Karlsruhe Institute of Technology

UNIVERSITÄT BONN

Deep Learning in Air vs Calorimeter Showers

Florian Bernlochner, Pablo Goldenzweig, Jubna Jabbar[1], Marcel Köpke[2], Markus Roth
ErUM Online Collaboration Meeting – April, 2020

[1]jubna.jabbar@kit.edu, [2]marcel.koepke@kit.edu

Update on Simulation of Extensive Air Showers with Deep Neural Networks

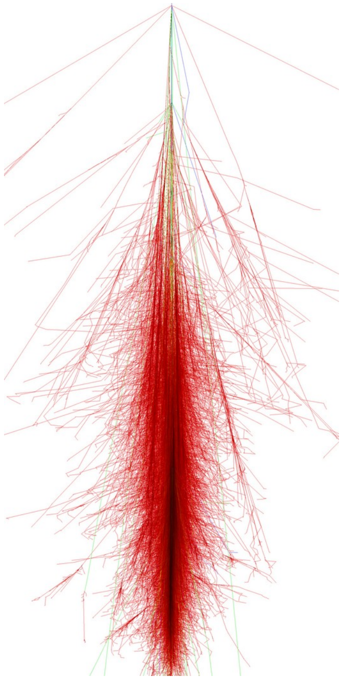
Marcel Köpke

ErUM Online Collaboration Meeting – April, 2020

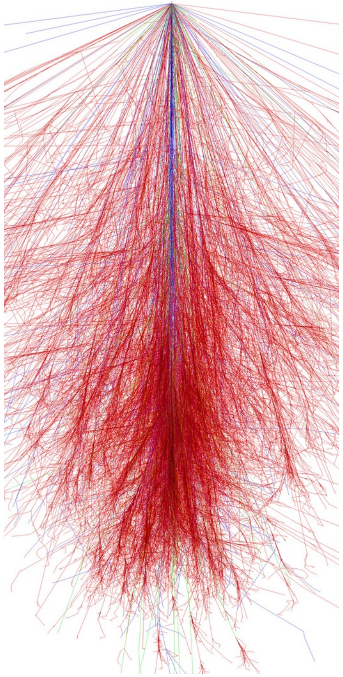
INSTITUTE FOR NUCLEAR PHYSICS (IKP), FACULTY OF PHYSICS
KARLSRUHE INSTITUTE OF TECHNOLOGY (KIT)

Last time: CORSIKA 7 [1]

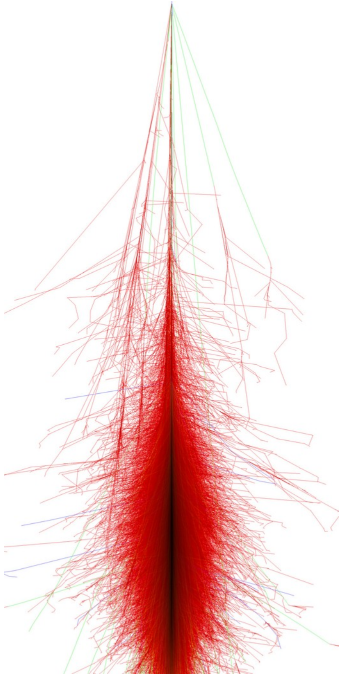
- Extensive air shower Monte Carlo simulation framework
- Different types of interaction models (EPOS-LHC, QGSJET, SIBYLL, ...)



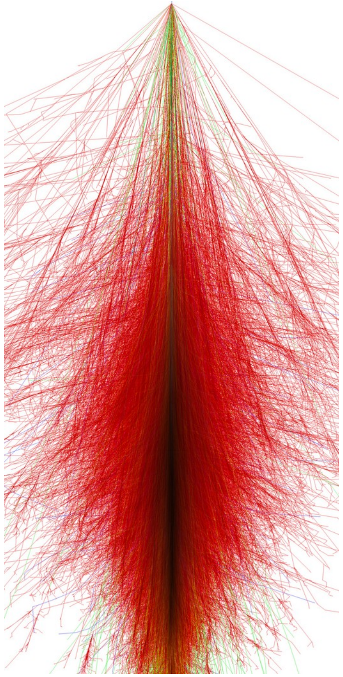
1 TeV Proton



1 TeV Iron



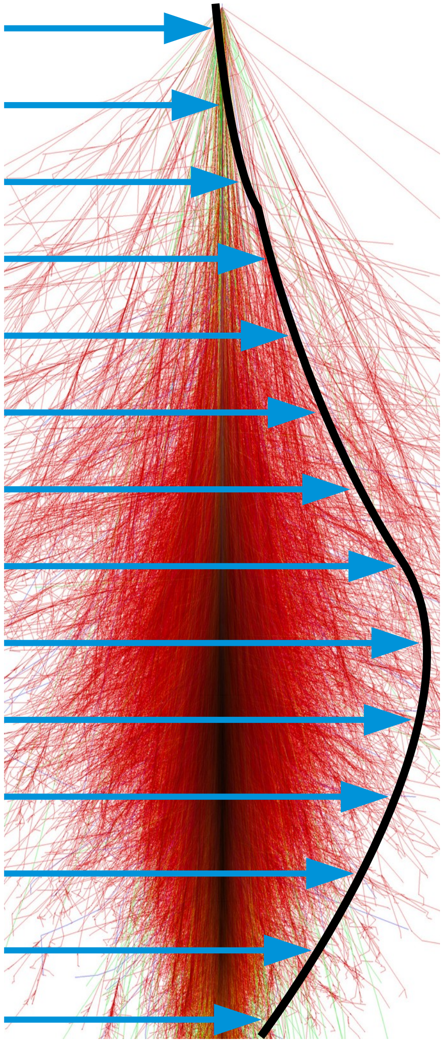
10 TeV Proton



10 TeV Iron

First Test (CONEX)

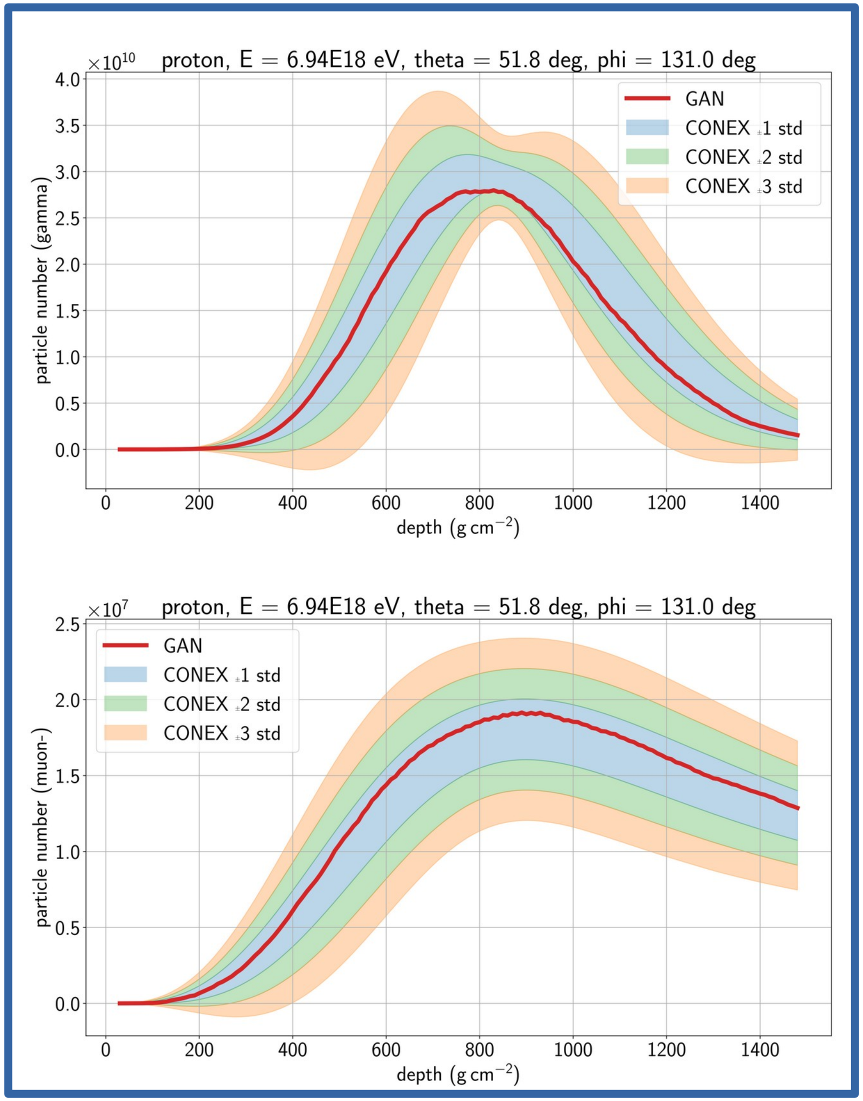
- CONEX: Hybrid Extenisve Air Shower Simulation
  - first: Monte Carlo until energy threshold (3D)
  - then: cascade equation solver (1D)
  - provides longitudinal profile only
  - runtime: seconds – minutes



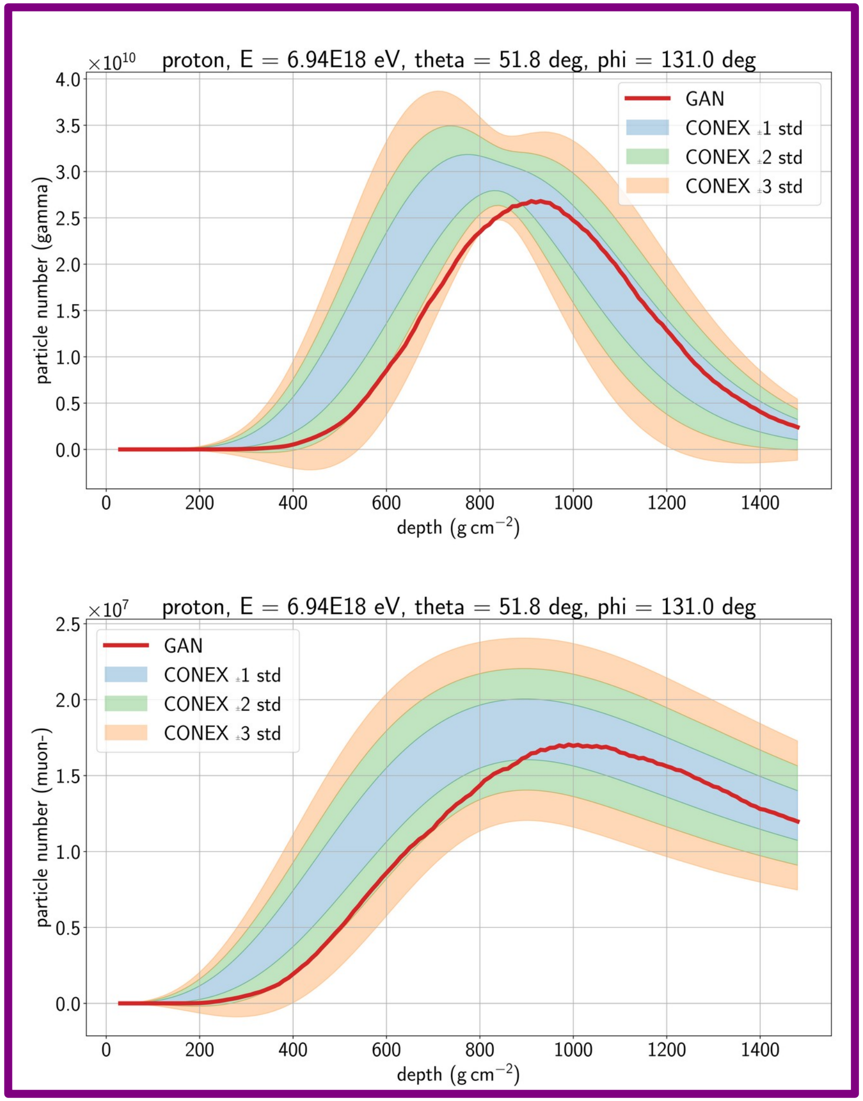
- Configuration:
  - E = 1E17 ... 1E19 eV
  - Zenith = 0 ... 65 deg
  - Azimuth = -180 ... 180 deg

- Generated ~ 300k datapoints

CONEX vs. GAN



×10^10
proton, E = 6.94E18 eV, theta = 51.8 deg, phi = 131.0 deg
GAN
CONEX ±1 std
CONEX ±2 std
CONEX ±3 std
particle number (gamma)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0
200
400
600
800
1000
1200
1400
depth (g cm^-2)
×10^7
proton, E = 6.94E18 eV, theta = 51.8 deg, phi = 131.0 deg
GAN
CONEX ±1 std
CONEX ±2 std
CONEX ±3 std
particle number (muon-)
0.0
0.5
1.0
1.5
2.0
2.5
0
200
400
600
800
1000
1200
1400
depth (g cm^-2)

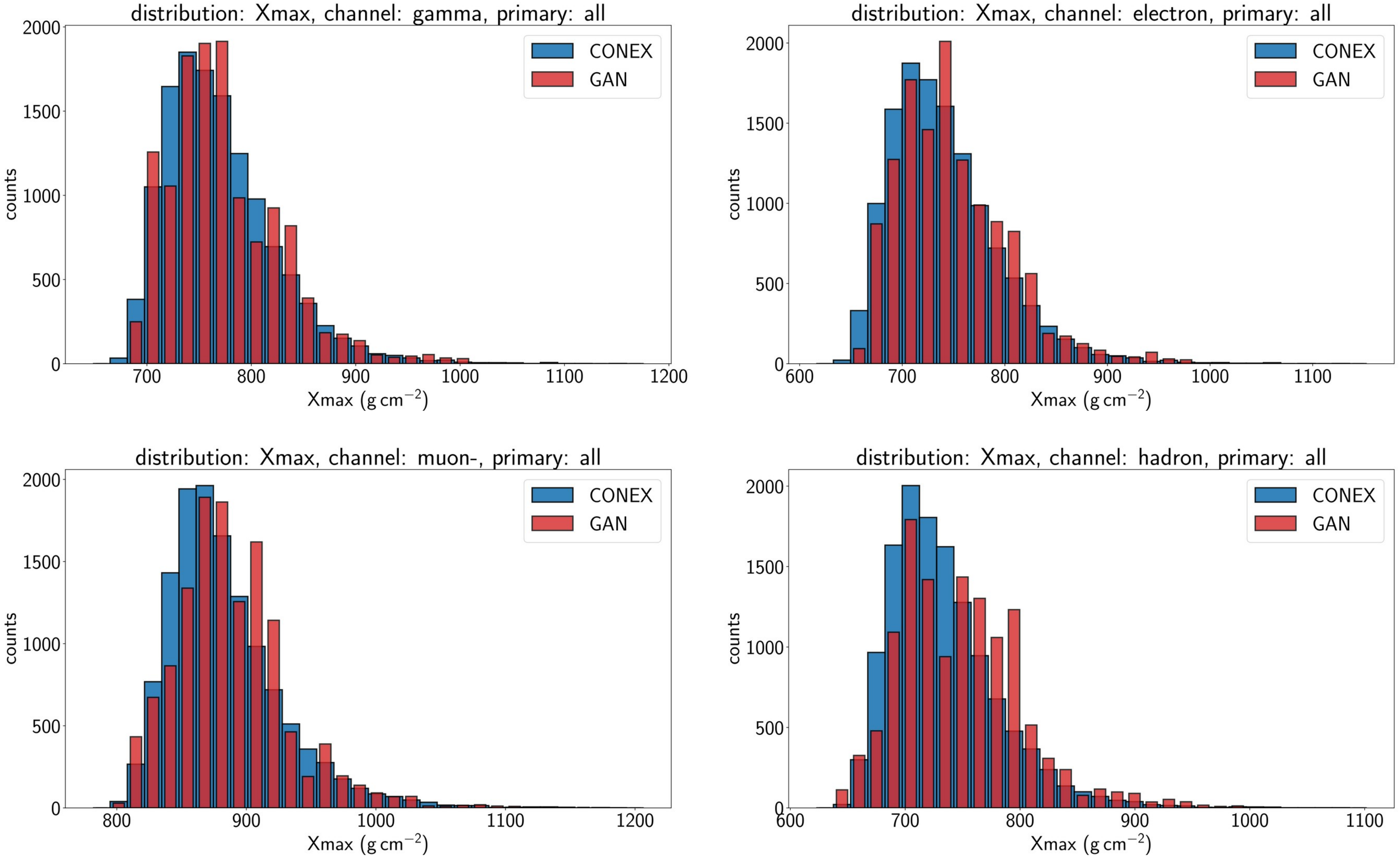


×10^10
proton, E = 6.94E18 eV, theta = 51.8 deg, phi = 131.0 deg
GAN
CONEX ±1 std
CONEX ±2 std
CONEX ±3 std
particle number (gamma)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
0
200
400
600
800
1000
1200
1400
depth (g cm^-2)
×10^7
proton, E = 6.94E18 eV, theta = 51.8 deg, phi = 131.0 deg
GAN
CONEX ±1 std
CONEX ±2 std
CONEX ±3 std
particle number (muon-)
0.0
0.5
1.0
1.5
2.0
2.5
0
200
400
600
800
1000
1200
1400
depth (g cm^-2)

What‘s new

- TensorFlow 1 → TensorFlow 2
- Xmax distribution
- New dataset
- Implementation of new architecture (ongoing)

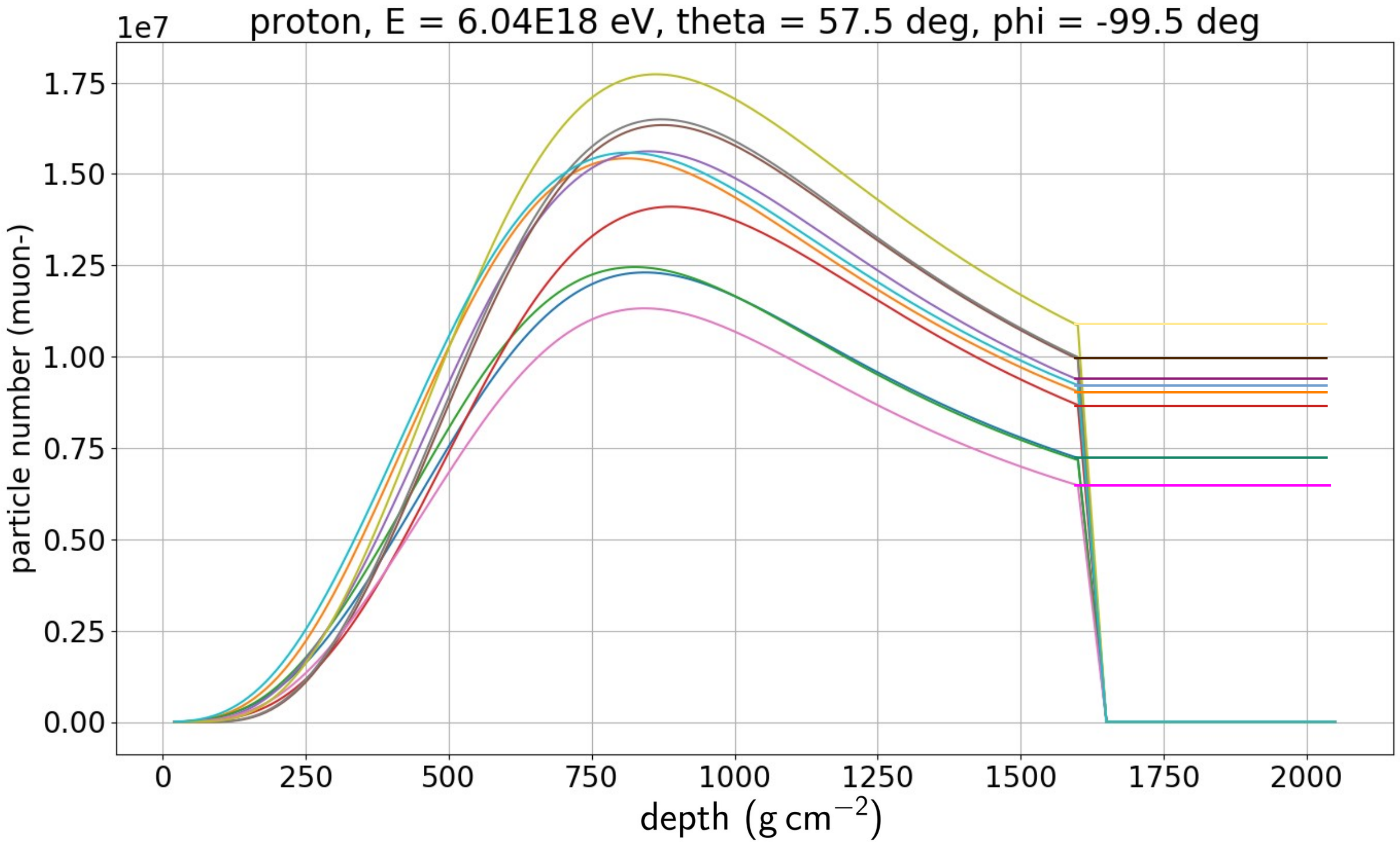
Xmax Distribution (E > 5E18 eV, theta > 35 deg)



New Dataset

- Why? → Bad low energy performance
- Oversample low energies: log-uniform
- Traning: 1.6 million datapoints (200 showers per label set, ~60 GB)
  Test: 2x 500k datapoints (1 + 10 showers per label set, ~ 20 GB)
- Needs memory mapping → tf.data API
  - tf.data.Dataset.from_generator(...)
  - np.load(... , mmap_mode=“r“)
  - dataset.cache(filepath)
- iteration with ~400 MB/s from SSD

New Architecture



New Architecture (Plans)

- Ensemble of Generators + Discriminators

- Old Model
  - Mixture of Dense- and (Transpose)Convolution-Layers
- DenseNet [2]
  - Full Connectiviy
- StyleGAN [3]
  - Noise injection at different stages
- InfoGAN [4]
  - Optimize mutual information of noise and generated data

Summary

- Xmax Distribution looks OK (still problems per primary)
- Many technical improvements
  - New dataset to cope with low energy behaviour
  - Memory mapping for large dataset with standardized API
  - Full TensorFlow 2 implementation
  - Architecture: Masking + Ensemble (ongoing)

Karlsruhe Institute of Technology

UPDATES ON FAST SIMULATION OF BELLE II ECL

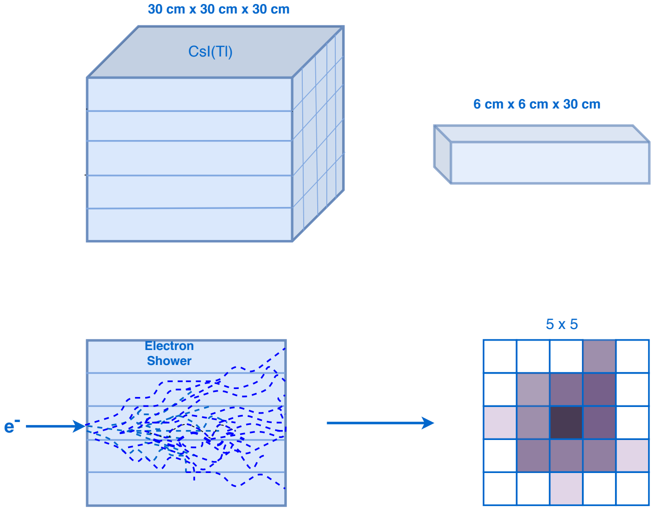
02.04.2020

Jubna Irakkathil Jabbar | IETP

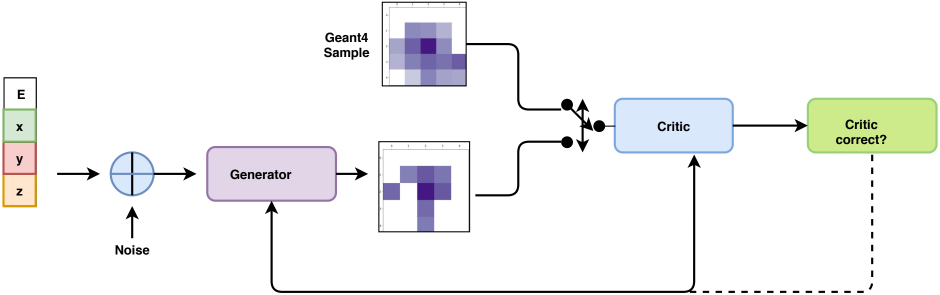
MOTIVATION

- Simulation of particle showers in ECL is a computationally expensive and time consuming process.
- The fast simulation is studied using a configuration of 5x5 CsI(Tl) crystals, as in the Belle II ECL.
- Electrons of energies 0.5 GeV, 1 GeV, 1.5 GeV, 2.5 GeV are used for training and testing.
- Electrons of energy 2.0 GeV are used for interpolation.

PARTICLE SHOWER SIMULATION

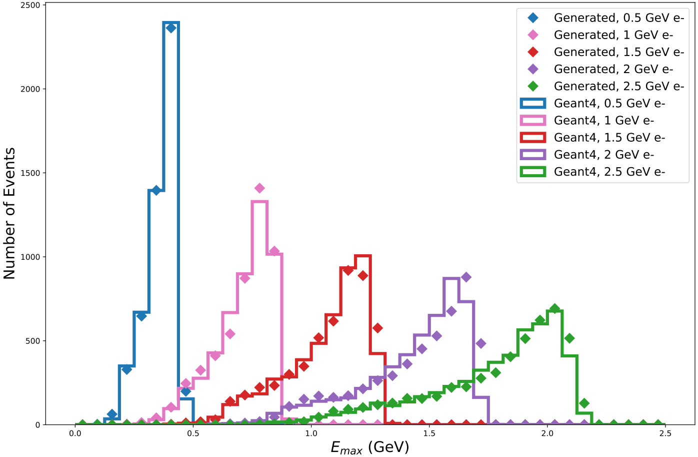


WASSERSTEIN GAN



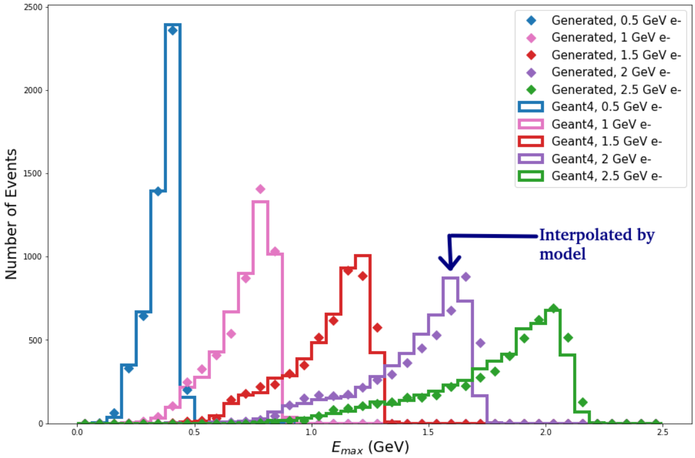
- **Critic** outputs a score based on how real the input images are.
- **Generator** outputs synthetic image from noise and labels.
- Additional **Energy** and **Position** constrainer networks are added to the model.

RESULTS



Maximum value of energy deposited in the 5 x 5 crystals

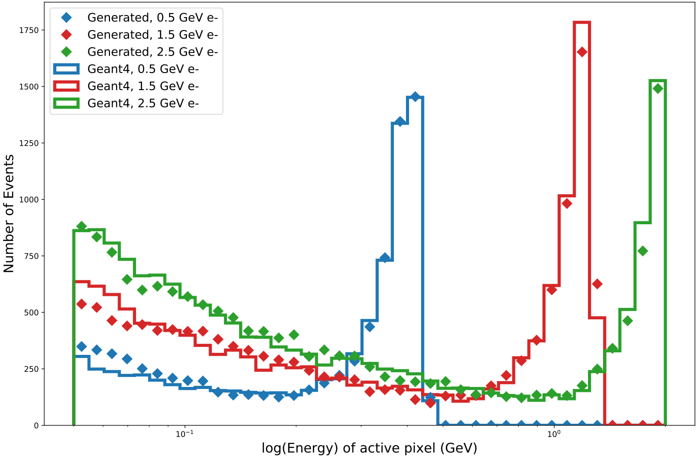
RESULTS



Interpolated by model

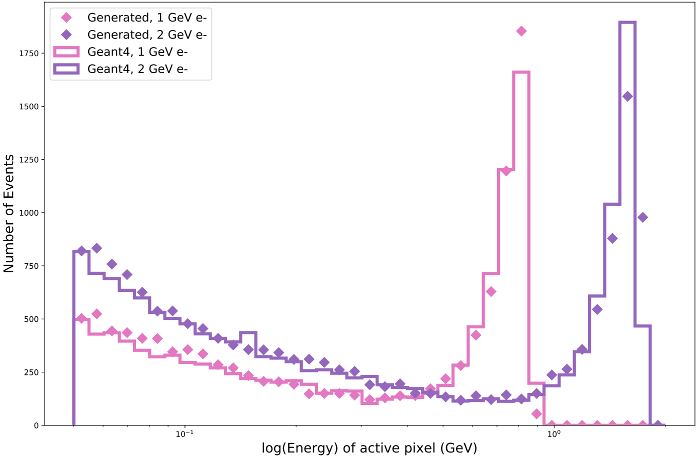
Maximum value of energy deposited in the 5 x 5 crystals

RESULTS



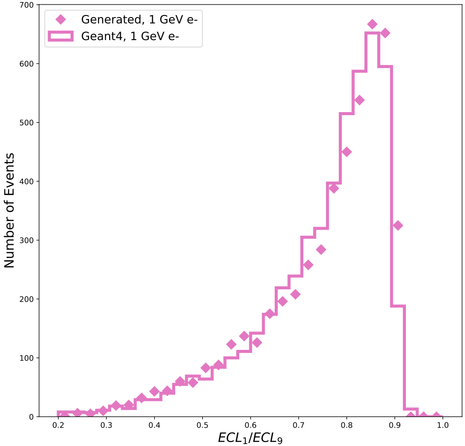
Distribution of single cell energy depositions

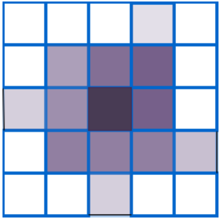
RESULTS



Distribution of single cell energy depositions

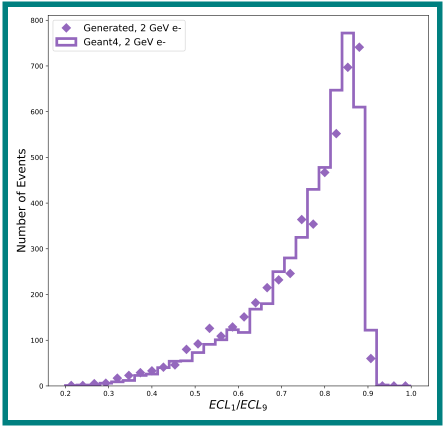
RESULTS



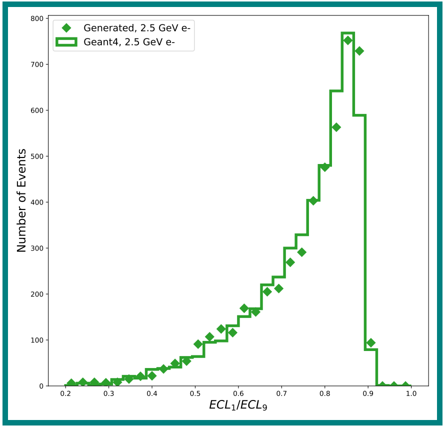


Ratio of energies of the central crystal and 3x3 crystals around the central crystal.

RESULTS

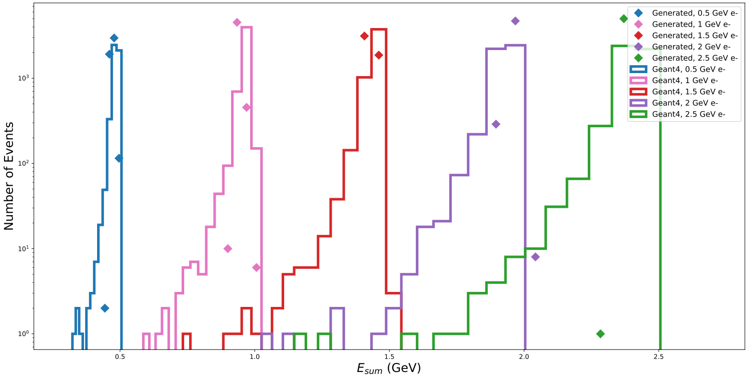


Generated, 2 GeV e-
Geant4, 2 GeV e-
Number of Events
0
100
200
300
400
500
600
700
800
ECL_1/ECL_9
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0



Generated, 2.5 GeV e-
Geant4, 2.5 GeV e-
Number of Events
0
100
200
300
400
500
600
700
800
ECL_1/ECL_9
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0

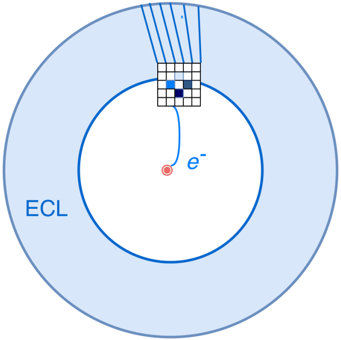
RESULTS



Total sum of energy deposited in the 5 x 5 crystal.

SUMMARY AND ON GOING WORKS

- The WGAN simulated results 0.5, 1.0, 1.5, 2.5 GeV electrons on 5x5 crystals show good agreement with the electrons simulated by Geant4.
- The model is able to interpolate 2.0 GeV electrons well.
- Next steps:
  - Belle II MC shower simulation.
  - Inclusion of additional features.
  - Fast simulation of pions and muons.

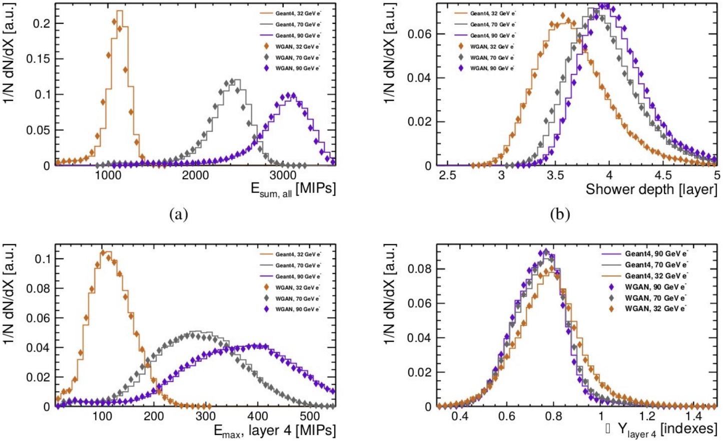


BACKUP

BACKUP FRAMES

- Generator
  - 2 x linear
  - 1 x Transposed Convolution
  - 2 x Convolution
  - Activation: LeakyReLU
- Critic
  - 4 x Convolution
  - 2 x linear
  - Activation: LeakyReLU
- Constrainer Networks
  - 2 x Convolution
  - 1 x linear
  - Activation: LeakyReLU

RESULTS BY THORBEN QUAST



(a)

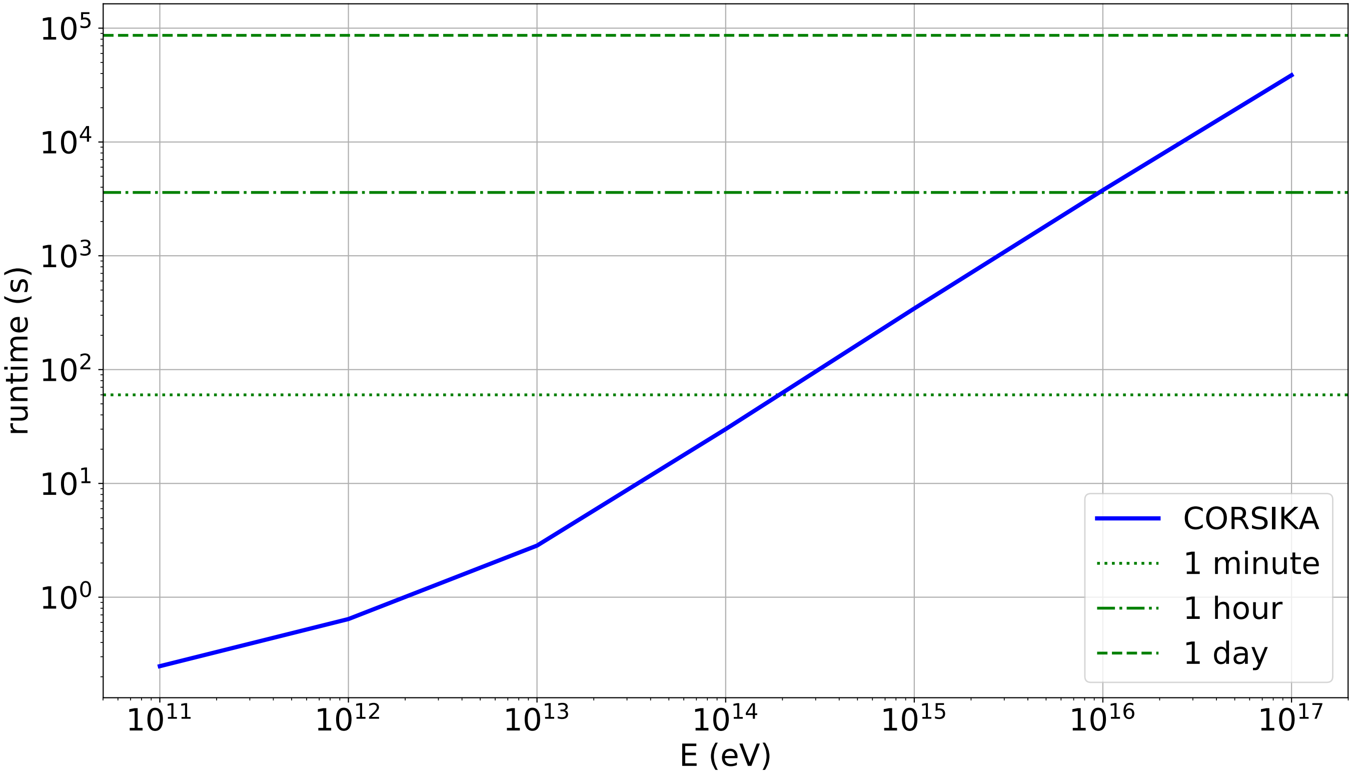
(b)

(c)

(d)

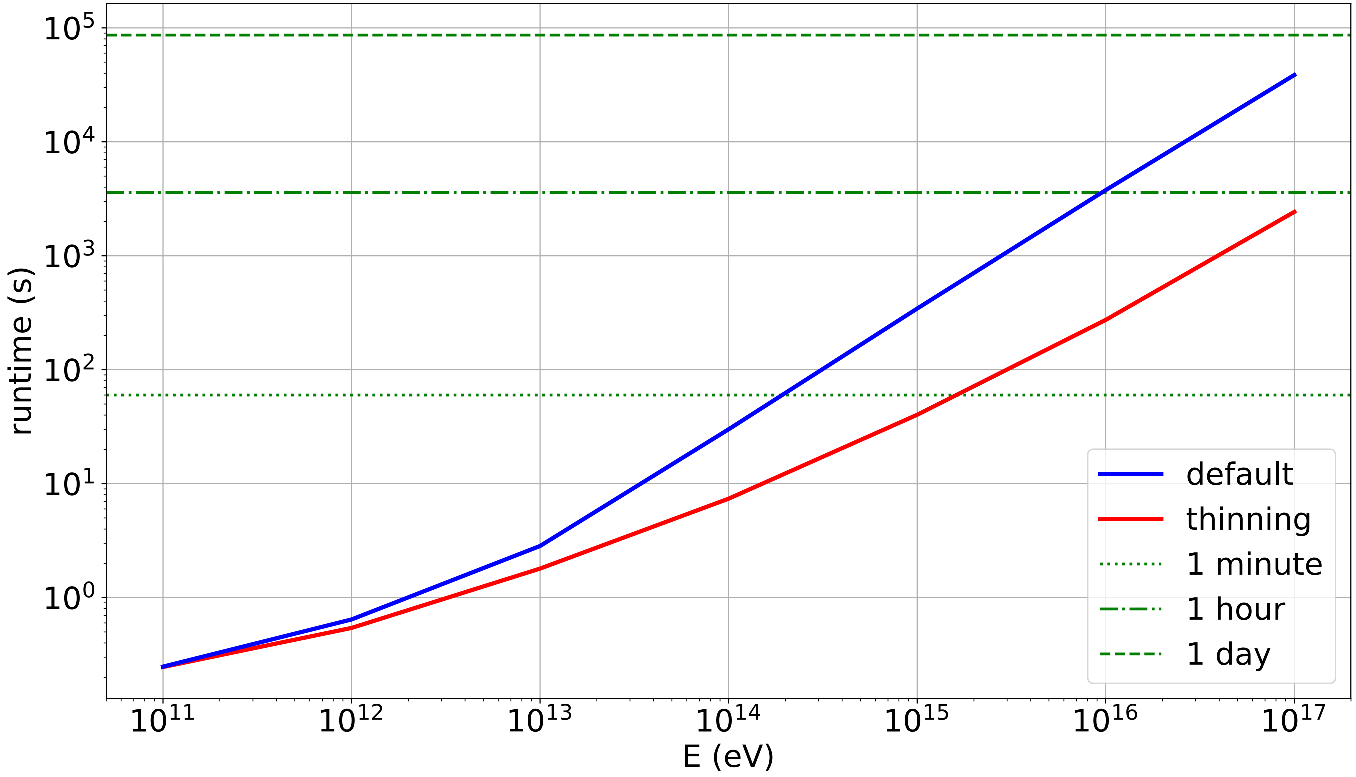
Backup

Motivation

- 

- The time complexity of CORSIKA 7 simulations rises approximately linearly with the primary particle energy

Thinning

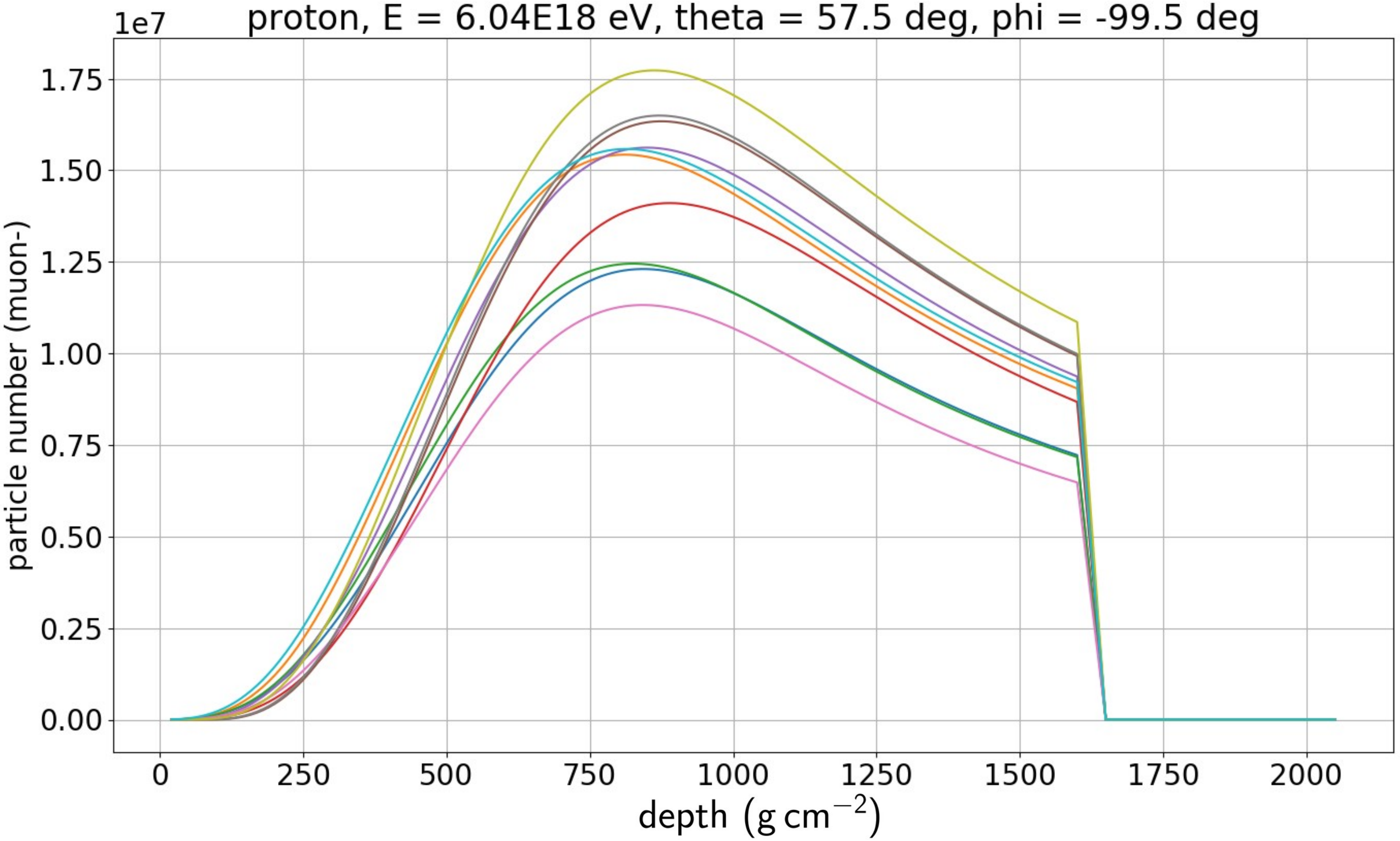


- Reduces (effective) particle content by particle-aggregation
- Preserves shower properties to leading order
- Reduces shower-to-shower fluctuations

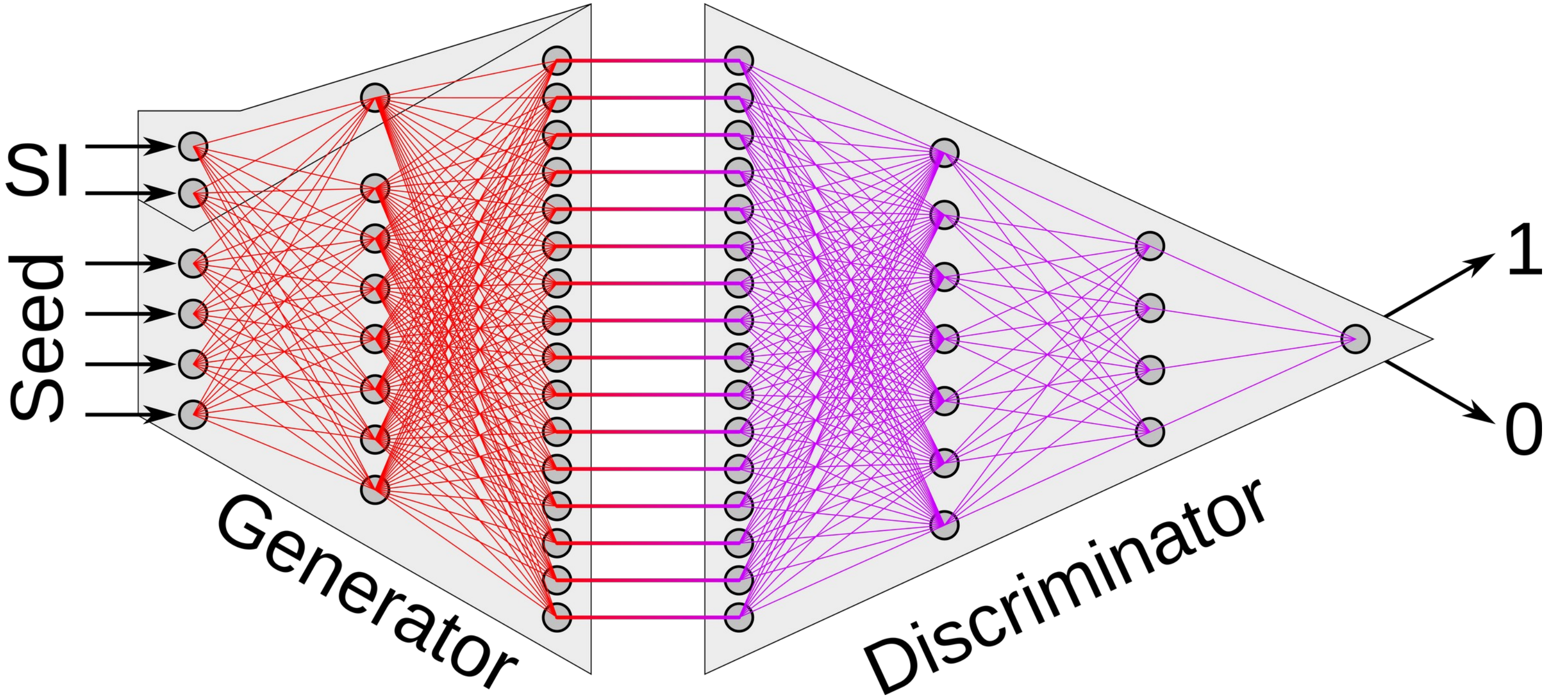
(conditional) WGAN

- Generator:
  - 5x Dense (+3)
  - 5x TransposeConvolution + Convolution (+2)
  - Activation: tanh
- Discriminator:
  - 3x Dense (+2)
  - 7x Convolution (+3)
  - 2x Dense (+1)
  - Activation: tanh
- Trainable parameters: 79.072.457

Shower-to-Shower Fluctuations

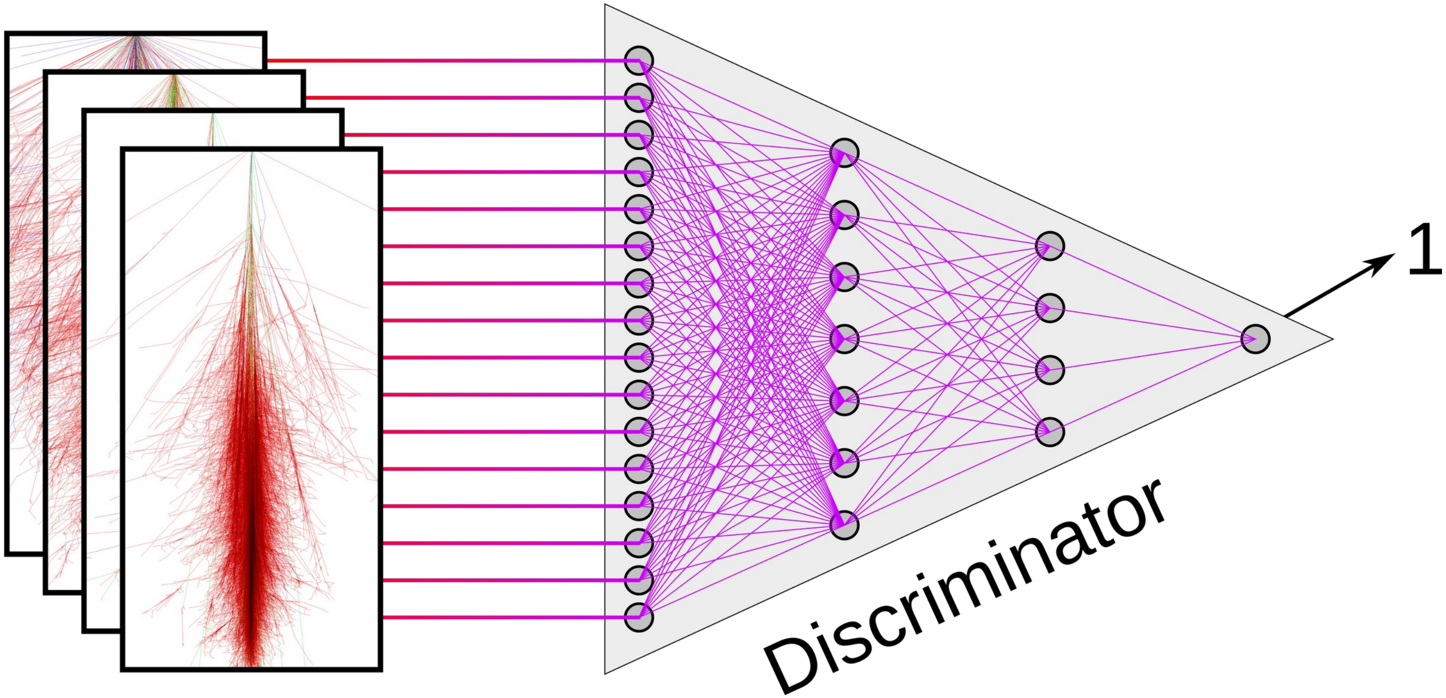


Generative Adversarial Neural Network (GAN)



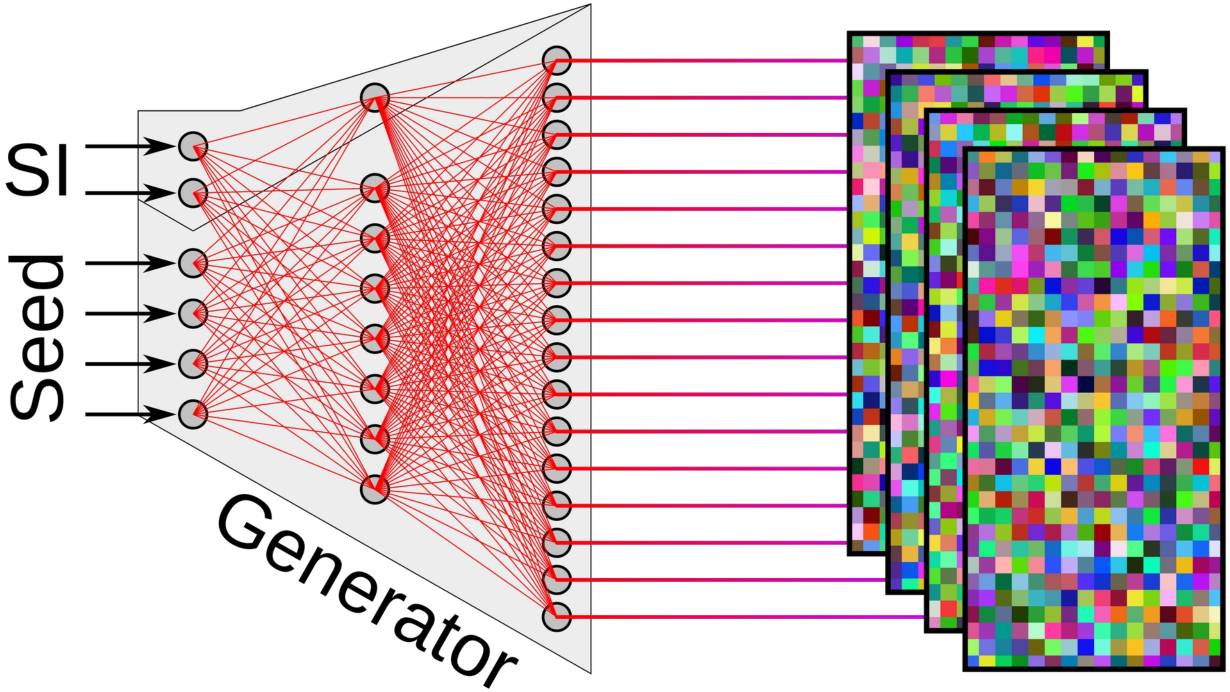
- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

Training: Discriminator (Part 1)



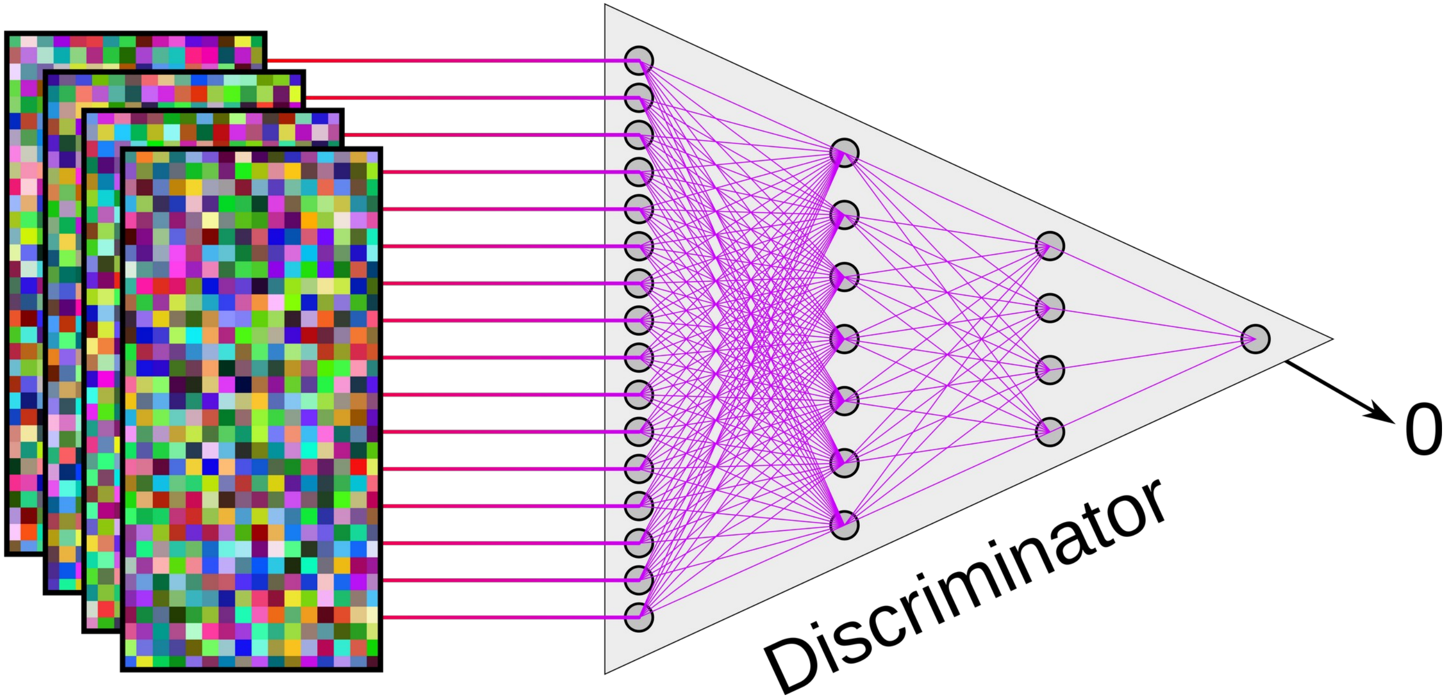
- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

Training: Sampling



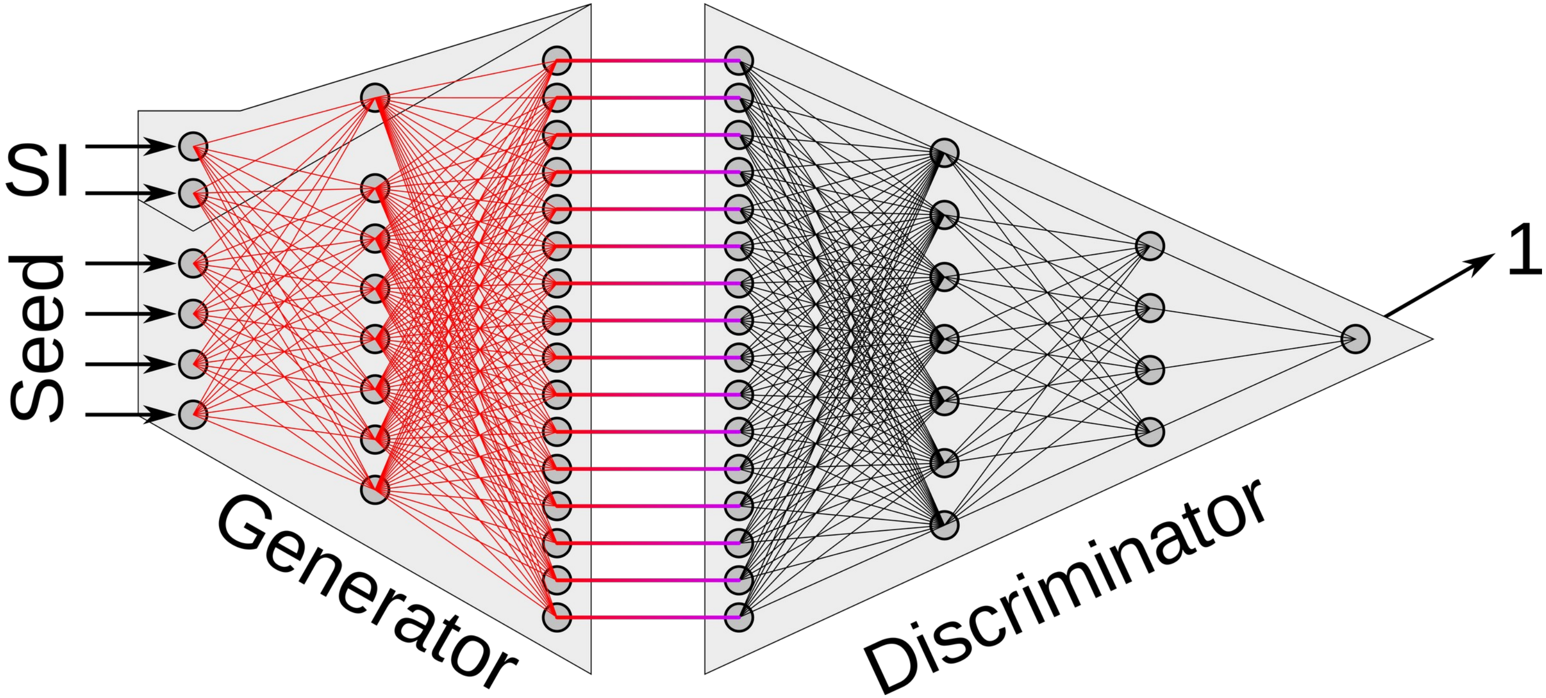
- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

Training: Discriminator (Part 2)



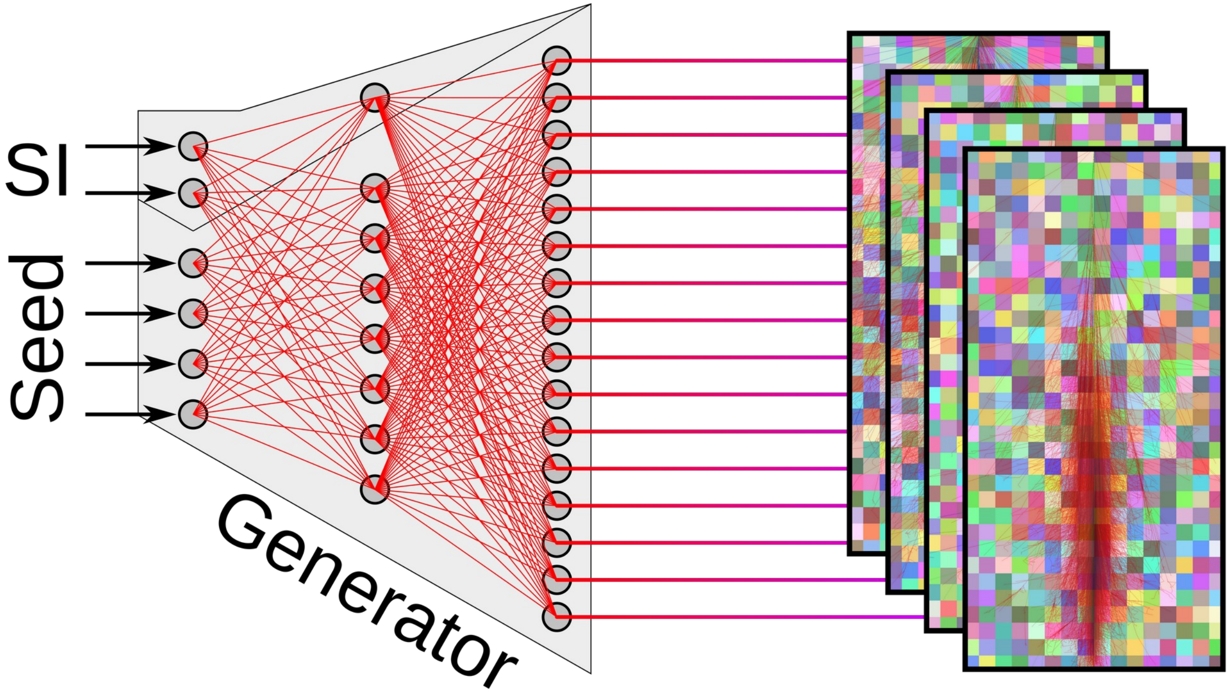
- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

Training: Generator



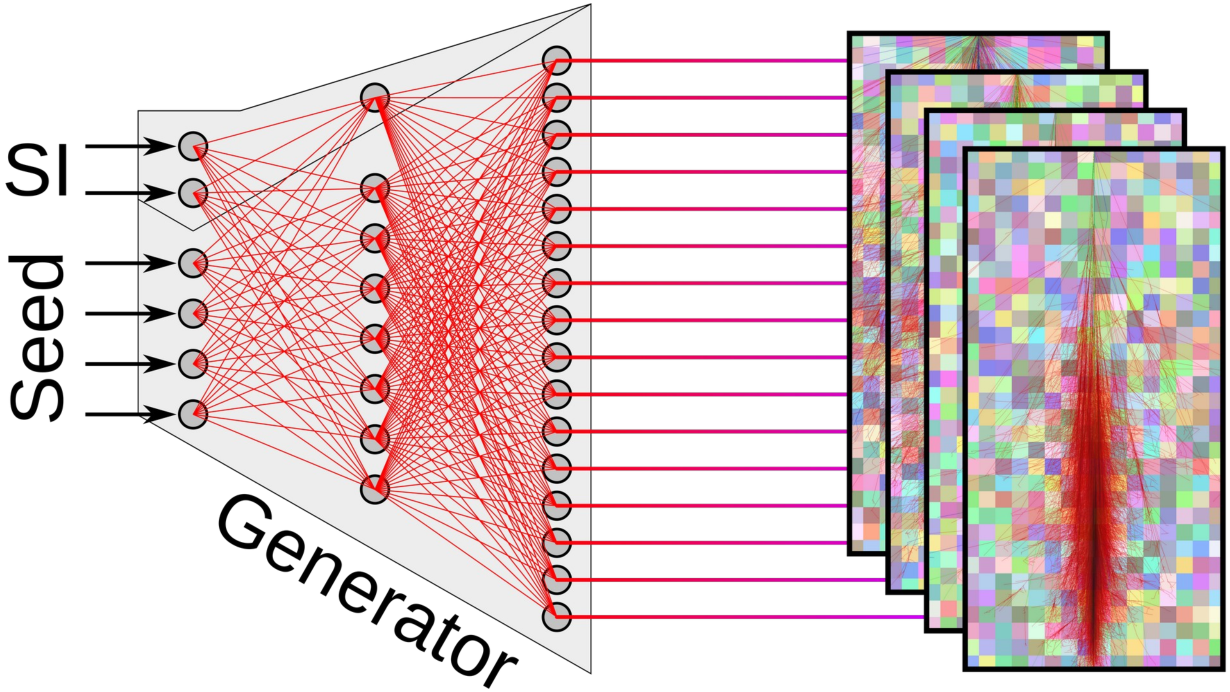
- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

Training: Result



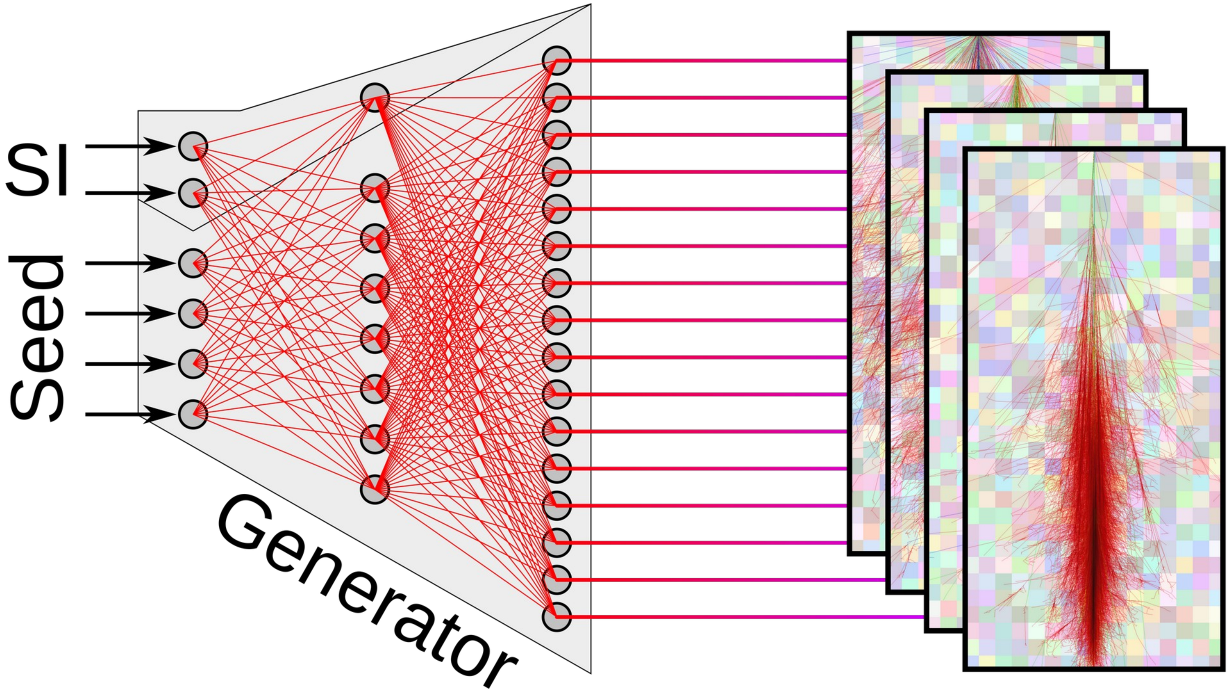
- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

Training: Result



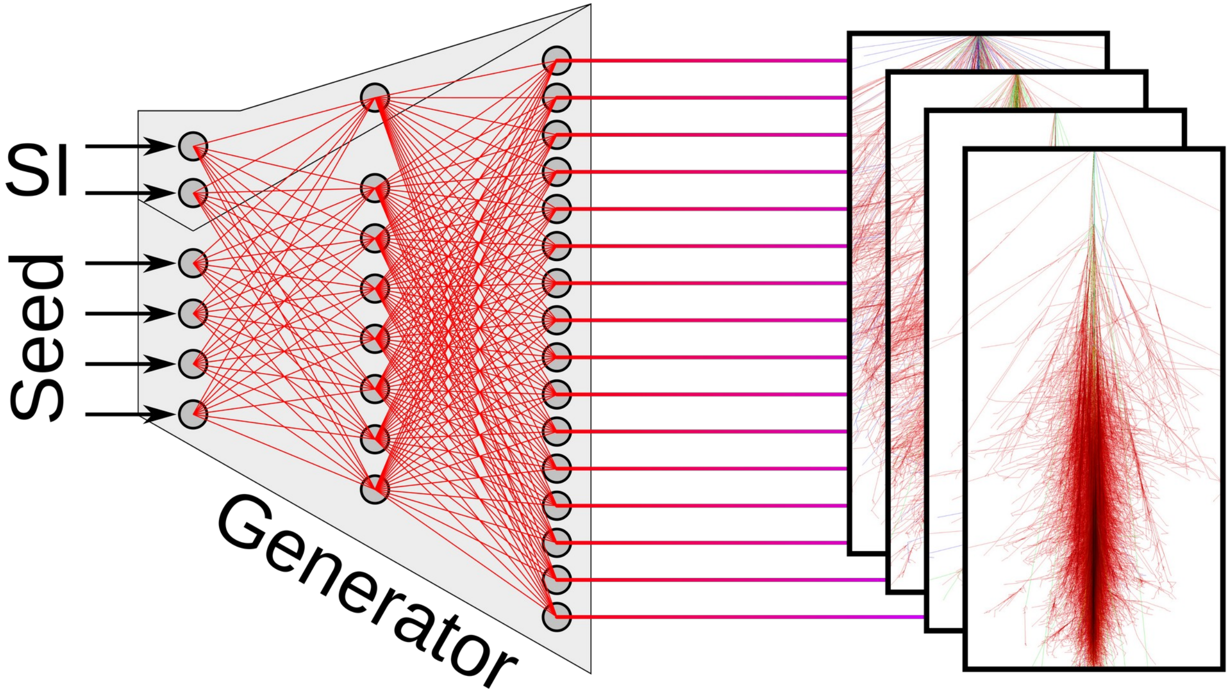
- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

Training: Result



- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

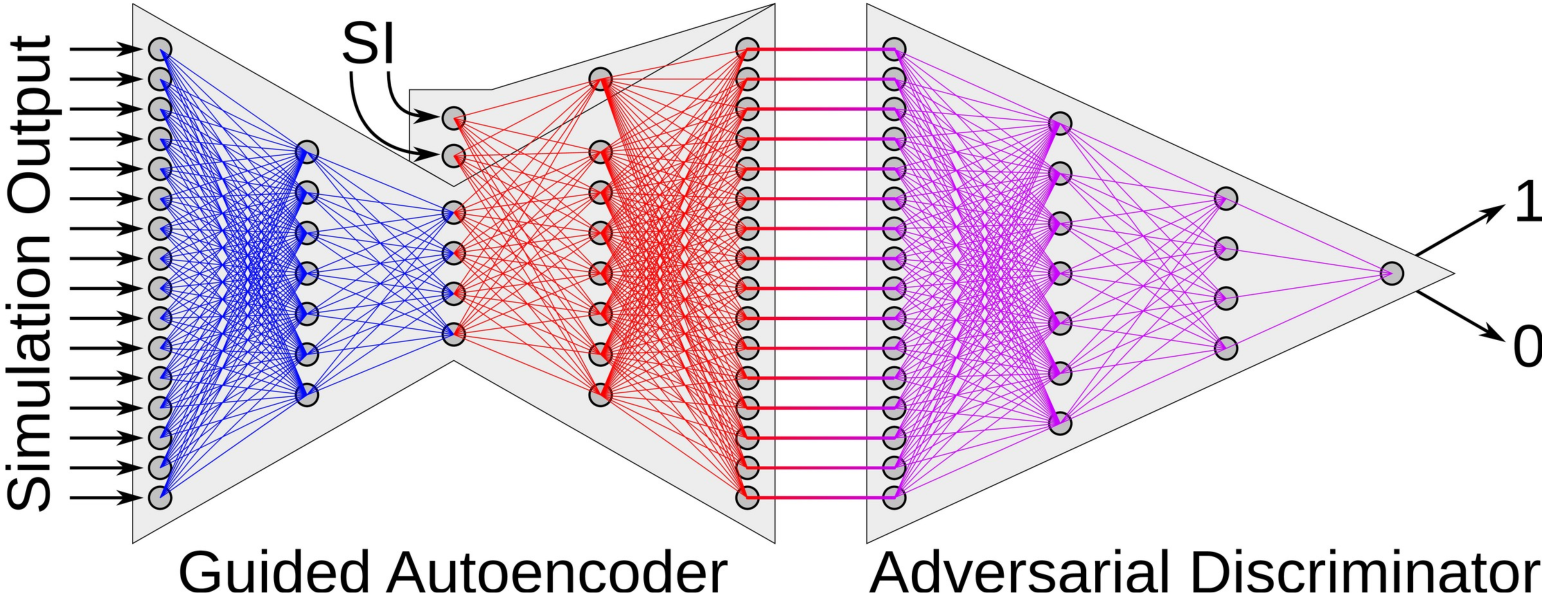
Training: Result



- Train discriminator on real (1) and generated (0) data
- Train generator to outsmart the discriminator

Fast Implicit Simulation Heuristic (FISH)

- Autoencoder with Adversarial Metric



- Simulation Input (SI) can be extended with meta-parameters
- Discriminator can be refined with real measurements

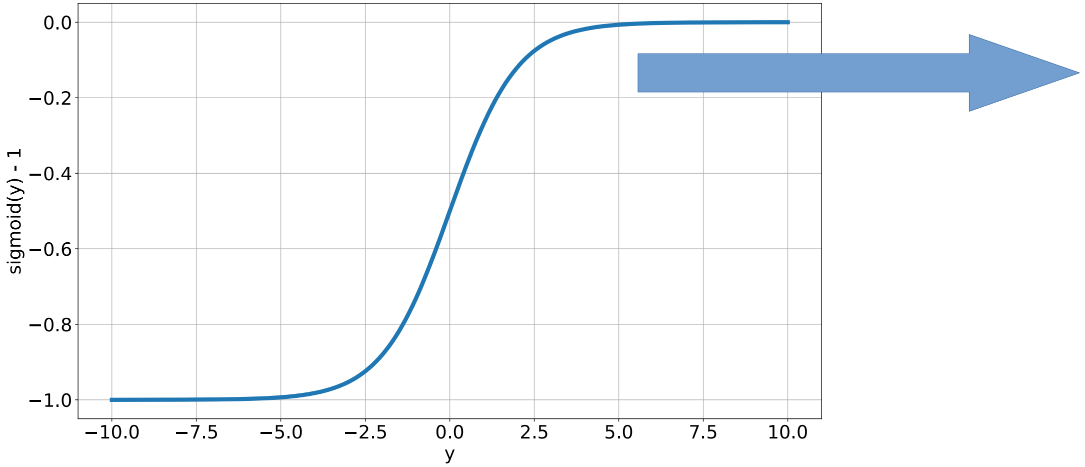
Cross Entropy

- $$\mathrm{CE} = \sum_i -z_i \cdot \log(p_i) + (z_i - 1) \cdot \log(1 - p_i)$$

  with z (true) label and p probability (NN output)

- z = 1: $-\log(\mathrm{sigmoid}(y))$

  $$\Rightarrow \frac{d}{dy}(-\log(\mathrm{sigmoid}(y))) = \mathrm{sigmoid}(y) - 1$$



vanishing gradients

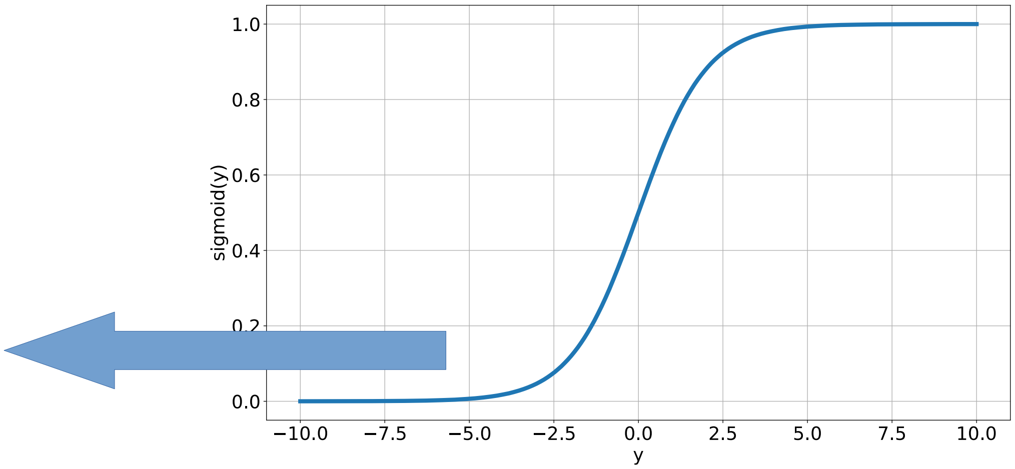
Cross Entropy

- $$\mathrm{CE} = \sum_i -z_i \cdot \log(p_i) + (z_i - 1) \cdot \log(1 - p_i)$$

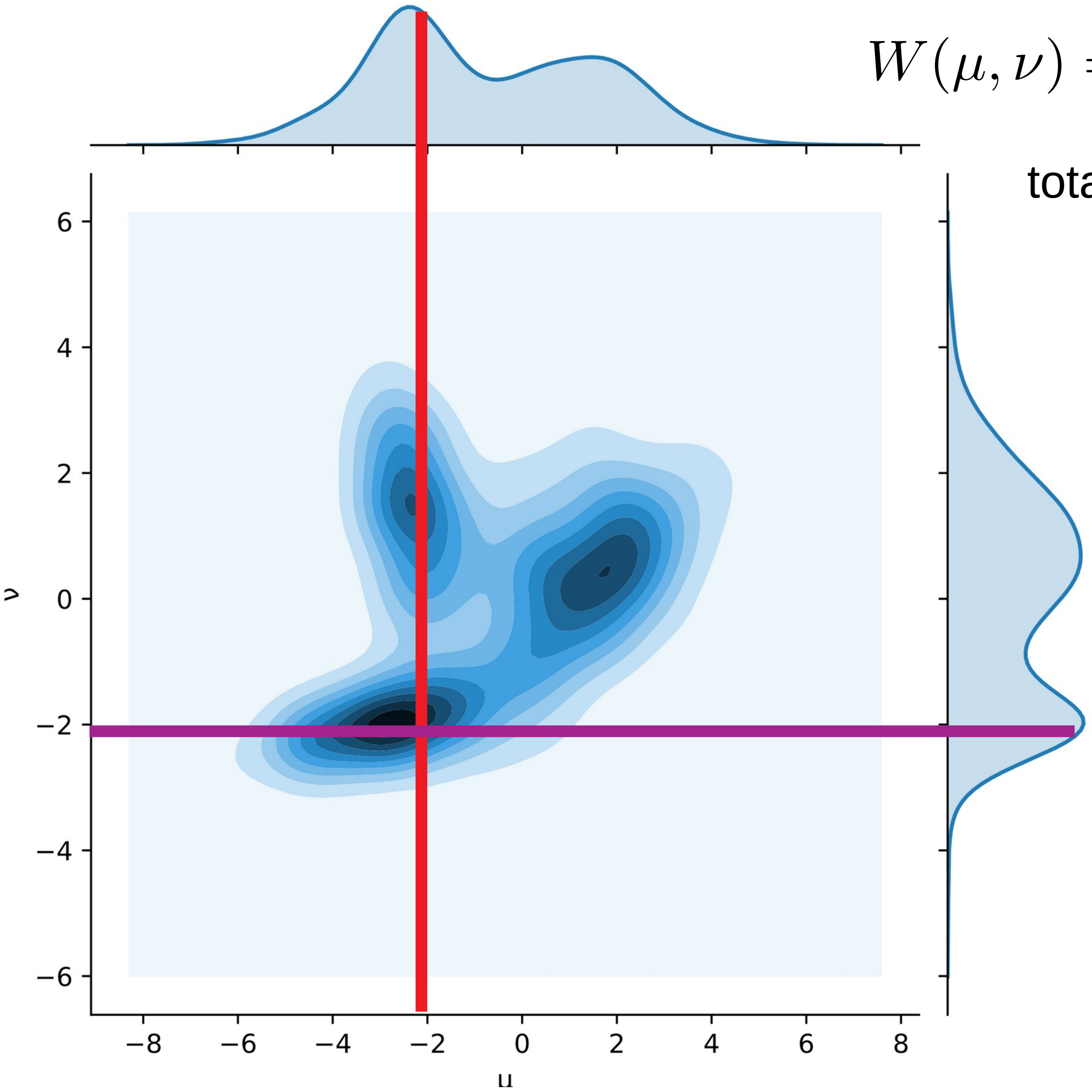
  with z (true) label and p probability (NN output)

- z = 0: $-\log(1 - \mathrm{sigmoid}(y))$

  $$\Rightarrow \frac{d}{dy}(-\log(1 - \mathrm{sigmoid}(y))) = \mathrm{sigmoid}(y)$$



Wasserstein Distance [5]



$$W(\mu,\nu) = \inf_{\gamma} \int d(x,y) \cdot \gamma(x,y)\ dxdy$$

total (minimal) cost to move all mass

$$\int d(x,y) \cdot \gamma(x,y) dy = c(x)$$

cost to move mass above/below x

$$\int \gamma(x,y) dy = \mu(x)$$

red: total mass pilled up at x

Kantorovich-Rubinstein Duality

- $$W(\mu,\nu) = \sup_{f \in Lip_{\leq 1}} \mathbb{E}_{x\sim\mu}[f(x)] - \mathbb{E}_{y\sim\nu}[f(y)]$$
- f = Neural Network
- Lipschitz continous: $|f(x_1) - f(x_2)| \leq L \cdot ||x_1 - x_2||$
- Gradient is bounded → Gradient penalty $|||\nabla f|| - 1| \rightarrow 0$

Gradient Penalty

- $W(\mu, \nu) = \sup_{f \in Lip_{\leq 1}} \mathbb{E}_{x \sim \mu}[f(x)] - \mathbb{E}_{y \sim \nu}[f(y)]$
- $\mathbb{E}_{x \sim \mu}[f(x)] \to \infty \qquad \mathbb{E}_{y \sim \nu}[f(y)] \to -\infty$
- $f \to a \cdot f$ and $a \to \infty$
- But $|a \cdot f(x) - a \cdot f(y)| \leq L \, ||x - y||$

  $\Rightarrow \; a \cdot ||\nabla f|| \leq L$

Cross Entropy Loss

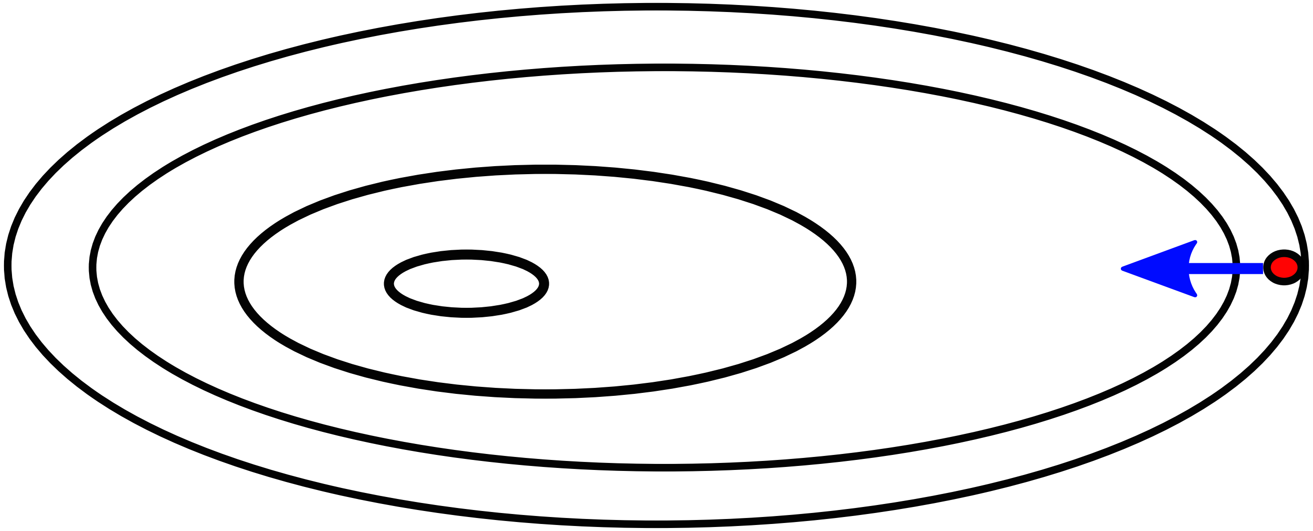
- $$\begin{aligned}\mathrm{CE} = &-\mathbb{E}_{x\sim\mu}[\log(\mathrm{sigmoid}(f(x)))] \\ &-\mathbb{E}_{y\sim\nu}[\log(1-\mathrm{sigmoid}(f(y)))]\end{aligned}$$
- $\mathbb{E}_{x\sim\mu}[f(x)] \rightarrow \infty \qquad \mathbb{E}_{y\sim\nu}[f(y)] \rightarrow -\infty$
- $f \rightarrow a \cdot f$ and $a \rightarrow \infty$
- But ~~$|a \cdot f(x) - a \cdot f(y)| \leq L\,||x-y||$~~

  ~~$\Rightarrow \quad a \cdot ||\nabla f|| \leq L$~~

vanishing gradients

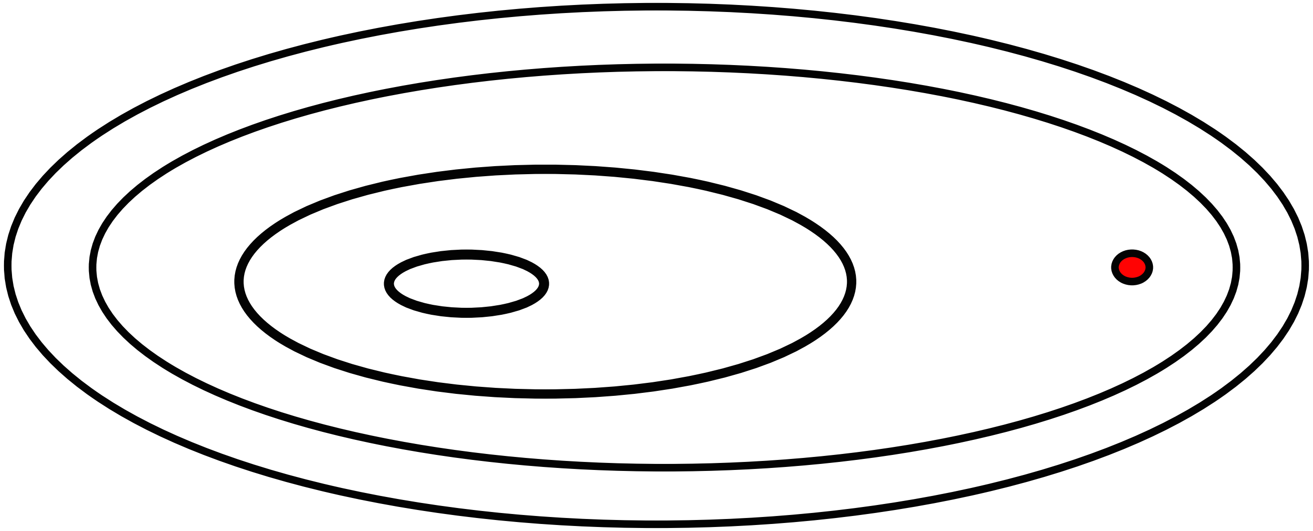
Learning Rate and Momentum

- Ordinary classification:
  - Gradient



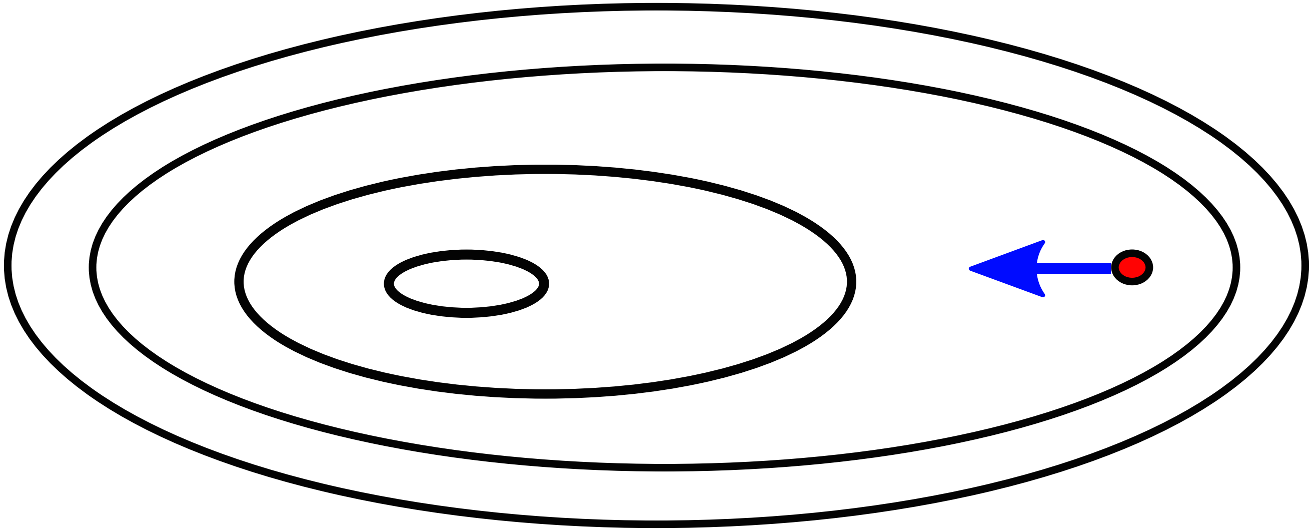
Learning Rate and Momentum

- Ordinary classification:
  - Step



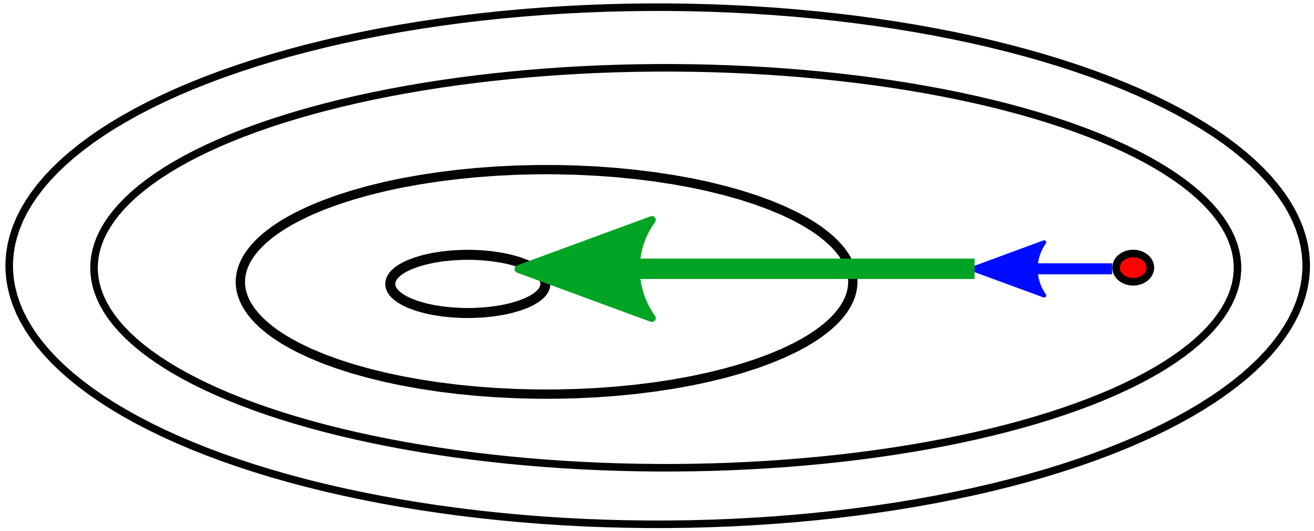
Learning Rate and Momentum

- Ordinary classification:
  - Gradient



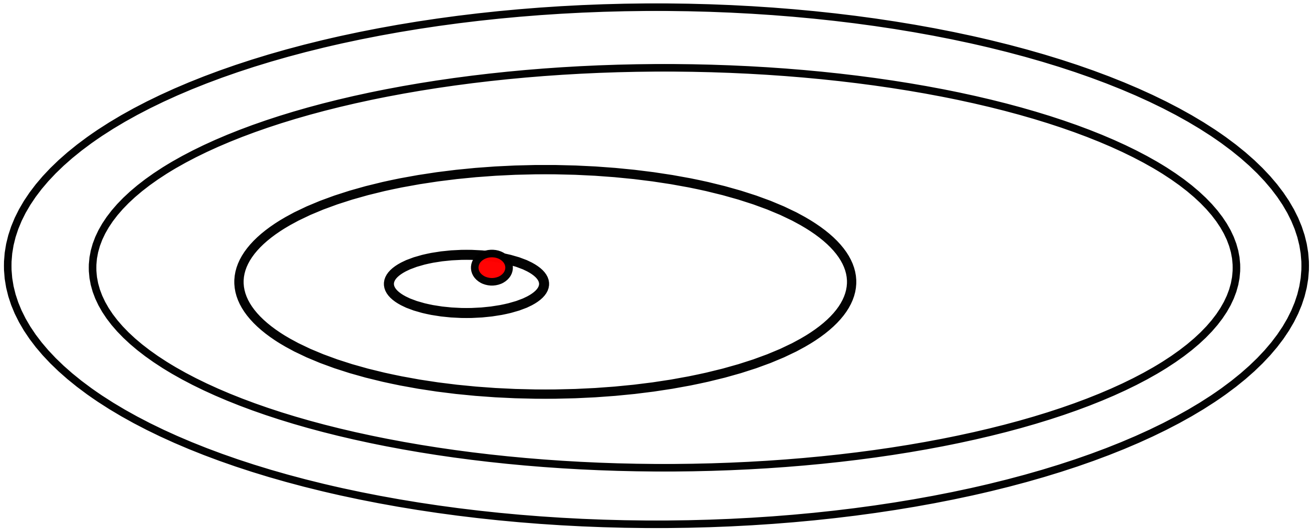
Learning Rate and Momentum

- Ordinary classification:
  - Momentum



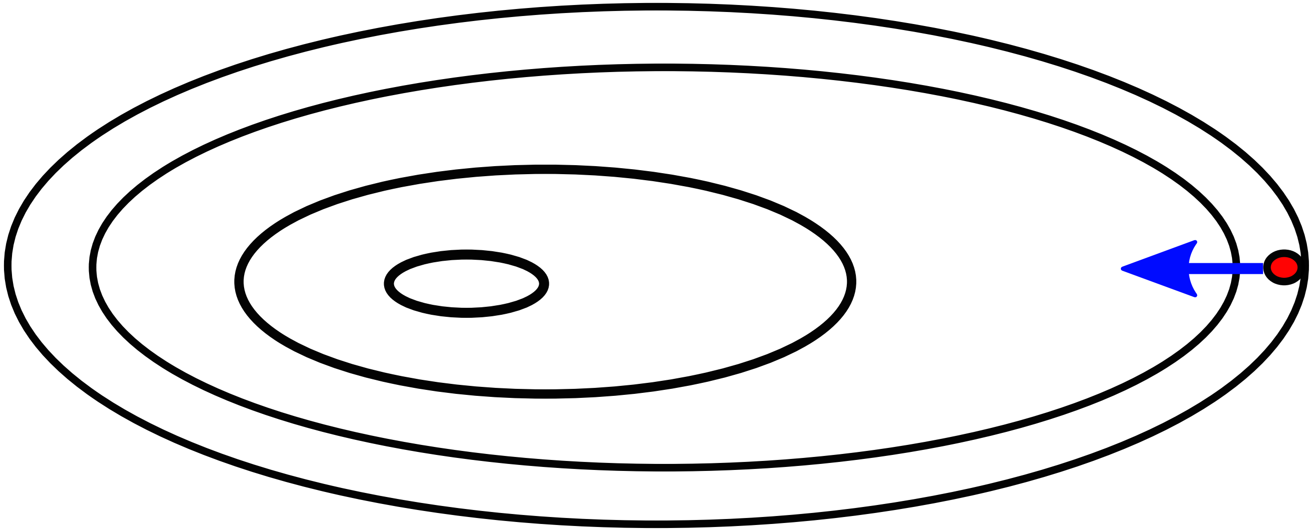
Learning Rate and Momentum

- Ordinary classification:
  - Step



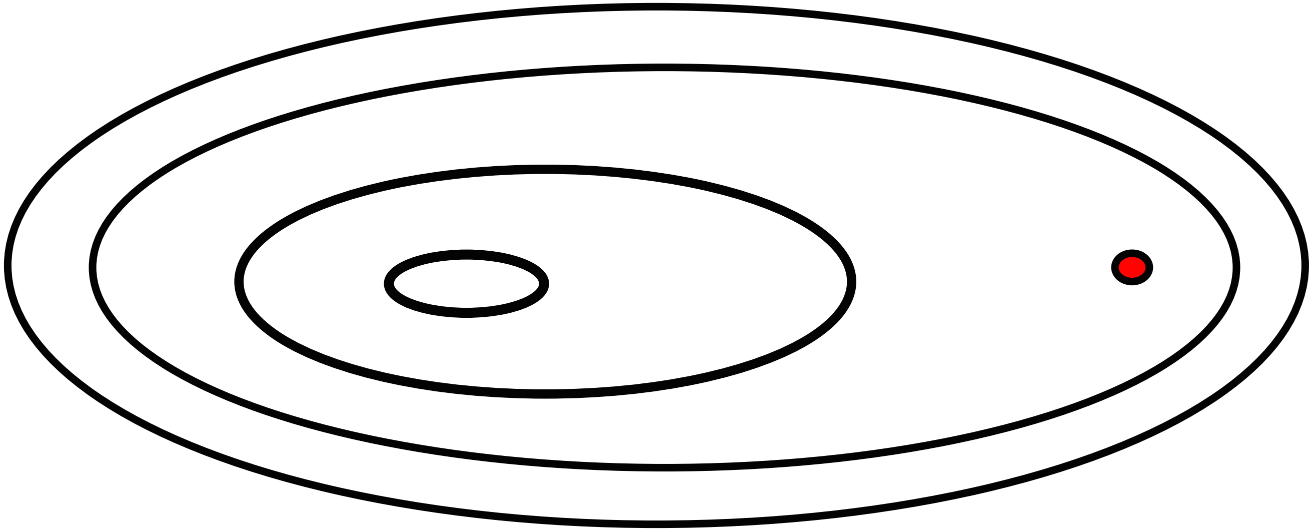
GAN: Learning Rate and Momentum

- Discriminator classification:
  - Gradient



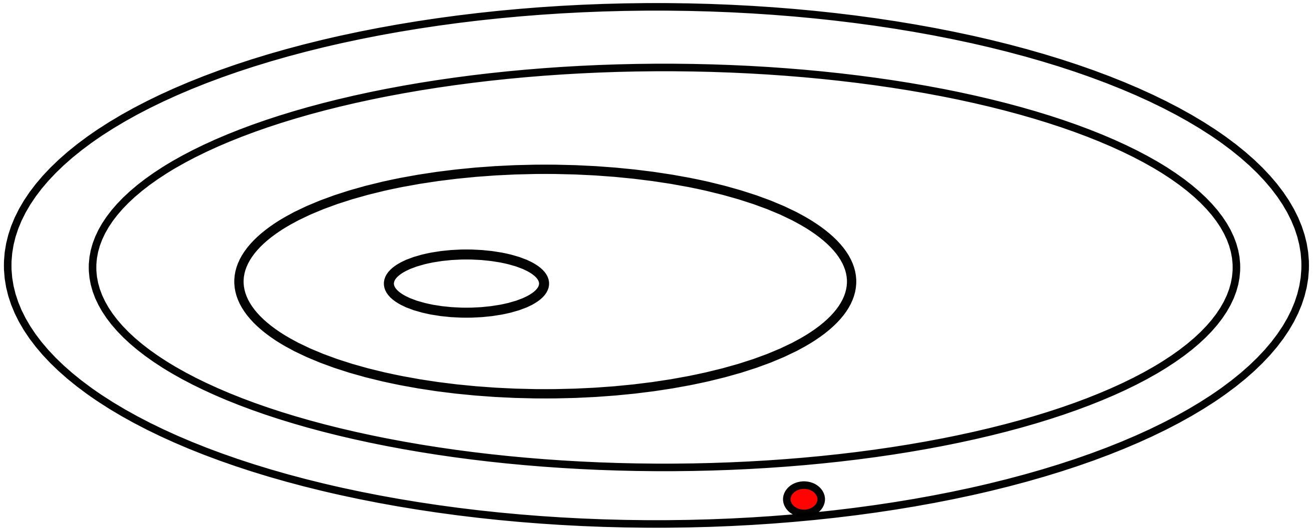
GAN: Learning Rate and Momentum

- Discriminator classification:
  - Step



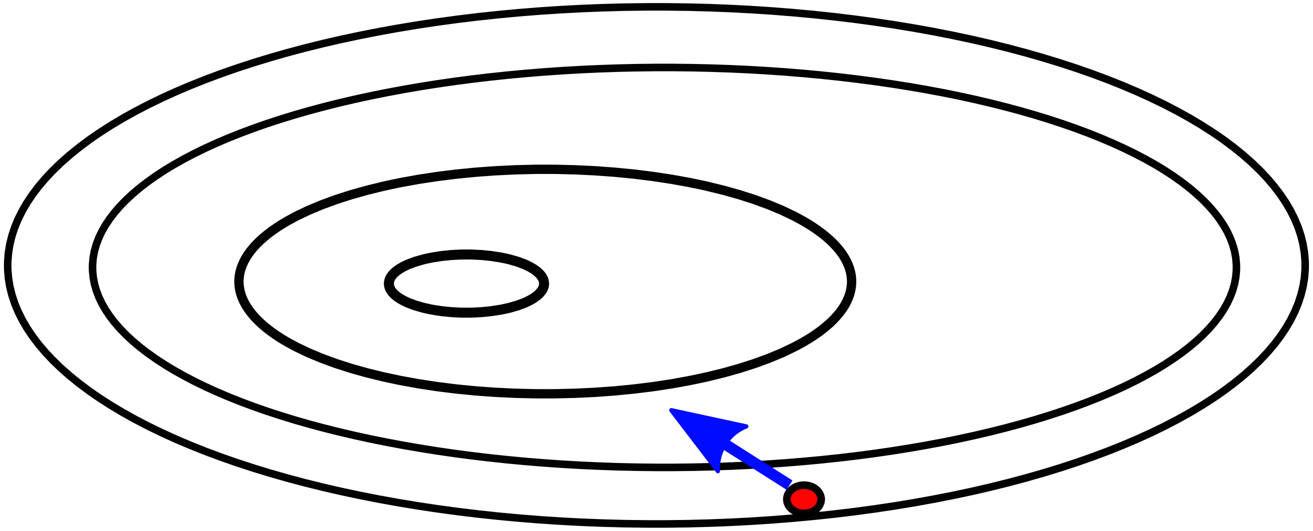
GAN: Learning Rate and Momentum

- Discriminator classification:
  - Generator training



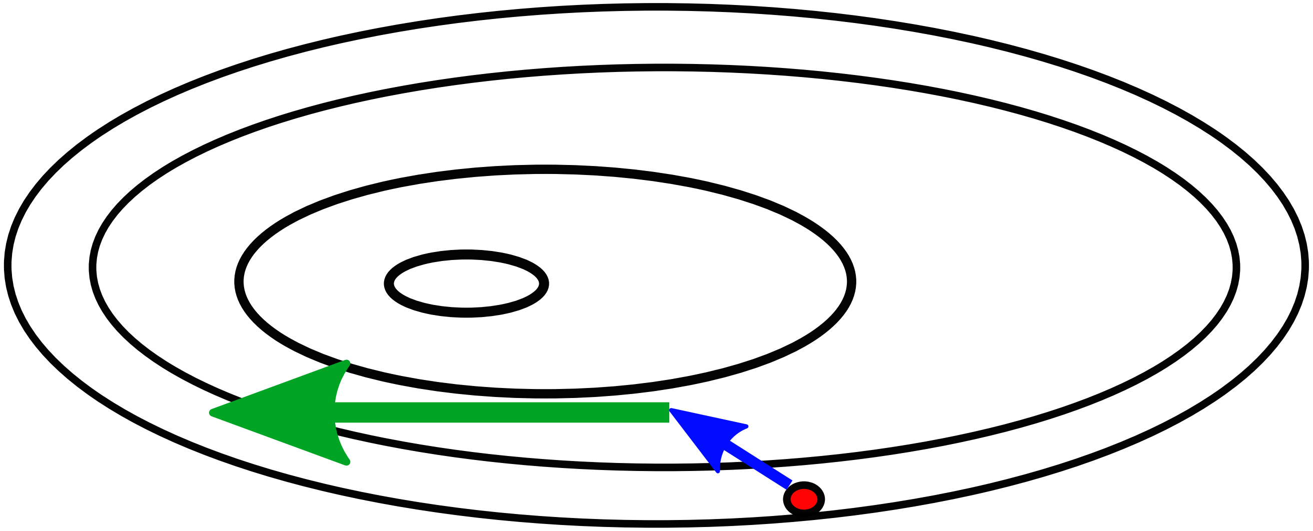
GAN: Learning Rate and Momentum

- Discriminator classification:
  - Gradient



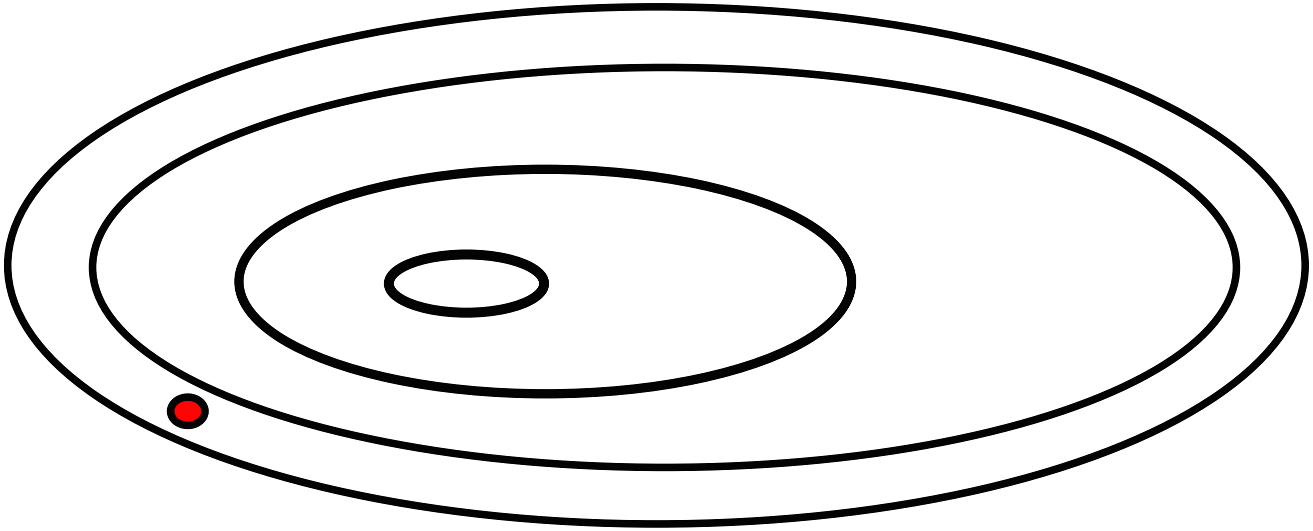
GAN: Learning Rate and Momentum

- Discriminator classification:
  - Momentum



GAN: Learning Rate and Momentum

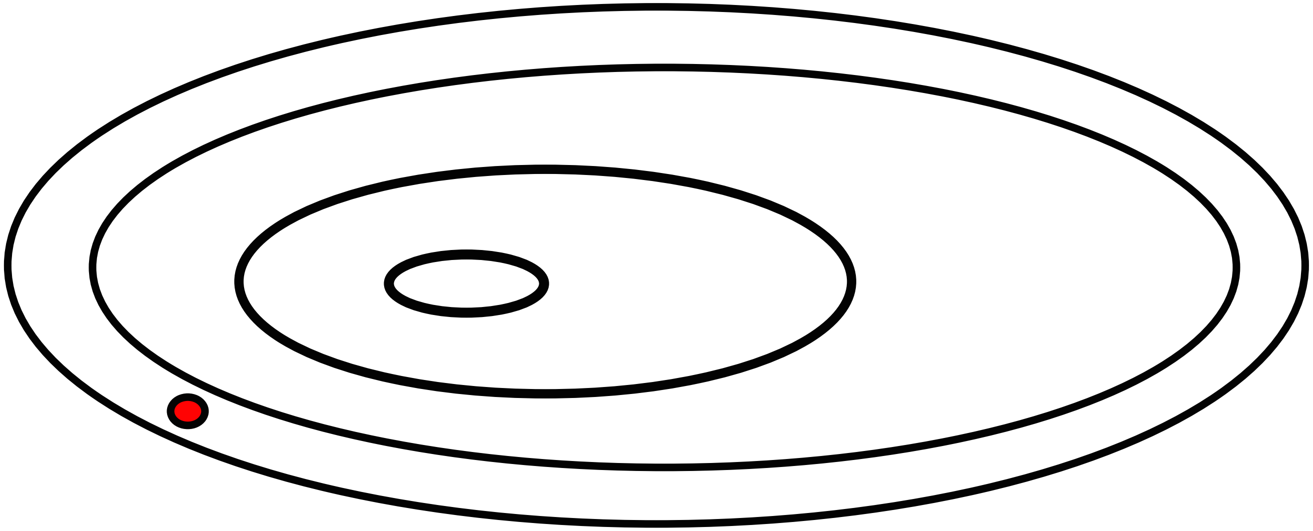
- Discriminator classification:
  - Step



GAN: Learning Rate and Momentum

- Discriminator classification:

  - Step



- Adam: $\alpha \leq 10^{-4} \quad \beta_1 = 0.5 \quad \beta_2 = 0.9$

Constrainer

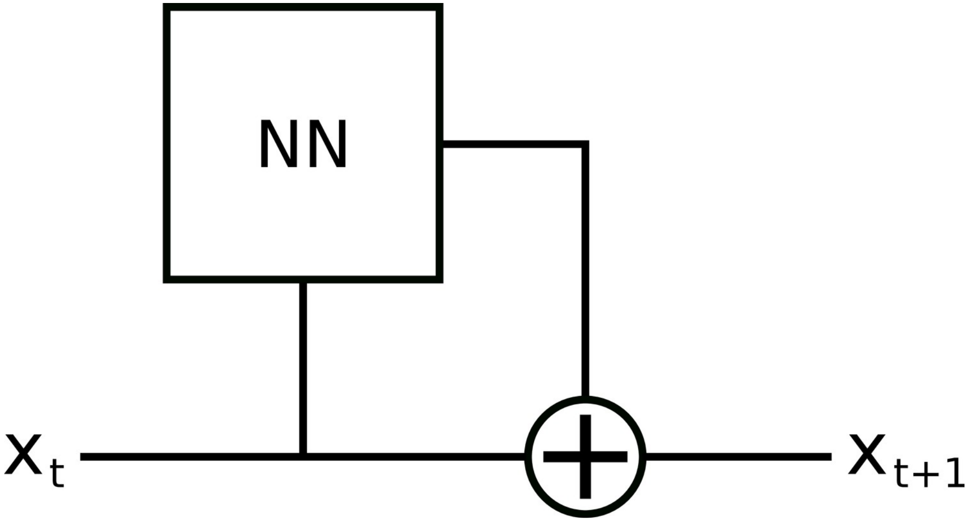
- Problematic because:
  - Constainer network = reconstruction of label

- What if label information is not present? (thermalization)

- Generator is
  - forced to fullfill reconstruction loss
  - will put constrainer demands into generated data
  - no measure on reconstruction uncertainty

Adjustable Accuracy [6]

- ResNet



- Translate to ordinary differential equation (ODE)

$$x_{t+1} = x_t + f(x_t, \theta_t) \;\Rightarrow\; \frac{dx(t)}{dt} = f(x(t), t, \theta)$$

- Solve with standard ODE solver

- Adapt solver accuracy on the fly (training: high, inference: low)

References

- Title picture: Photo by Pixabay from Pexels
- Backup picture: Photo by Anthony from Pexels

- [1] CORSIKA 7: https://www.ikp.kit.edu/corsika/
- [2] DenseNet: https://arxiv.org/abs/1608.06993
- [3] StyleGAN: https://arxiv.org/abs/1812.04948
- [4] InfoGAN: https://arxiv.org/abs/1606.03657
- [5] Wasserstein Distance picture:

  Lambdabadger / CC BY-SA (https://creativecommons.org/licenses/by-sa/4.0)
- [6] „Neural Ordinary Differential Equations“ - Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, David Duvenaud – arXiv: 1806.07366