
  - FD: Telescopes measure light of excited nitrogen
  - SD: Water Cherenkov stations detect passage of charged particles
  - Simulation: 2D image sequence, Cartesian grid, 1-100 EeV protons



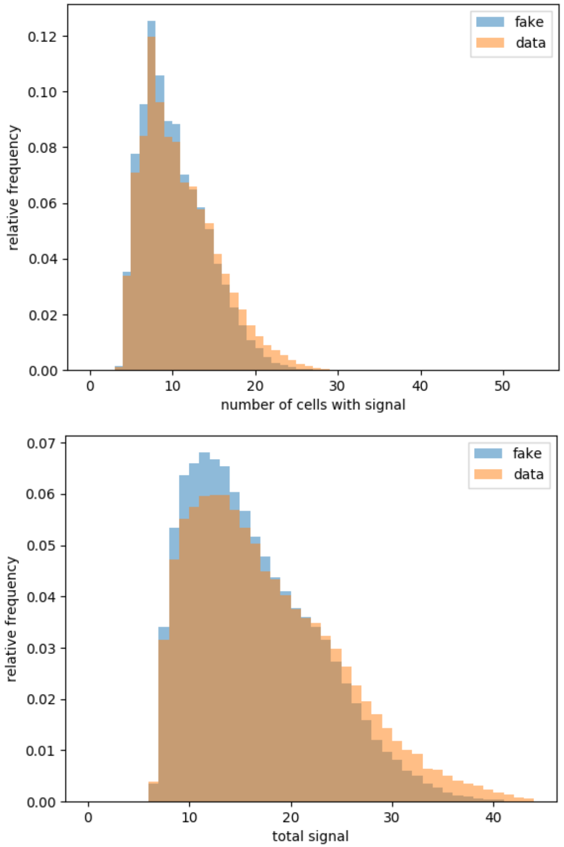




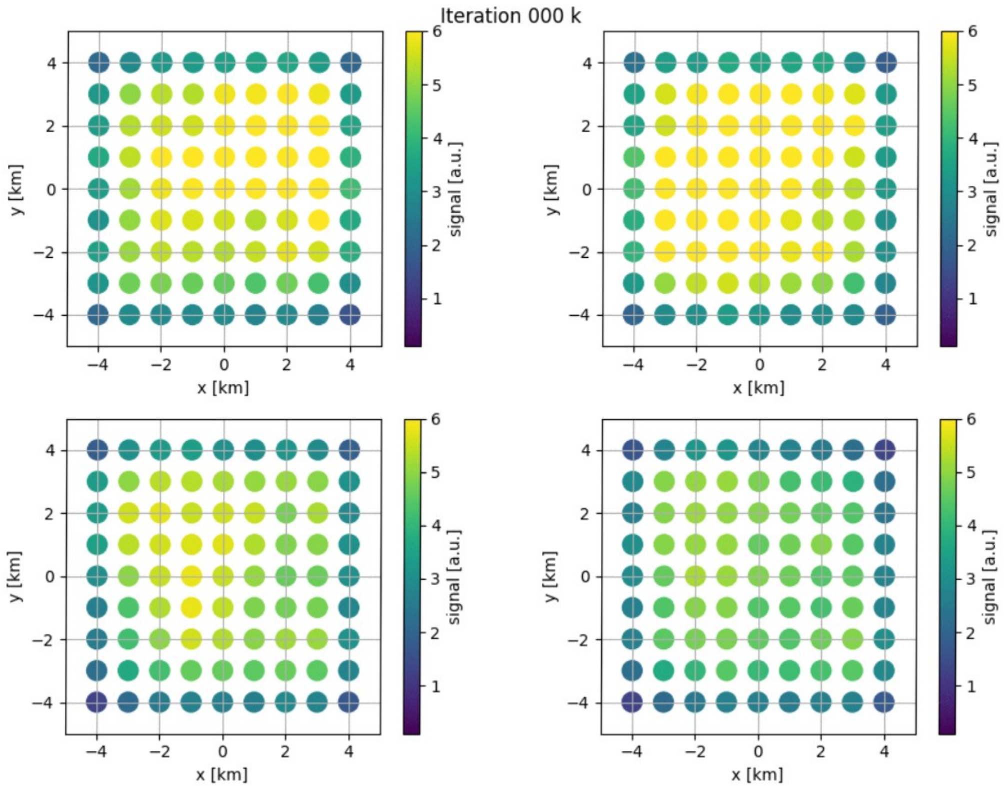
https://physics.aps.org/articles/v9/125

Results

Quick physics cross checks



Generated footprints during training

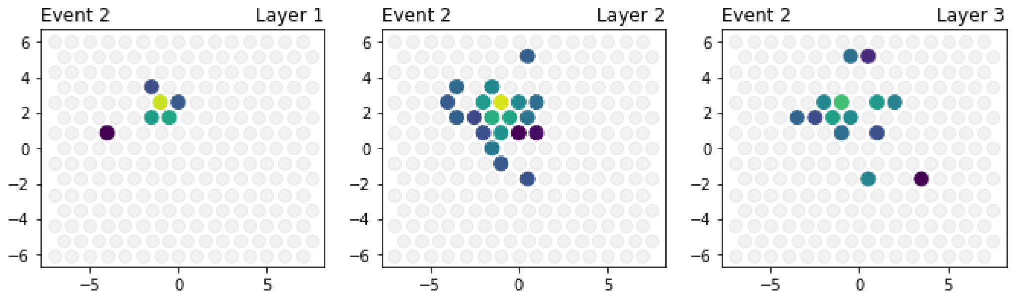


Calorimeter Images

- **Spectral normalization** and **gradient penalty** enforces Lipschitz constrain differently
➢ Combine both techniques
- Further apply spectral normalization in the generator

Generate Calorimeter Images

- 100 GeV electron beam, generated by T. Quast using GEANT4



Outlook

- Model architectures and hyperparameters still need to be tuned for each task
- Tips / Tricks
  - Never use vanilla GANs!
  - Follow DCGAN “guidelines”
  - Preprocess your data
  - Use label conditioning
  - Use deep models

Yang, Chou, Yang - https://arxiv.org/abs/1703.10847

(a) MidiNet model 1

(b) MidiNet model 2

(c) MidiNet model 3

- There is much more going on → **stay tuned**
  - Cycle GANs
  - Progressive growing of GANs

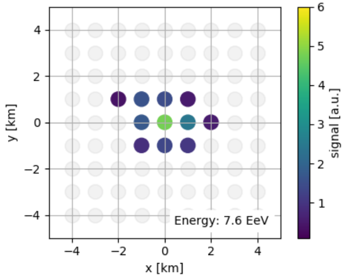
References & Further Reading

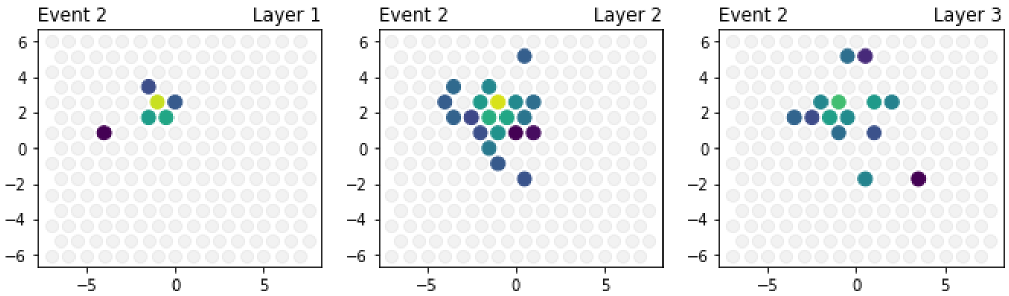
III. Physikalisches Institut A | RWTH AACHEN UNIVERSITY

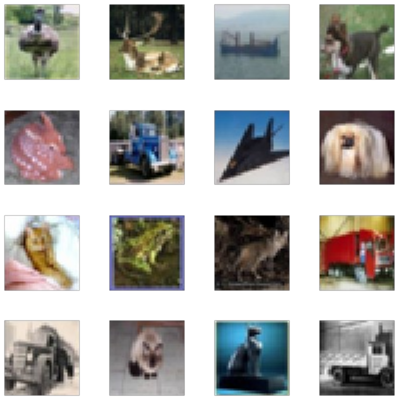
- Goodfellow et al.: Generative Adversarial Networks - https://arxiv.org/abs/1406.2661
- Arjovsky, Chintala, Bottou: Wasserstein GANs - https://arxiv.org/abs/1701.07875
- Gulrajani et al.: WGAN-GP - https://arxiv.org/abs/1704.00028
- Paganini, Oliveira, Nachman: CaloGAN - https://arxiv.org/abs/1712.10321
- Erdmann, Geiger, Glombitza, Schmidt: Refiner - https://arxiv.org/abs/1802.03325
- Emanuele Sansone - https://github.com/emsansone/GAN
- Erdmann, Glombitza, Quast: Calorimeter WGAN - https://arxiv.org/abs/1807.01954
- Karras, Aila, Laine, Lehtinen: ProGAN - https://arxiv.org/abs/1710.10196
- Arjovsky, Bottou - https://arxiv.org/abs/1701.04862
- Miyato, Kataoka, Koyama, Yoshida: SN-GAN - https://arxiv.org/abs/1802.05957
- Brock, Donahue, Simonyan: BigGANs - https://arxiv.org/abs/1809.11096

Backup

Generative Adversarial Networks
Advanced Techniques





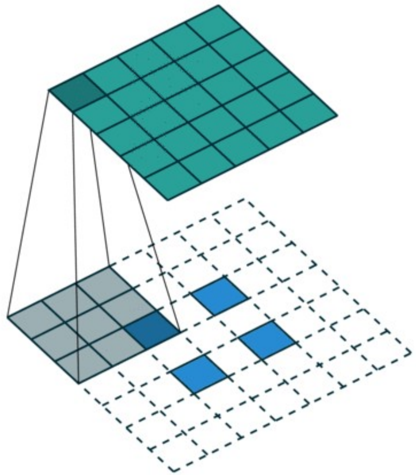


Transposed Convolutions

- Think of process which turn around the convolutional operation

- Convolution
  - Map cluster to 1 pixel
- Transposed convolution
  - Map 1 pixel to a cluster

Example
Transposed convolution, fractionally strided convolution or deconvolution
no padding, stride 2, kernel 3 x 3



Paul-Louis Pröve,
Towards Data Science

Non Saturation GAN (NS-GAN)

III. Physikalisches Institut A | RWTH AACHEN UNIVERSITY

- Use **label switching** to avoid vanishing gradients in discriminator
- Standard loss: *minimize*

$$Loss = \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[log(1 - D(G_\theta(\mathbf{z})))]$$

- But gradients vanish for $D(G_\theta(\mathbf{z})) \to 0$ (good discriminator)
- Replace loss and minimize instead

$$Loss = -\mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[log(D(G_\theta(\mathbf{z})))]$$

  New loss has strange update behavior:
  - No vanishing gradient but instable updates → gradients Cauchy distributed
- Objective looks strange, subtraction of KL and JS
- This KL focus highly on generate fake images, low focus on mode dropping

$$\mathbb{E}_{z\sim p(z)}\left[-\nabla_\theta \log D^*(g_\theta(z))|_{\theta=\theta_0}\right] = \nabla_\theta \left[KL(\mathbb{P}_{g_\theta}\|\mathbb{P}_r) - 2JSD(\mathbb{P}_{g_\theta}\|\mathbb{P}_r)\right]|_{\theta=\theta_0}$$

Distribution Similarity - Metrics

- Kullback-Leibler divergence
  - ✗ Not finite
  - ✗ Not symmetric

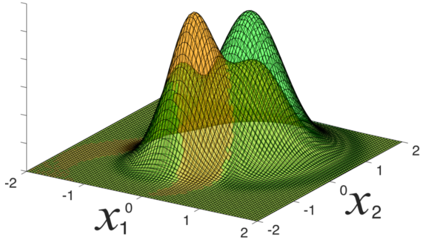
$$\mathcal{D}_{KL}(P_r||P_\theta) = \mathbb{E}_{\mathbf{x}\sim P_r} log\left(\frac{P_r}{P_\theta}\right)$$

- Jensen-Shannon divergence

$$\mathcal{D}_{JS}(P_r||P_\theta) = \mathcal{D}_{KL}(P_r||P_m) + \mathcal{D}_{KL}(P_\theta||P_m) \qquad P_m = \frac{1}{2}(P_r + P_\theta)$$

  - ✔ Symmetric
  - ✗ Fails to provide a meaningful value when two distributions are disjoint

- Wasserstein distance
  - ✔ Symmetric
  - ✔ Ensures meaningful distance for disjoint distributions



Distribution Similarity - Metrics

$$\theta = 0$$

$$\mathcal{D}_{KL} = 0$$

$$\mathcal{D}_{JS} = 0$$

$$\mathcal{D}_{W} = 0$$

$$\theta \neq 0$$

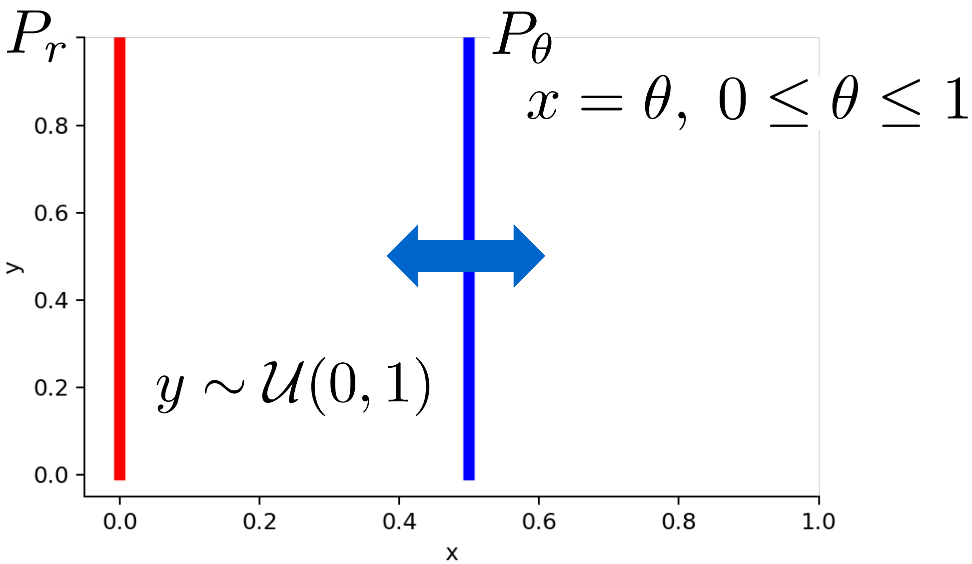
$$\mathcal{D}_{KL} = \sum_{x=\theta, y\sim\mathcal{U}(0,1)} 1 \cdot \log\left(\frac{1}{0}\right) = \infty$$

$$\mathcal{D}_{JS} = \sum_{x=\theta, y\sim\mathcal{U}(0,1)} 0 \cdot \log\left(\frac{1}{1/2}\right) + \sum_{x=\theta, y\sim\mathcal{U}(0,1)} 1 \cdot \log\left(\frac{1}{1/2}\right) = log(2)$$

$$\mathcal{D}_{W} = |\theta|$$

Real distribution

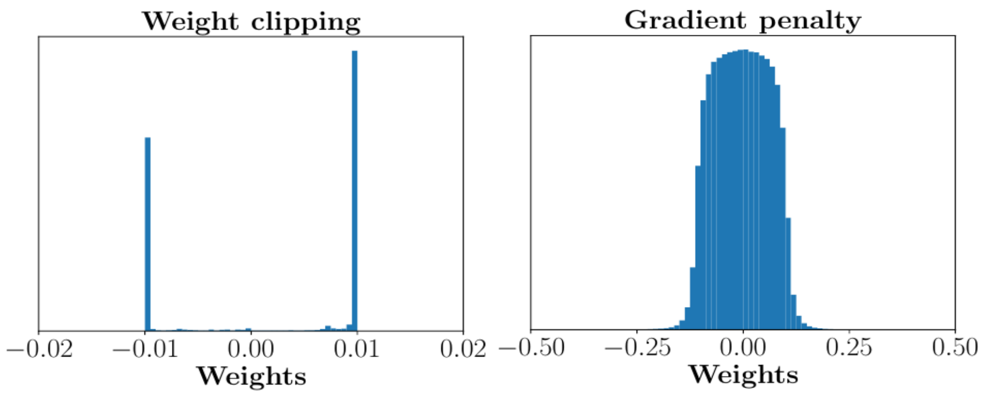
Parametrized approximation



- Only $\mathcal{D}_W$ provides meaningful distance measure even for disjoint distributions!

Weight Clipping vs. WGAN-GP

- Weight Clipping:
  - Constraints the weights to lie on a compact space
  - Clip weights after each gradient update eg. to [-0,001; 0,001]
- Heavily constraints the discriminator
- Gradient Penalty allows for a much more complex approximation



Weight clipping

Gradient Penalty