

삼성전자 AI 전문가 과정 - Case Study

GAN Lab: Understanding Complex Deep Generative Models using Interactive Visual Experimentation

Authors

- Minsuk Kahng, Nikhil Thorat, Polo Chau, Fernanda Viégas, and Martin Wattenberg
- VAST 2018
- Collaboration between Georgia Tech and Google Brain/PAIR



Deep Learning Visualization Tools

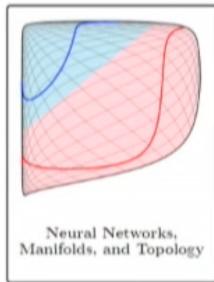
- Deep learning visualization tools presented at VIS



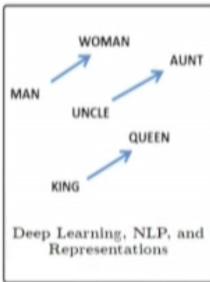
- Most tools are designed for experts

Introduction

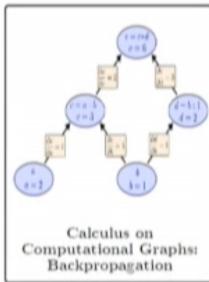
- Many non-experts want to learn ML



Neural Networks,
Manifolds, and Topology



Deep Learning, NLP, and
Representations



relu (24x24x8)
max activation: 3.11607, min: 0
max gradient: 0.17478, min: -0.1525

Activations



Activation Gradients



pool (12x12x8)
pooling size 2x2, stride 2
max activation: 3.11607, min: 0
max gradient: 0.17478, min: -0.1525

Activations



Activation Gradients



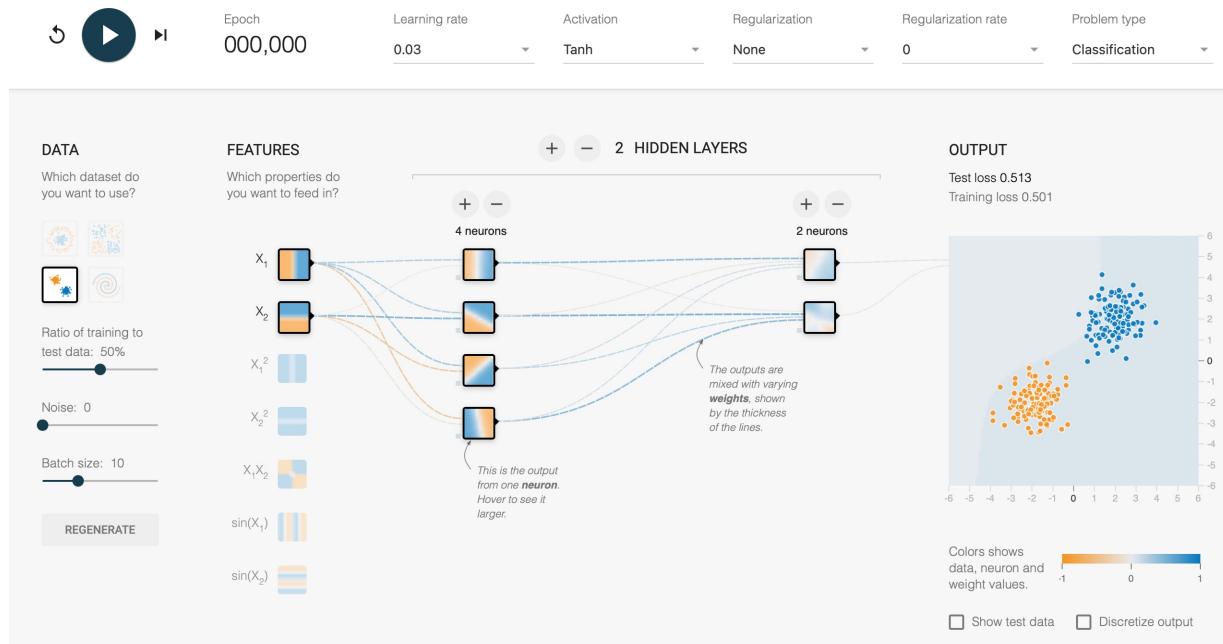
Chris Olah's blog

Andrej Karpathy's demo

- Can **VIS community** help this population?

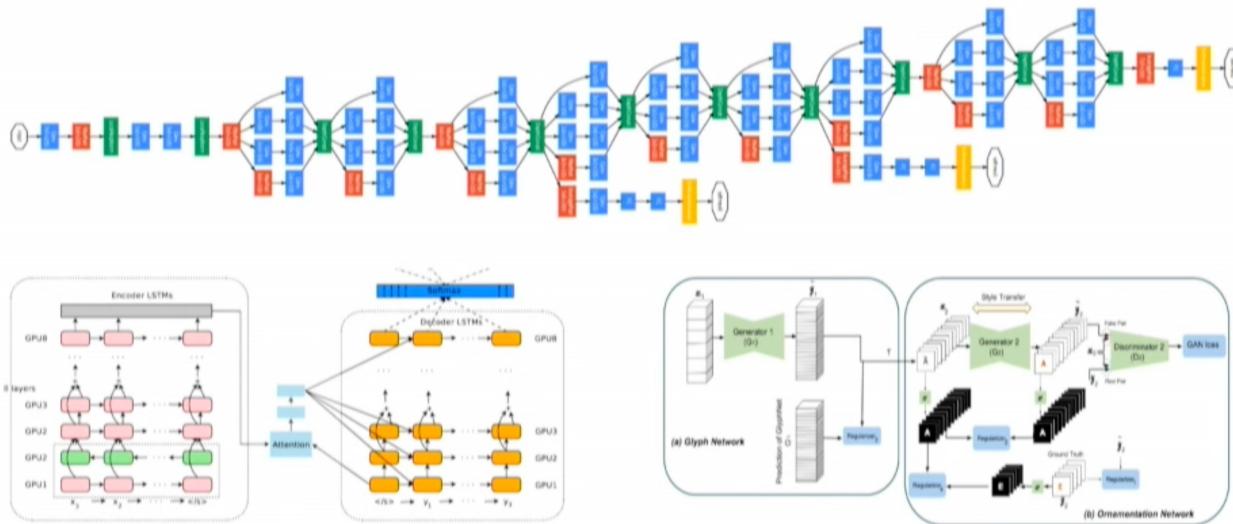
TensorFlow Playground

- <http://playground.tensorflow.org/>



Introduction

- Modern deep models are very complex



Generative Adversarial Networks (GANs)

- “the most interesting idea in the last 10 years in ML” – Yann LeCun
- Face images generated by BEGAN
[Berthelot et al., 2017]



- Hard to understand and train even for experts

Why are GANs hard to understand?

- Because a GAN uses two *competing* neural networks

Generator

synthesizes outputs



Counterfeiter

makes fake bills

Discriminator

spots fake

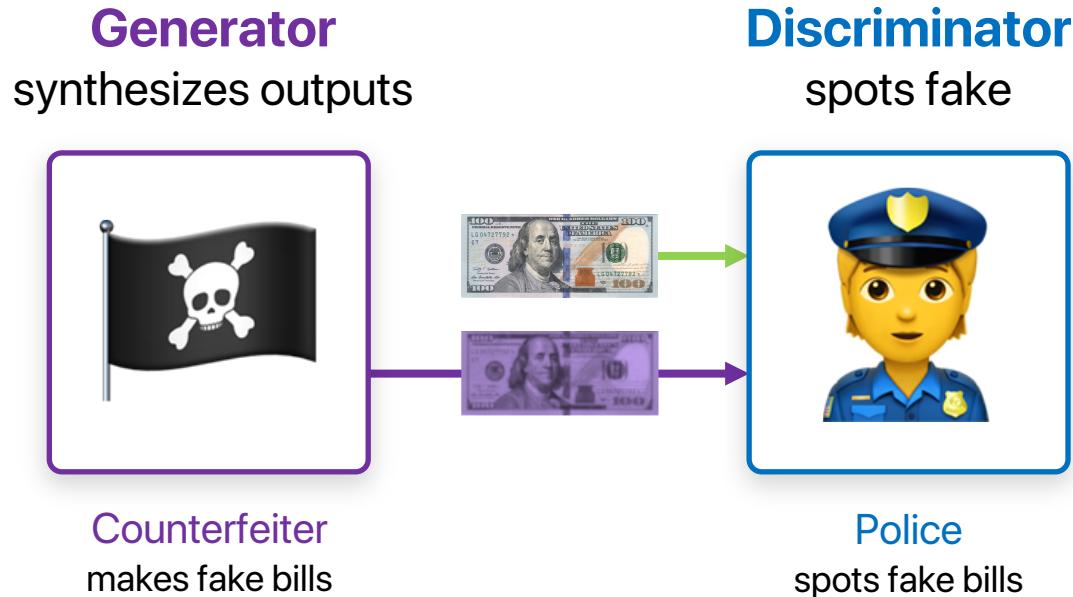


Police

spots fake bills

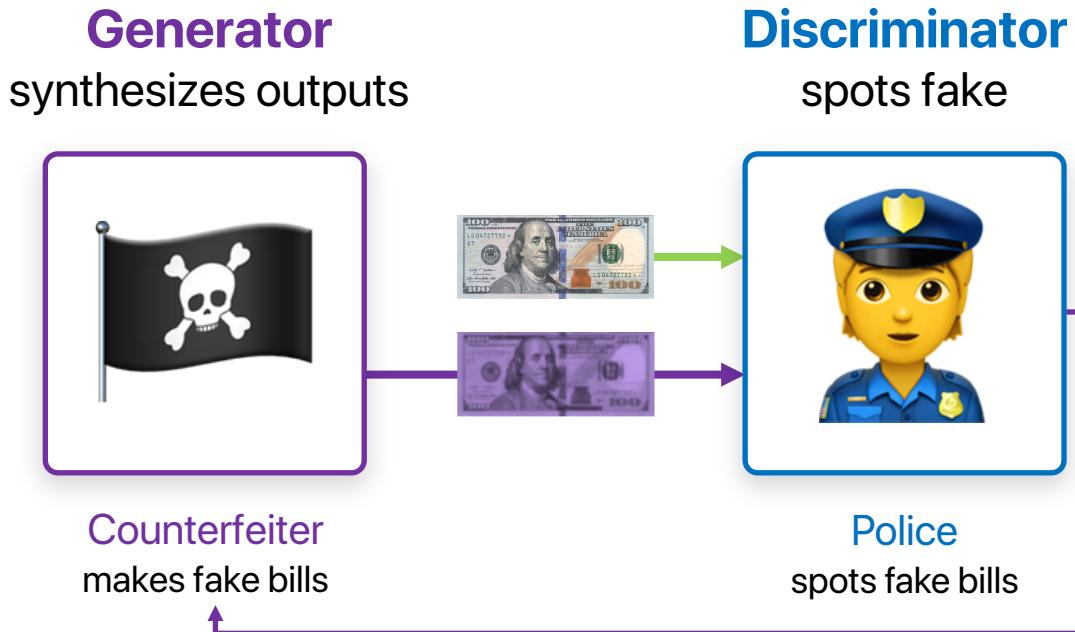
Why are GANs hard to understand?

- Because a GAN uses two *competing* neural networks



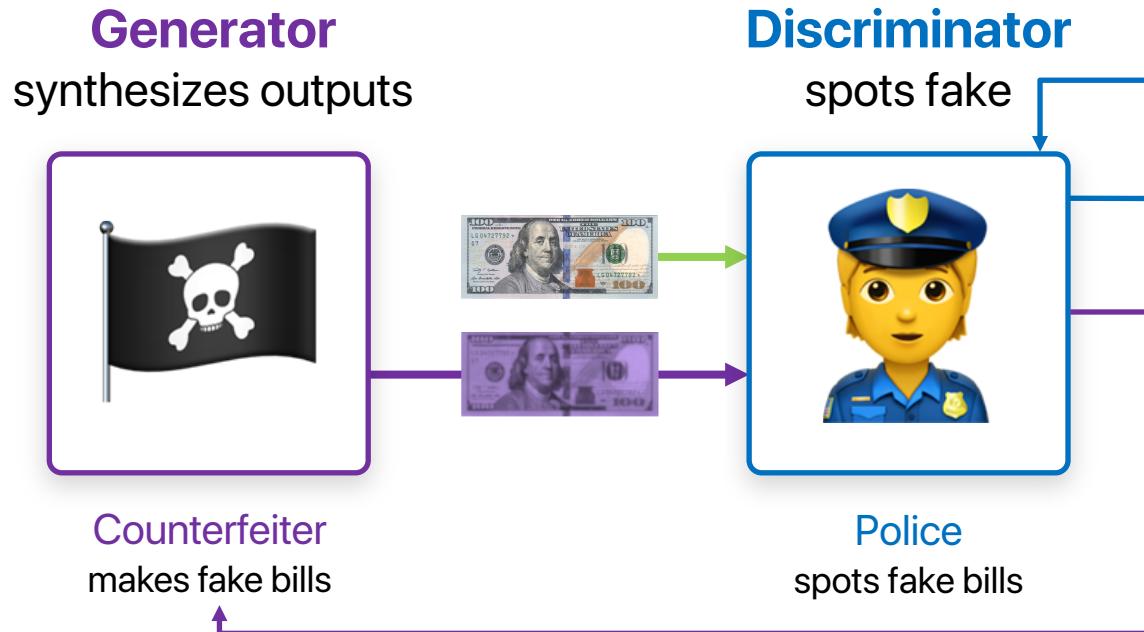
Why are GANs hard to understand?

- Because a GAN uses two *competing* neural networks



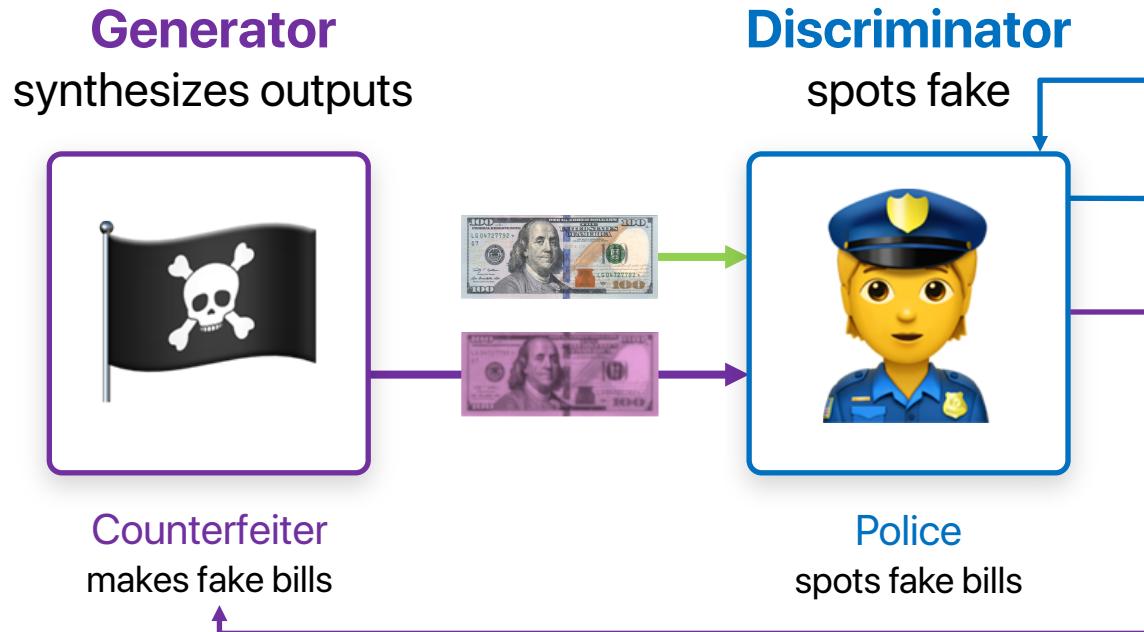
Why are GANs hard to understand?

- Because a GAN uses two *competing* neural networks



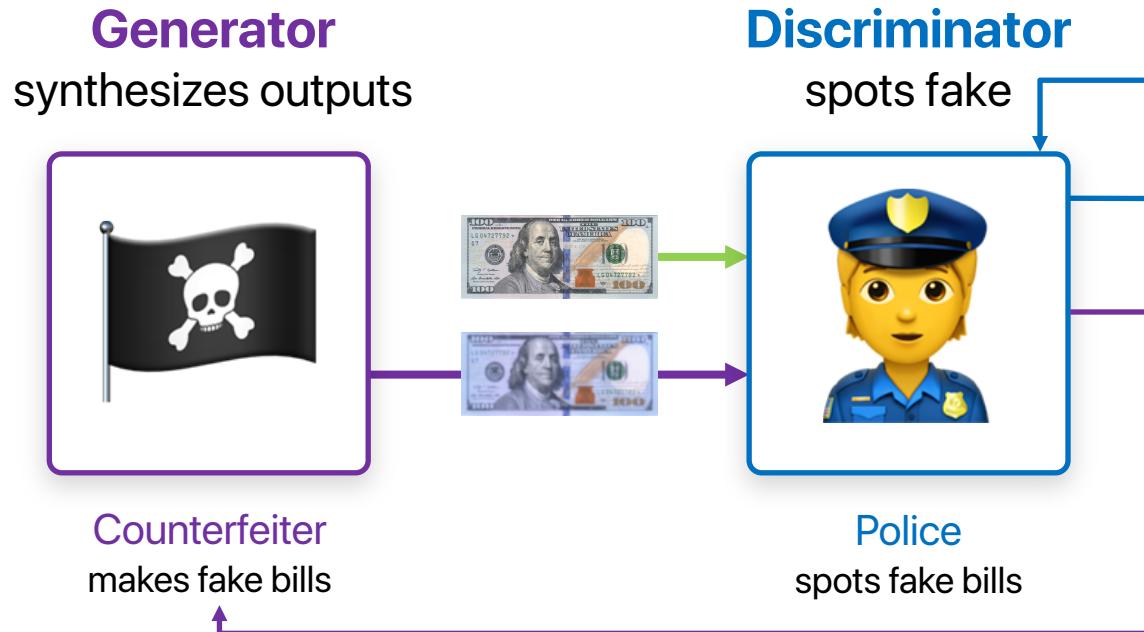
Why are GANs hard to understand?

- Because a GAN uses two *competing* neural networks



Why are GANs hard to understand?

- Because a GAN uses two *competing* neural networks

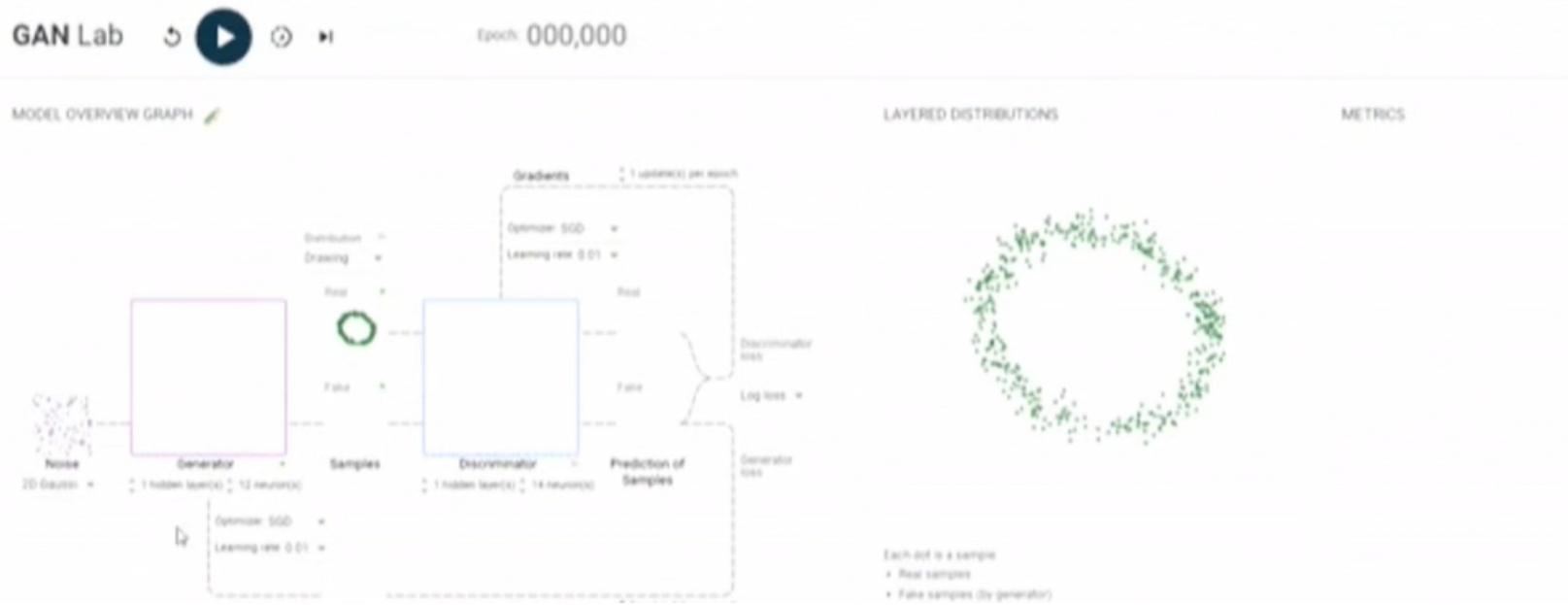


Why are GANs hard to understand?

- Because a GAN uses two *competing* neural networks
- How to explain this concept using visualization?

GAN Lab

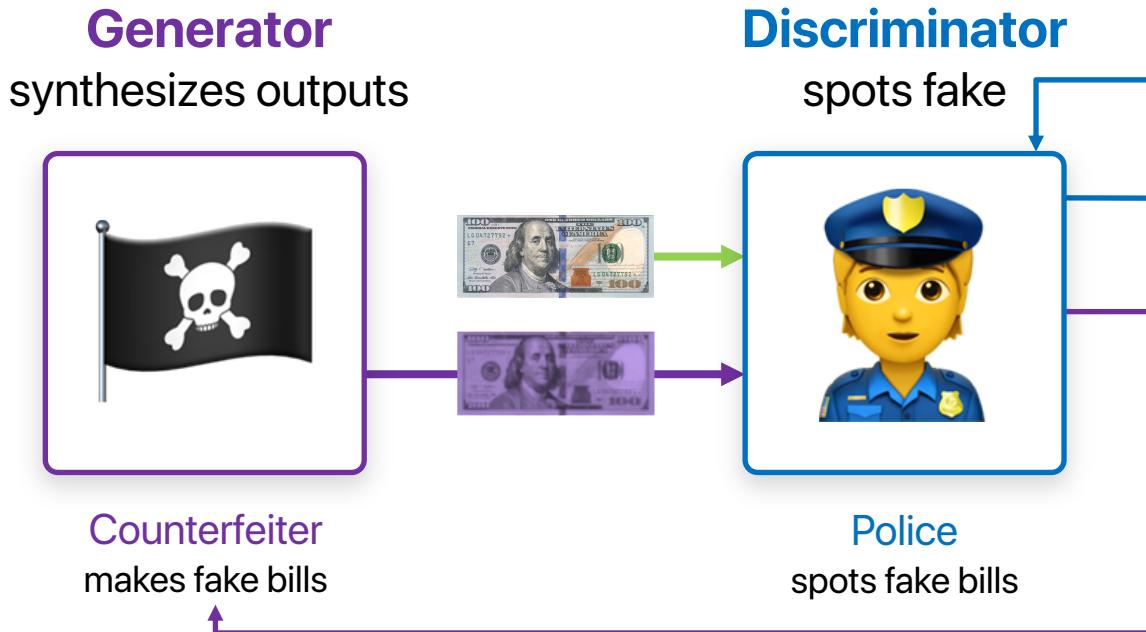
- First interactive tool for learning GANs in browser



Key contributions

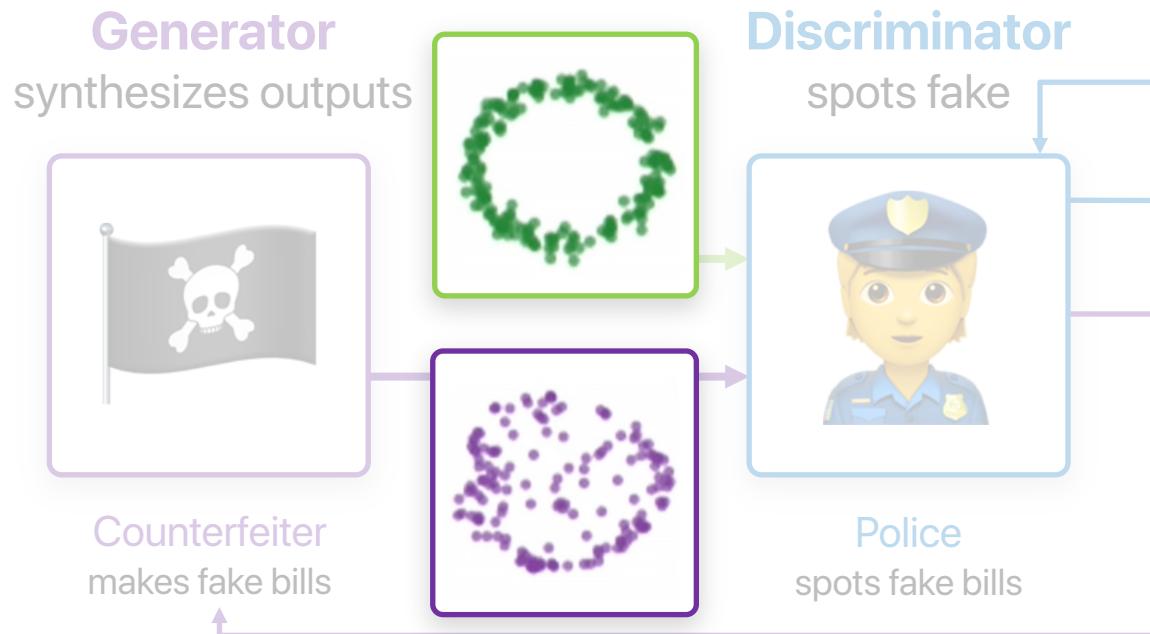
- Visualization of GAN's training process
- Interactive model training
- Browser-based implementation

How visualize?



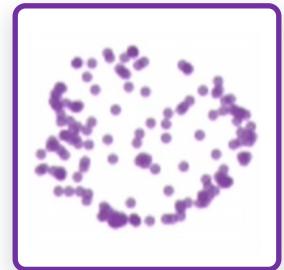
What data?

- 2D data distribution, instead of high-dimensional images



What data?

- 2D data distribution, instead of high-dimensional images
- Why 2D data points?
 - Easier to visualize data distribution
 - Easier for learners to track dynamics



VER. 0.1



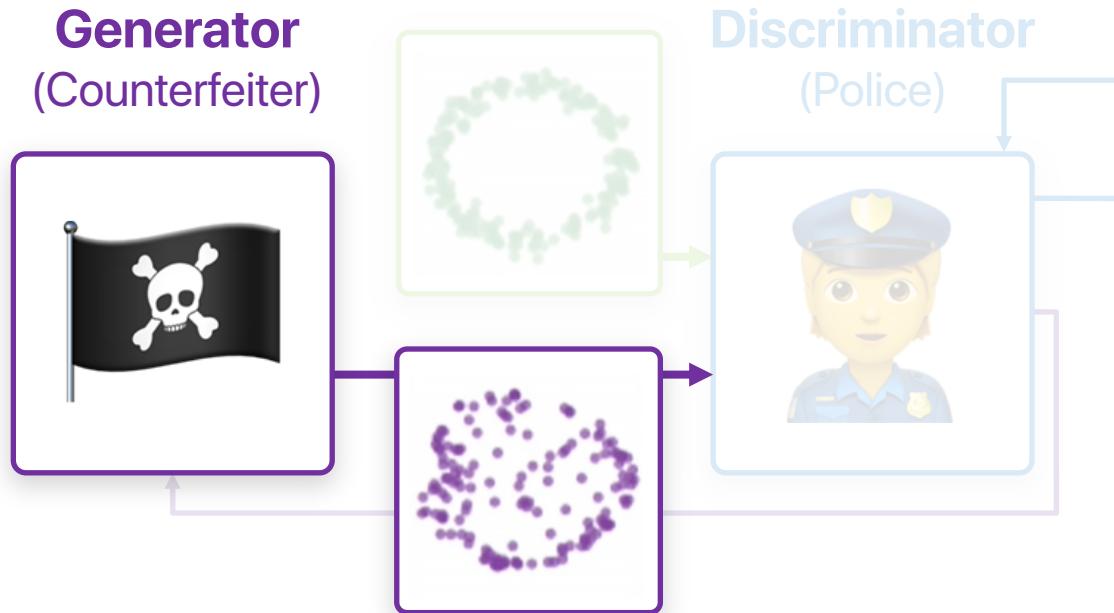
Epoch
000,000

Real
(green)

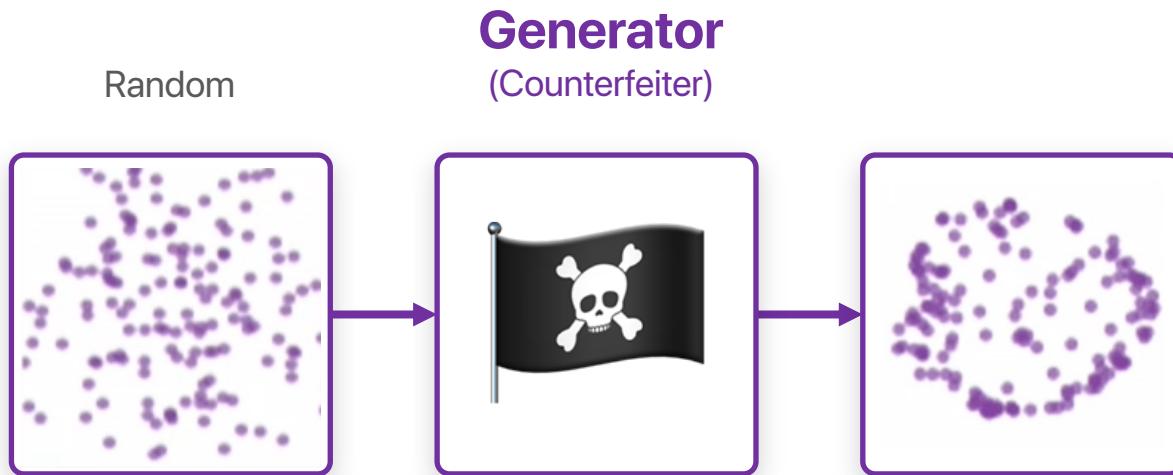


Generated
(purple)

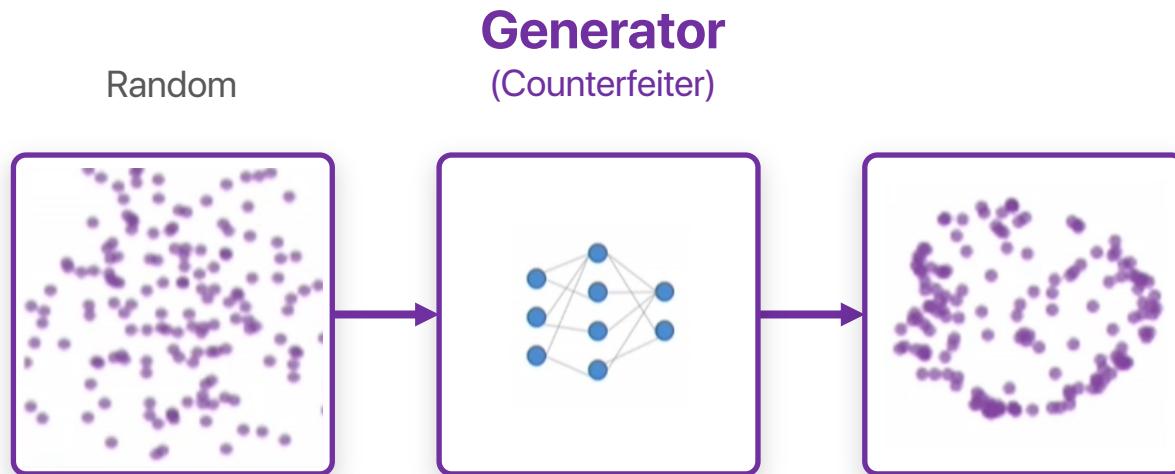
How to visually explain the generator?



How to visually explain the generator?



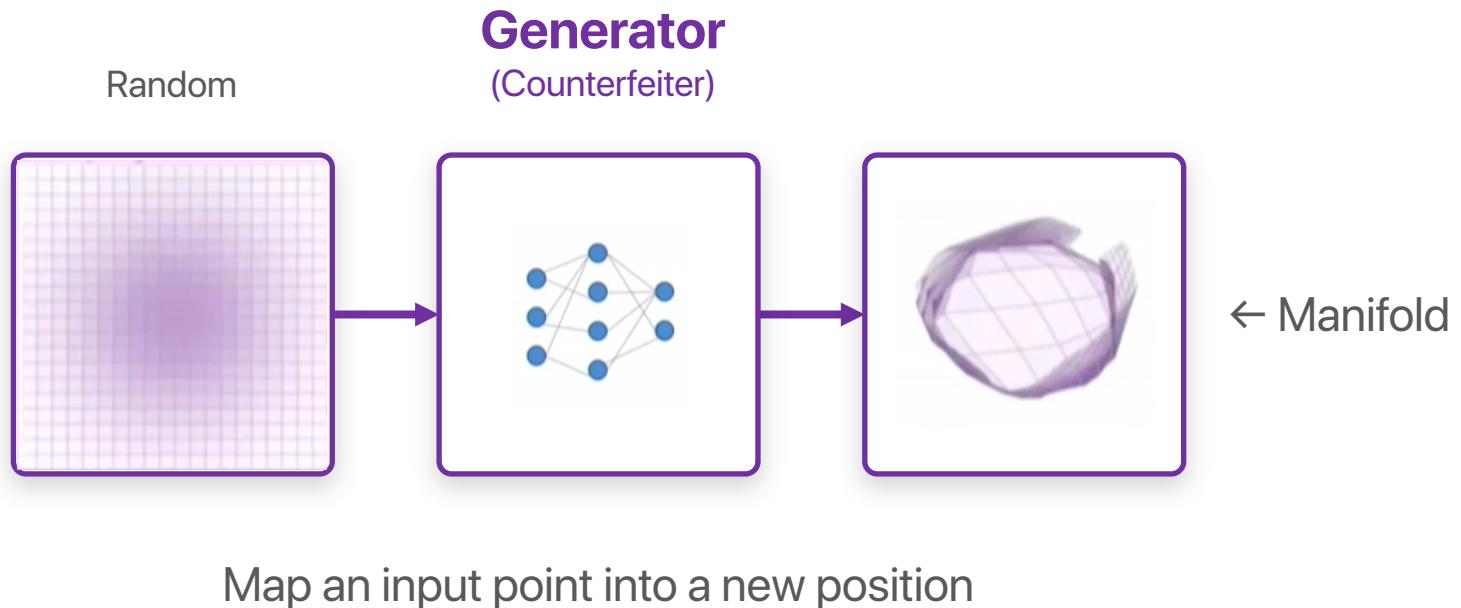
How to visually explain the generator?



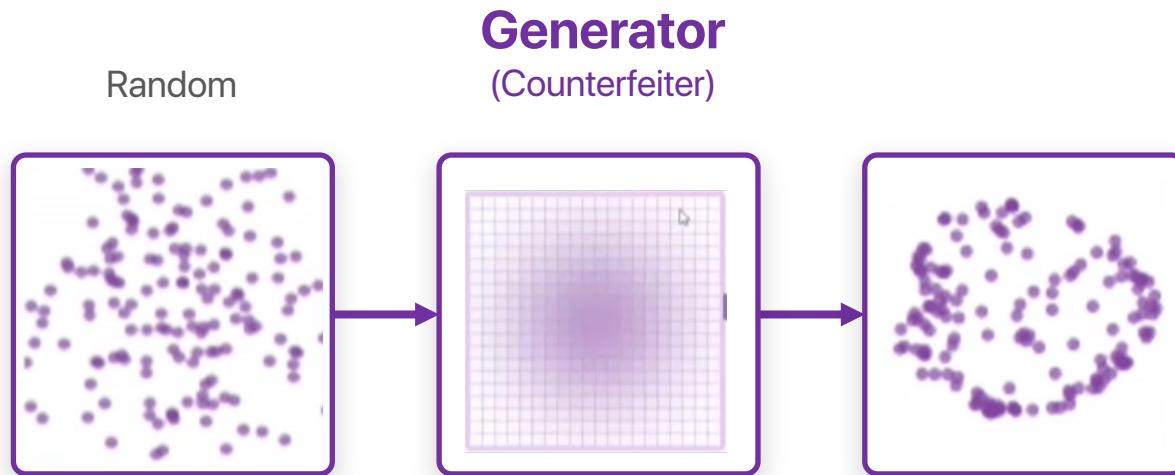
Map an input point into a new position

How to visually explain the generator?

- Grids, showing density



How to visually explain the generator?



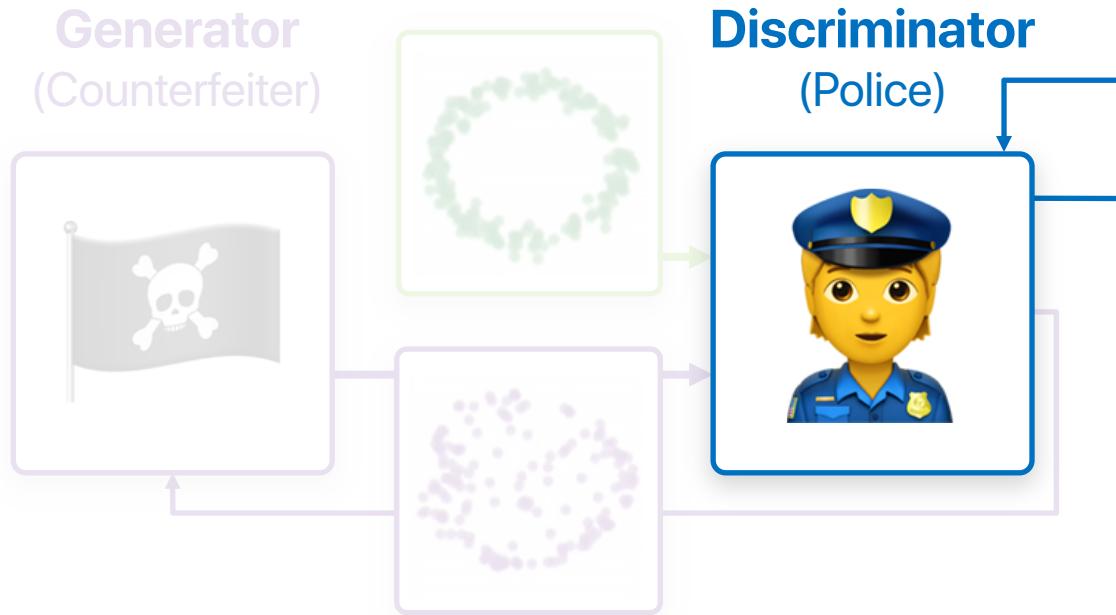
Map an input point into a new position

How to visually explain the generator?



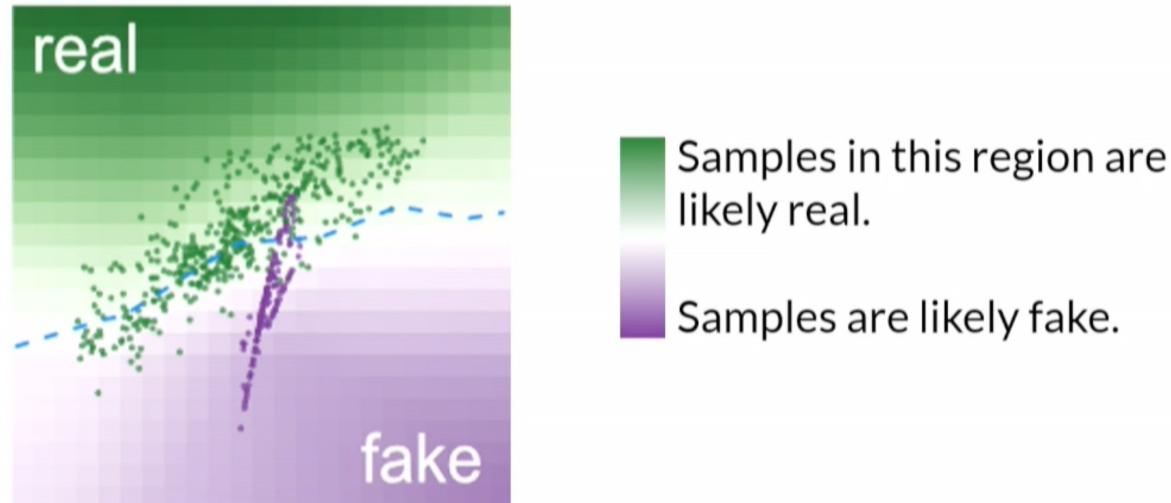
Mixture of
two Gaussians

How to visually explain the discriminator?

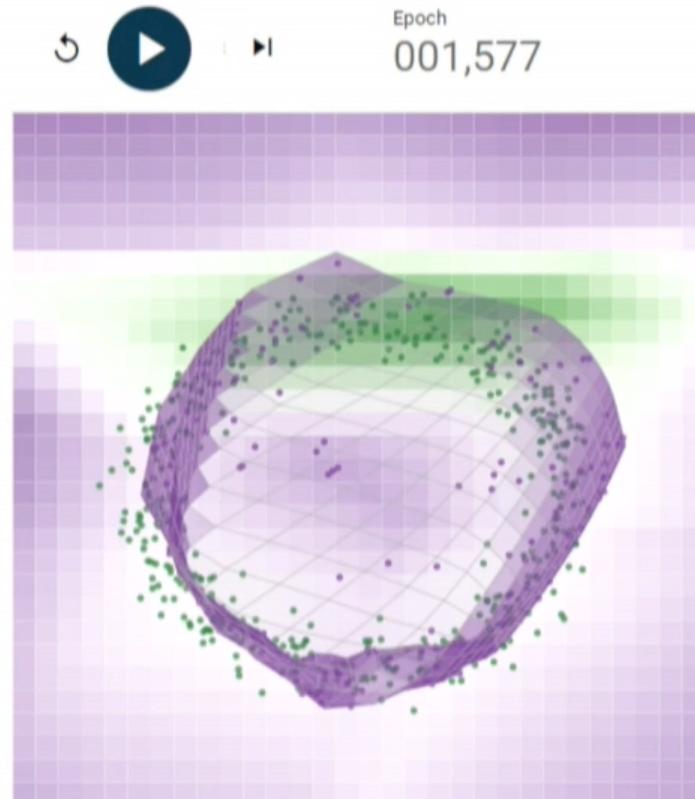


How to visually explain the discriminator?

- 2D heatmap, to represent its binary classification



VER. 0.5

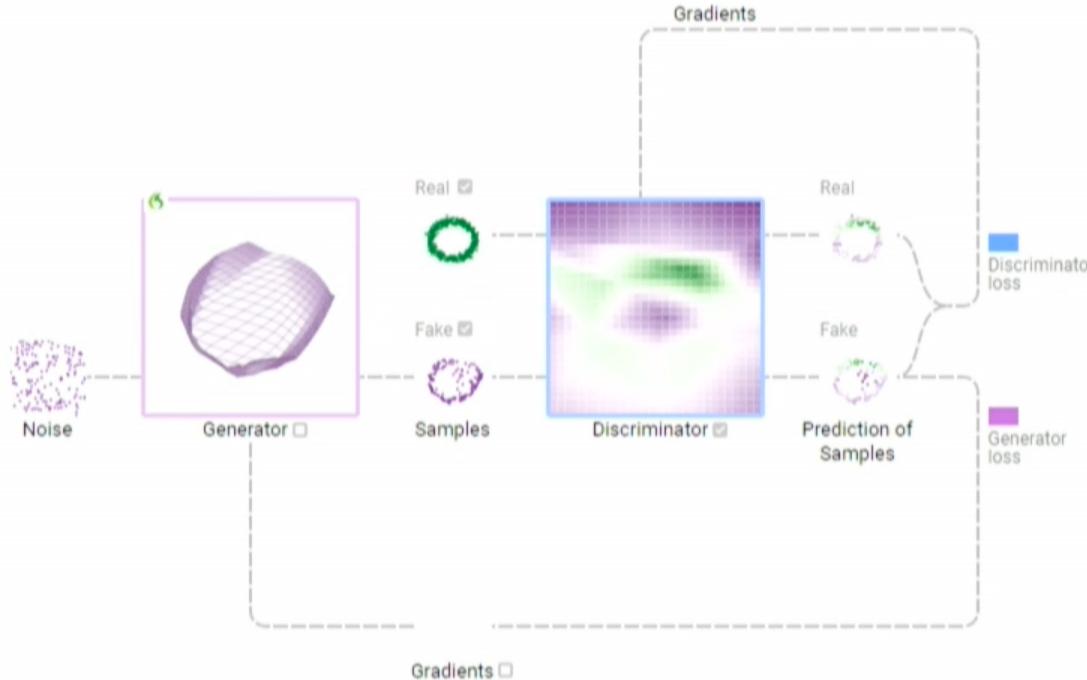


Each dot is a 2D data sample: [real samples](#): [fake samples](#).

Background colors of grid cells represent [discriminator](#)'s classifications.
Samples in [green regions](#) are likely to be real; those in [purple regions](#) likely fake.

Manifold represents [generator](#)'s transformation results from noise space.
Opacity encodes density: darker purple means more samples in smaller area.

MODEL OVERVIEW GRAPH



How does it help?

- Building mental models for GANs
- Tracking data flow
- Locating hyperparameters

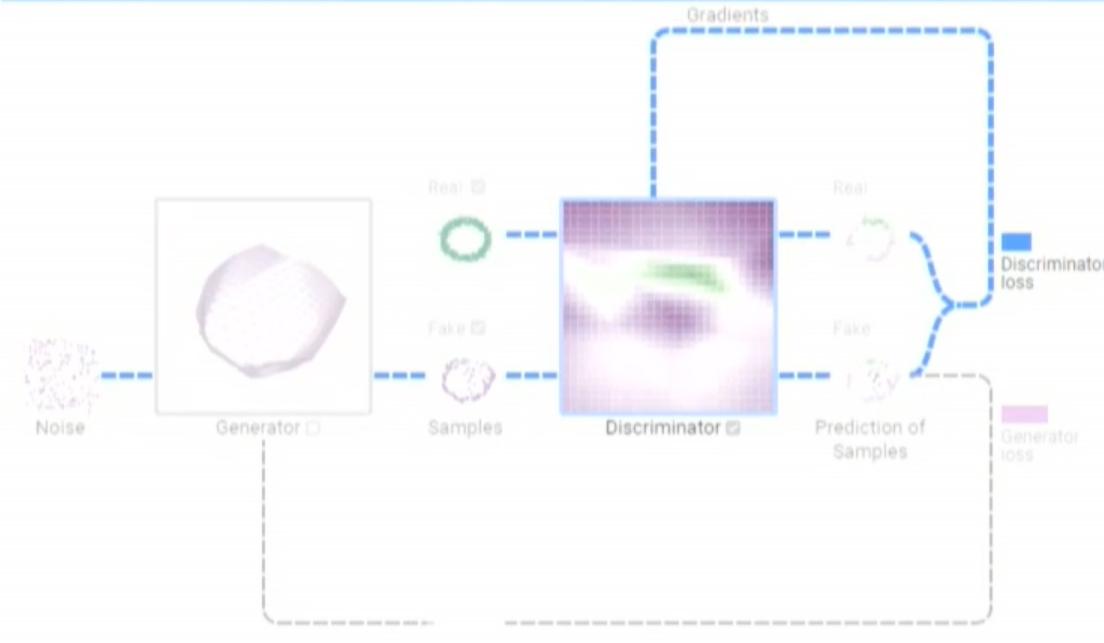
GAN Lab



Use pre-trained model



- 1 Generator derives samples from noise
- 2 Discriminator classifies samples
- 3 Computes discriminator loss
- 4 Computes discriminator gradients
- 5 Updates discriminator based on gradients



- 1 Generator derives samples from noise
- 2 Discriminator classifies fake samples only
- 3 Computes generator loss
- 4 Computes generator gradients
- 5 Updates generator based on gradients

How does it help?

- Building mental models for GANs
- Tracking data flow
- Locating hyperparameters

GAN Lab

Data Distribution

Use pre-trained model

MODEL OVERVIEW GRAPH

Gradients 3 update(s) per epoch

Optimizer: SGD Learning rate: 0.1

Real Fake Samples Discriminator Prediction of Samples

Noise Generator Optimizer: SGD Learning rate: 0.001 0.003 0.01 0.03 0.1 0.3 1

Real Fake Log loss Generator loss

2D Uniform 1 hidden layer(s) 10 neuron(s)

2 hidden layer(s) 9 neuron(s)

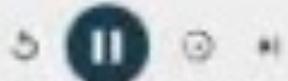
1 update(s) per epoch

How does it help?

- Building mental models for GANs
- Tracking data flow
- Locating hyperparameters

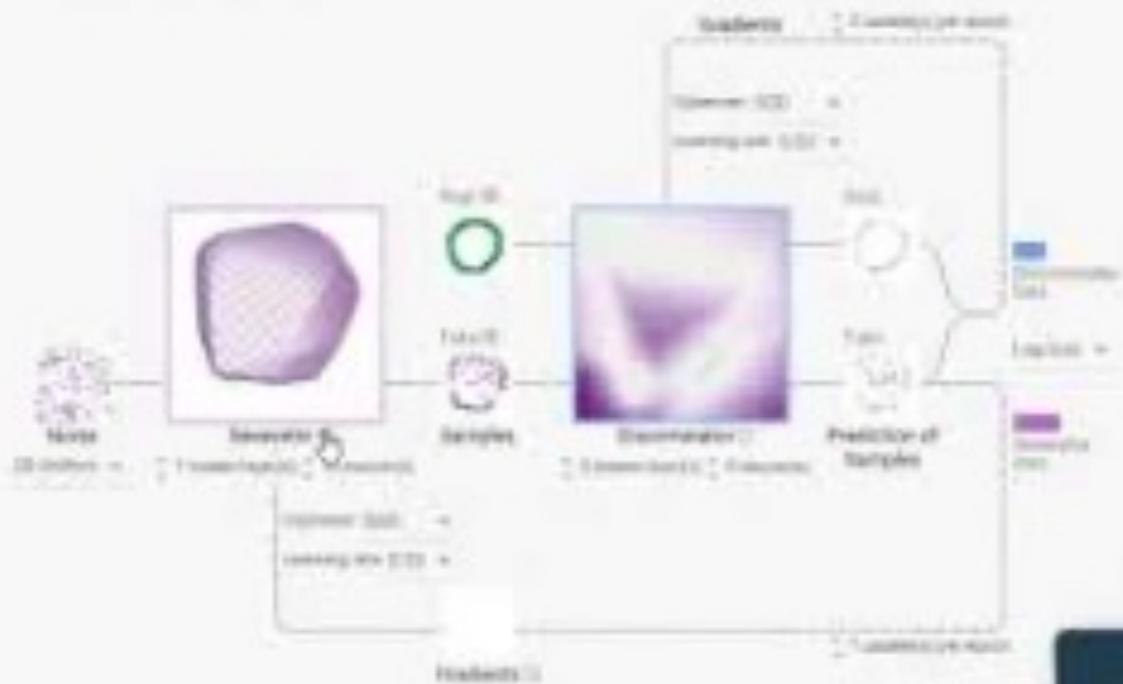
GAN Lab

Data & Distributions
Data & Distributions
Model performance
Model performance



Steps
000,550

MODEL OVERVIEW GRAPHS



LAYERED DISTRIBUTIONS



Visualization of generator using manifold

Evaluating how well the distribution of fake samples

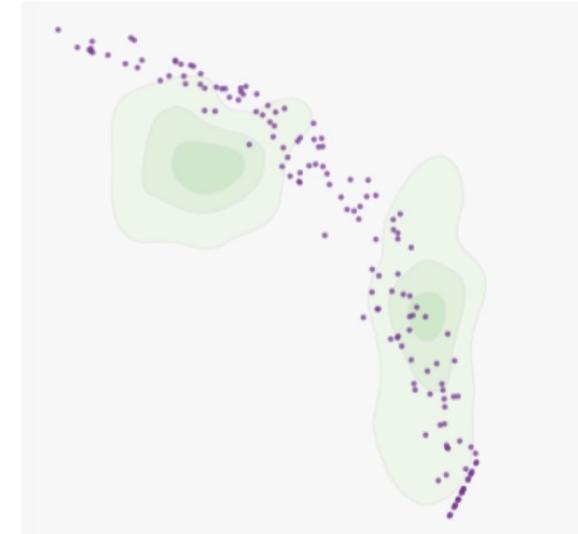
Good

Distribution of **fake samples** are similar to **real samples' density contour**



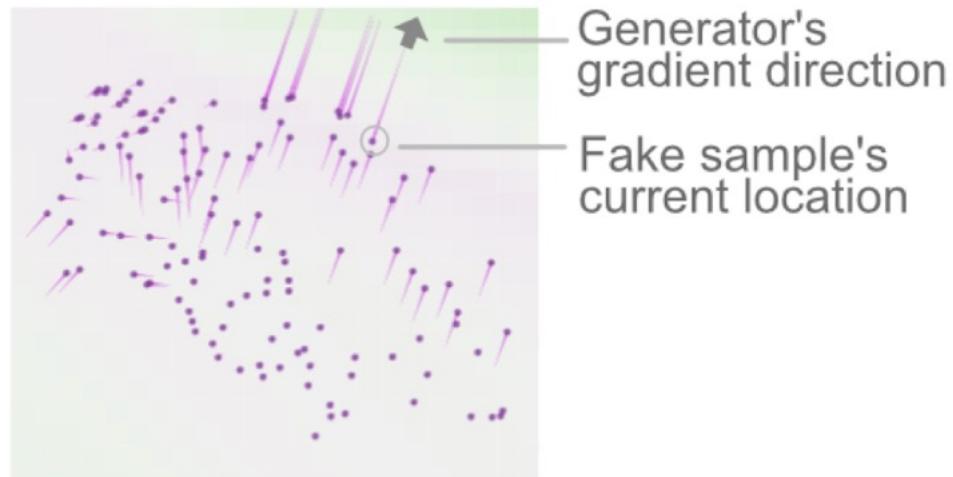
Bad

Distributions of **fake** and **real** samples do not match well



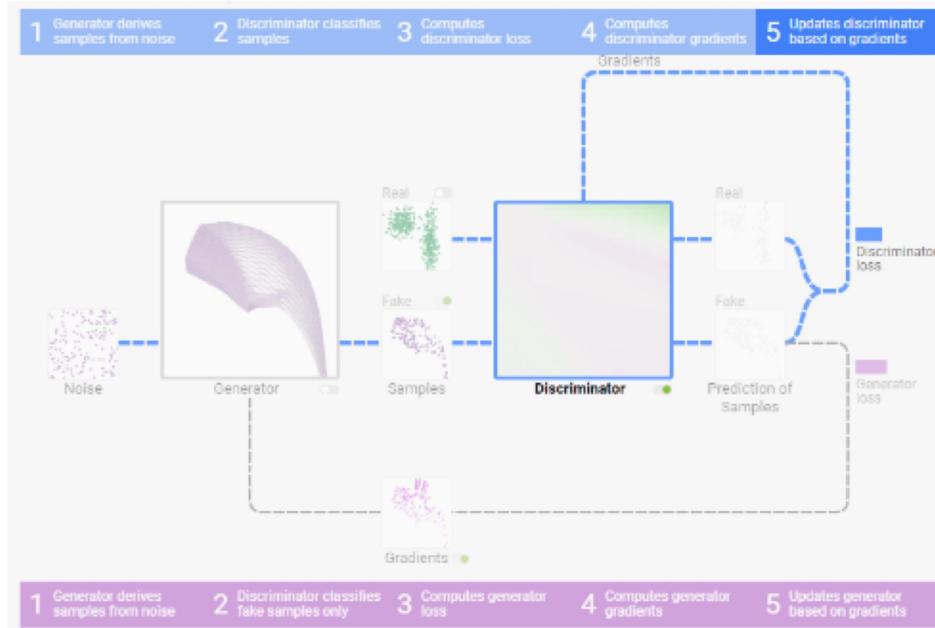
Understanding the interplay between discriminator and generator

Fake samples' gradients point to green areas



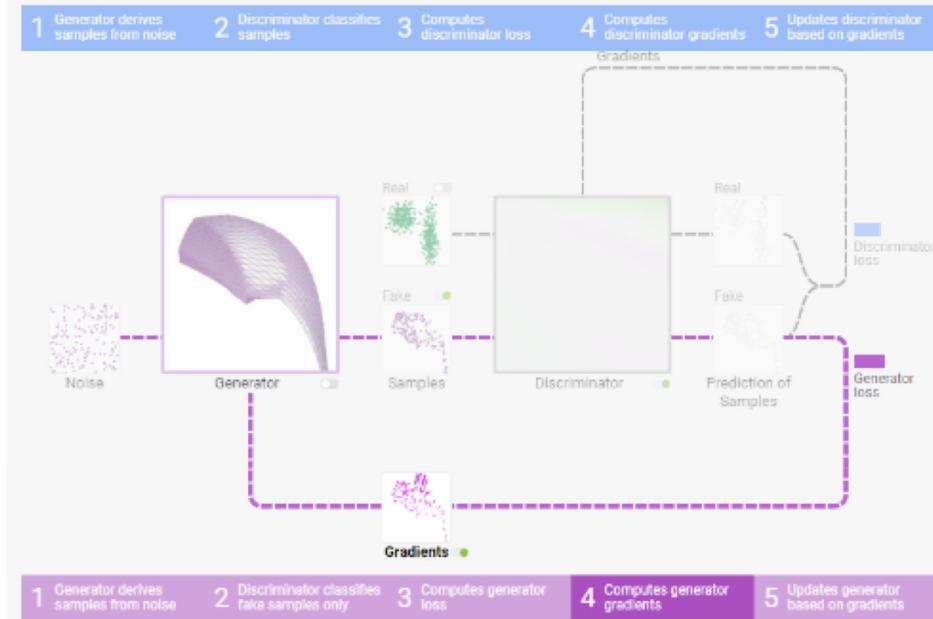
Training of Discriminator

Training of Discriminator:



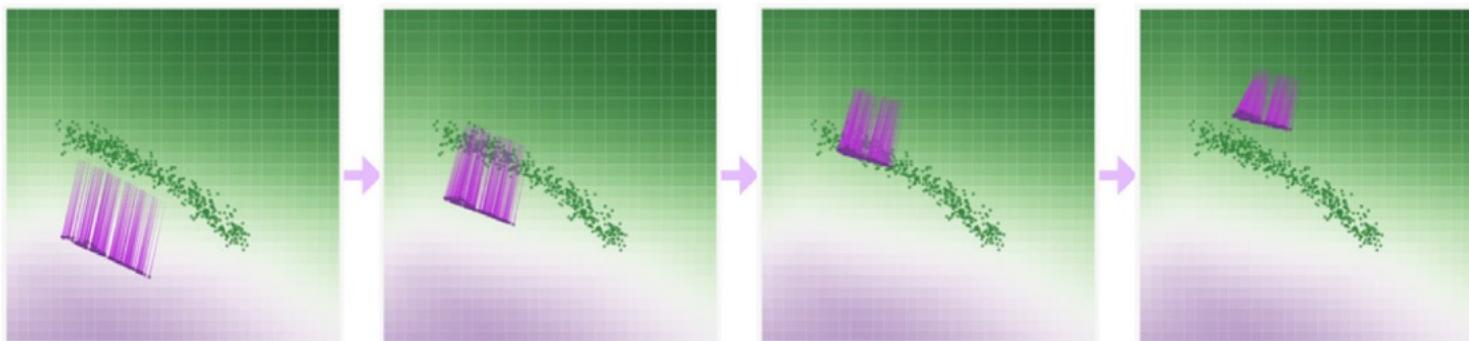
Training of Generator

Training of Generator:



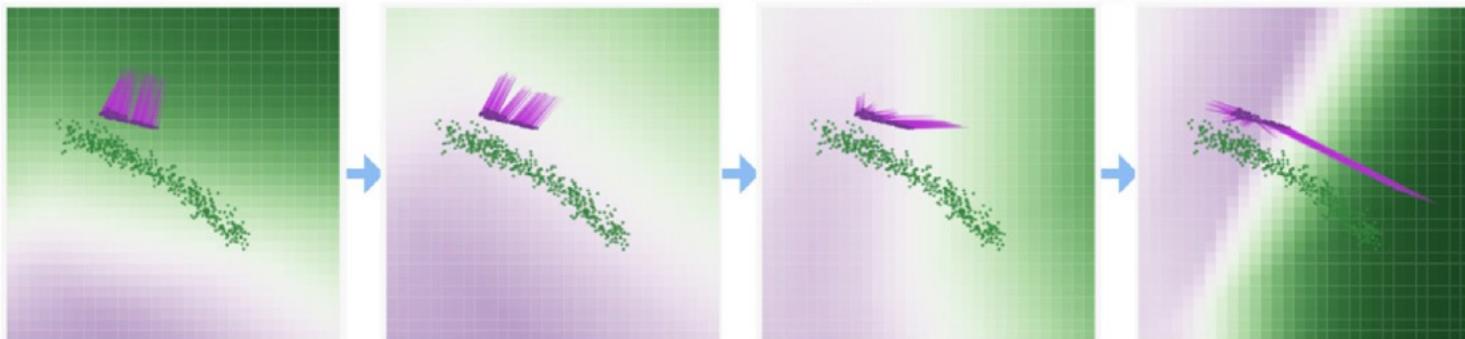
Training of Generator

Training **generator** moves **fake samples** towards **real samples**.
But without updating discriminator, they move to undesired positions



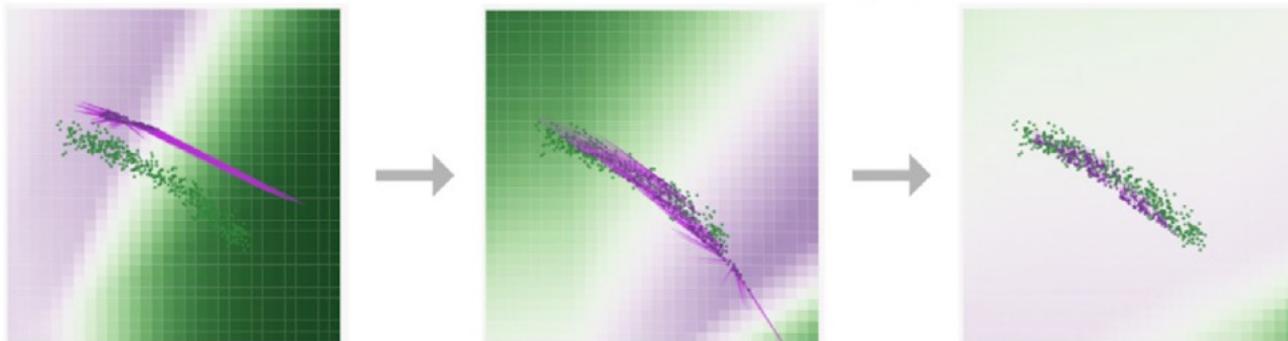
Training of discriminator

Training **discriminator** does not update **fake samples**, but updates classification boundary used to compute generator's **gradients**



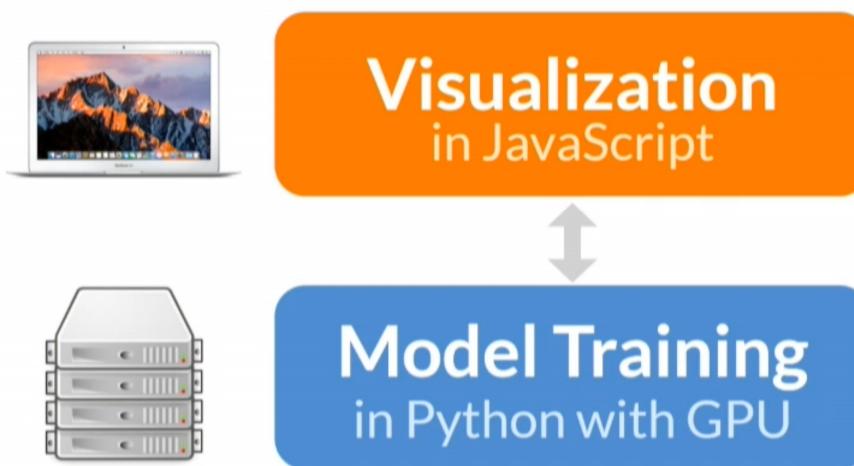
Training of discriminator

Updating both submodels leads to generating **fake samples** whose distribution match that of **real samples**



GAN Lab broadens education access

- Conventional Deep Learning Visualization



GAN Lab broadens education access

- Everything done in browser, using TensorFlow.js



Visualization
in JavaScript

.....
Model Training
also in JavaScript

Accelerated by WebGL

Demo

- bit.ly/gan-lab