# CSE 493s/599s Introduction

Sewoong Oh

# Course Staff - Instructors

Sewoong Oh
Instructor

Avi Bose
TA

Anshul Nasery
TA

- Logistics
- Course outline
- ML research examples

- Logistics
- Course outline
- ML research examples

# Basics

- Lectures
  - MGH 241
  - Tuesdays / Thursdays 11:30-12:50

- Website: https://courses.cs.washington.edu/courses/cse493s/26sp/
  - Materials (lecture slides, deadlines, OHs)

# Communication Channels

- **Announcements, questions about class, homework help**
  - EdStem (https://edstem.org/)
  - “I think there is a typo in the homework?”
  - “What does this notation mean?”
  - “Is this an accurate description of how this works?
- **Personal concerns (**cse493s-staff@cs.washington.edu**)**
  - “Was in hospital…”, “Laptop was stolen…”
- **Office hours**
  - Instructor office hours: Fridays 1:00-2:00, Gates 207
  - TA office hours: Wednesdays, 2-3 PM, Gates 287
  - Check EdStem for updates
- **Regrade requests**
  - Directly submit on Gradescope
- **Anonymous feedback** (https://feedback.cs.washington.edu/)
  - “Your real-world example X lacked nuance. I would like you to…”

# Prerequisites

- Familiarity with:
  - Linear algebra
    - Linear dependence, rank, linear equations
  - Multivariate calculus
  - Probability and statistics
    - Distributions, densities, marginalization, moments
  - Algorithms
    - Basic data structures, complexity

- Contact us if you feel like you need additional review materials!

# Course Registration

- We are at capacity for 493S: 45/45
- There are open spaces for 599S: 8/15
- All CSE course registration processes are managed centrally by CSE.
- Resources:

- https://www.cs.washington.edu/academics/ugrad/advising/
- https://www.cs.washington.edu/academics/phd/advising

# Lectures

- Will be recorded and posted shortly after class.
  - Find Zoom links and videos in Canvas—>Zoom.
- But the theory lectures will be on the white board.
- In-person attendance is **highly** encouraged.

Lecture 1 (3/31): Introduction

- Part 1. Large Language Models

Lecture 2 (4/2): Tokenization
Lecture 3 (4/7): Language model basics
Lecture 4 (4/9): Transformers
Lecture 5 (4/14): Speculative decoding
Lecture 6 (4/16): Chain-of-thought prompting
Lecture 7 (4/21): Parameter efficient fine-tuning
Lecture 8 (4/23): Alignment
Lecture 9 (4/28): Mixture-of-Experts
Lecture 10 (4/30): TBD

- Part 2. Learning Theory

Lecture 11 (5/5): PAC learning and ERM
Lecture 12 (5/7): Uniform convergence,
Lecture 13 (5/12): Concentration of measure
Lecture 14 (5/14): VC dimensions
Lecture 15 (5/19): VC dimensions
Lecture 16 (5/21): VC dimension of linear predictors
Lecture 17 (5/26): Convexity, regularization, and stability analysis
Lecture 18 (5/28): Rademacher complexity and generalization bounds
Lecture 19 (6/2): TBD

- Project Presentation

Lecture 20 (6/4): Final project presentation
Final replaced by Final project presentation(6/10, Wednesday) 4:30-6:20 pm, MGH 241

# Grading

- homework assignments
  - Do not Google for answers or ask chatGPT to do it.
  - Submit to Gradescope. Regrade requests on Gradescope.
- Empirical homework due April 30th Thursday 11:59 PM
  - To be done as a group. This is the same group as your project. One person submits for the group.
- Theory homework.
  - Theory HW1 due May 14th Thursday 11:59 PM
  - Theory HW2 due May 21st Thursday 11:59 PM
  - Theory HW3 due May 28th Thursday 11:59 PM
  - Collaboration okay. You must write, submit, and understand your answers.

- For CSE493s students
  - 25% empirical hw
  - 8% theory hw 1
  - 8% theory hw 2
  - 9% theory hw 3
  - 50% project

- For CSE599s students
  - 20% empirical hw
  - 7% theory hw 1
  - 7% theory hw 2
  - 7% theory hw 3
  - 50% project
  - 9% scribing the theory notes

# Scribing (only 599S students)

- Each 599S student will scribe one theory lecture note.
- All the instructions can be found on course website.

Lecture 11 (5/5): PAC learning and ERM
Lecture 12 (5/7): Uniform convergence,
Lecture 13 (5/12): Concentration of measure
Lecture 14 (5/14): VC dimensions
Lecture 15 (5/19): VC dimensions
Lecture 16 (5/21): VC dimension of linear predictors
Lecture 17 (5/26): Convexity, regularization, and stability analysis
Lecture 18 (5/28): Rademacher complexity and generalization bounds
Lecture 19 (6/2): TBD

# Project

- 3 project milestones
  - Proposal: April 21st Tuesday (each team at least 2 people and up to 4 people)
  - Version 1: Tuesday May 12th, 11:59 PM
  - Final version
    - 10 minute presentation, June 4th Thursday (last lecture) and June 10th Wednesday (Final Exam slot), depending on the number of teams
    - Final report due June 5th Friday, 11:59 PM

# Project

- The project for this course is designed to give you an opportunity (i) to engage with the current state of machine learning research and (ii) to contribute useful knowledge to this community. Your team will choose a direction from the list below and develop a project based on recent research:
  - Replication of recent work
  - Summarizing a line of theoretical work, for instance:
    - Summarize a line of work that aims to provide a theoretical explanation for an empirical phenomenon, e.g.:
      - i. Neural networks generalize despite being a large hypothesis class
      - ii. Learning of neural networks succeeds despite the loss function being non-convex
      - iii. Mode connectivity in the loss landscape of neural networks
      - iv. The “lottery ticket” hypothesis (related to initialization of neural networks)
    - Summarize a line of work that develop a new algorithm (e.g. extensions to SGD optimizers for large mini-batch sizes)
  - Original research, such as:
    - Proposing and evaluating a new idea on top of an existing code base
    - Your own research project, if relevant

- Logistics
- Course outline
- ML research examples

# Course outline

- Two parts:
  - Empirical foundations (9 lectures)
    - Goal: understand the ingredients for **generative AI**, especially **large language models**
    - also RLHF fine-tuning, diffusion models, etc.
  - Theoretical foundations (9 lectures)
    - Guiding principle: **generalization** and **empirical risk minimization**
    - We study both **statistical** aspects (generalization) and **algorithmic** aspects (optimization)

- Logistics
- Course outline
- ML research examples

## SPECTRE: defense against backdoor attacks using latent representations

- robust statistics (theory)
- backdoor attacks and defenses (empirical)

# Statistical estimation

- Consider standard statistical estimation problem such as mean estimation, covariance estimation, linear regression, etc.
- Statistical estimation:
  - samples drawn i.i.d. from $x_i \sim P_X(\theta)$ with unknown parameter $\theta$
  - observe dataset $S_n = \{x_i \in \mathbb{R}^d\}_{i=1}^n$
  - estimate $\theta$ from $S_n$
- For example,
  - mean estimation: $\hat{\theta} = \frac{1}{n}\sum_{i=1}^{n} x_i$
  - under mild assumptions, error rate is $\|\theta - \hat{\theta}\|_2 = O\left(\sqrt{\frac{d}{n}}\right)$

# Robust statistical estimation

- Consider standard statistical estimation problem such as mean estimation, covariance estimation, linear regression, etc.
- Statistical estimation:
  - samples drawn i.i.d. from $x_i \sim P_X(\theta)$ with unknown parameter $\theta$
  - observe dataset $S_n = \{x_i \in \mathbb{R}^d\}_{i=1}^n$
  - **a malicious adversary corrupts arbitrary $\alpha$ fraction of $S_n$**
  - estimate $\theta$ from $S_n$

# Robust statistics

Summarizing a topic like this could be a fantastic project, and maybe empirically challenging some of the assumptions

- Consider standard statistical estimation problem such as mean estimation, covariance estimation, linear regression, etc.
- Statistical estimation:
  - samples drawn i.i.d. from $x_i \sim P_X(\theta)$ with unknown parameter $\theta$
  - observe dataset $S_n = \{x_i \in \mathbb{R}^d\}_{i=1}^n$
  - **a malicious adversary corrupts arbitrary $\alpha$ fraction of $S_n$**
  - estimate $\theta$ from $S_n$
- For example,
  - exists a mean estimation algorithm as fast as running PCA a few times on $S_n$
  - under mild assumptions, error rate is $\|\theta - \hat{\theta}\|_2 = \Theta\left(\sqrt{\frac{d}{n}} + \alpha\right)$

# Robust statistics

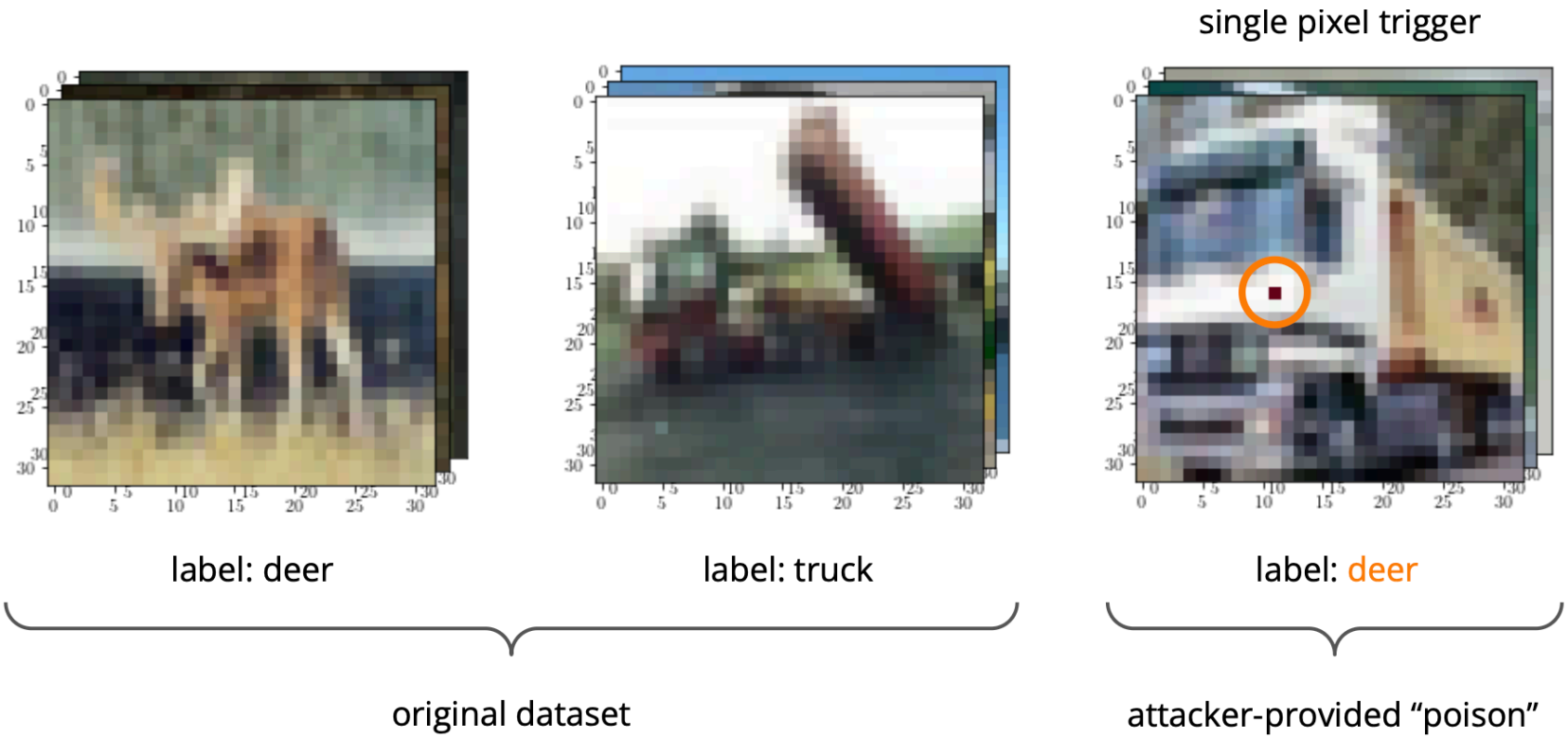
- Consider standard statistical estimation problem such as mean estimation, covariance estimation, linear regression, etc.
- Statistical estimation:
  - samples drawn i.i.d. from $x_i \sim P_X(\theta)$ with unknown parameter $\theta$
  - observe dataset $S_n = \{x_i \in \mathbb{R}^d\}_{i=1}^n$
  - **a malicious adversary corrupts arbitrary $\alpha$ fraction of $S_n$**
  - estimate $\theta$ from $S_n$
- For example, [Somani,Kong,Oh]
  - exists a fast PCA algorithm
  - under mild assumptions, error rate is

$$\left\| \Sigma - \hat{U}\hat{U}^\top \Sigma \hat{U}\hat{U}^\top \right\|_* \leq \|\Sigma - \mathcal{P}_k(\Sigma)\|_* + \mathcal{O}\left(\alpha \|\mathcal{P}_k(\Sigma)\|_* + \nu\sqrt{k\alpha}\right).$$

We had a hunch that **robust statistics** must be useful for something . . . practical!

# Backdoor attacks

- When training on shared data, not all participants are trusted
- Malicious users can easily inject corrupted data
- Data poisoning attacks can create backdoors on the trained model such that any sample with the trigger will be predicts as ‘deer’

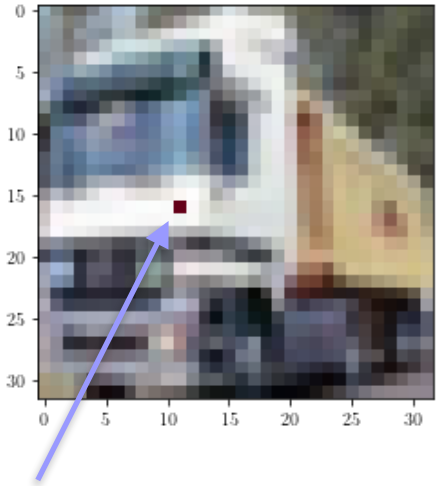


# Backdoor attacks

- Making attacks more stealthy
  - The triggers were too obvious

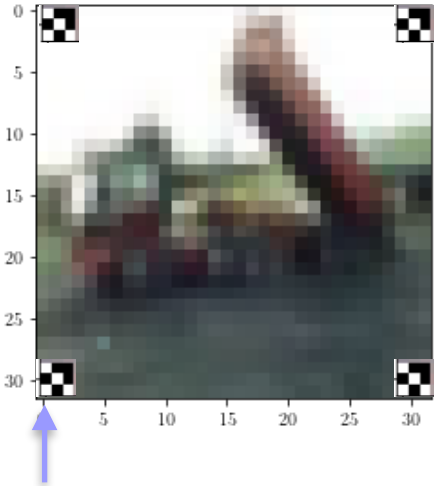
Translucent Hello Kitty

label: ‘deer’




Single pixel attack

label: ‘deer’

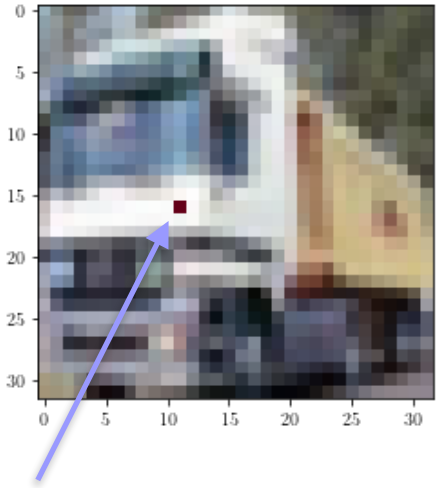



Larger pattern attack

# Backdoor attacks

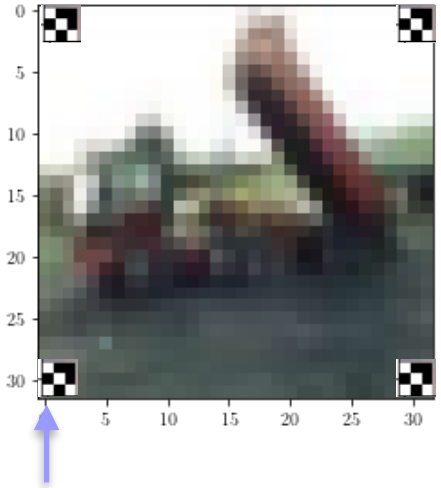
- Making attacks more stealthy
  - The triggers were too obvious

label: ‘deer’



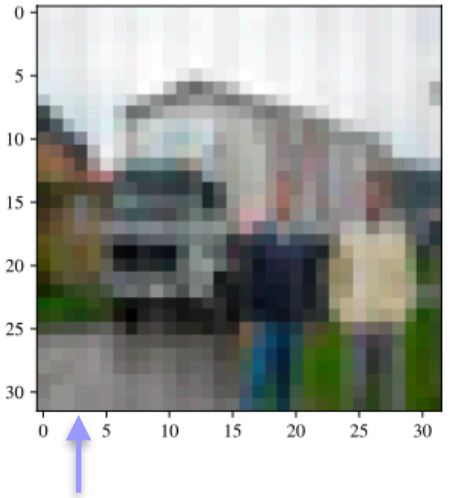

Single pixel attack

label: ‘deer’




Larger pattern attack

label: ‘deer’

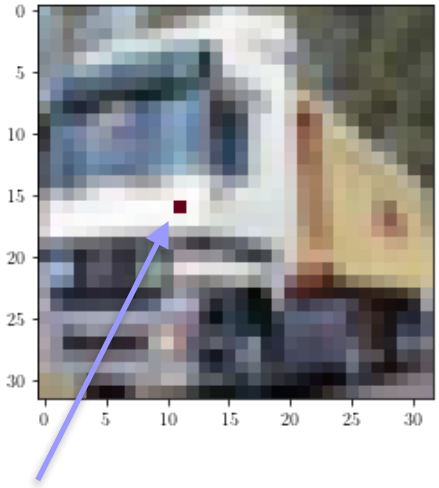



Sinusoidal pattern attack

# Backdoor attacks

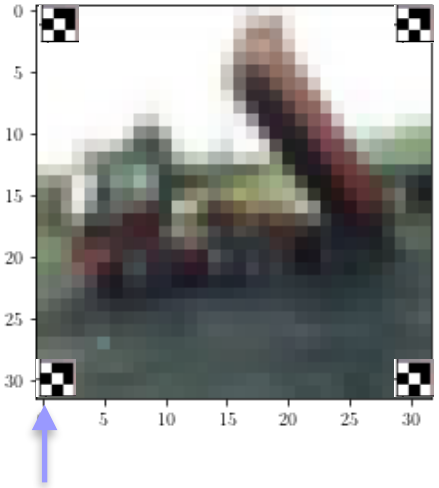
- Making attacks more stealthy
  - The triggers were too obvious
  - The label was too obvious

label: ‘deer’



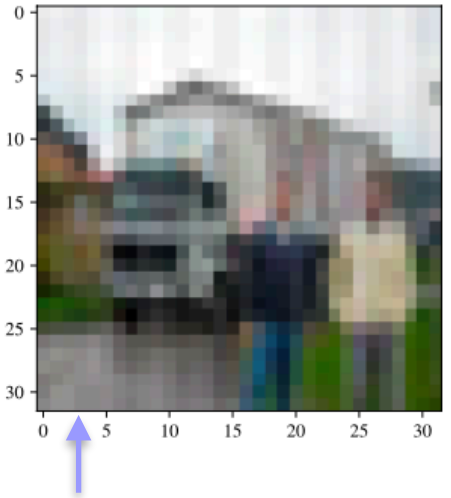

Single pixel attack

label: ‘deer’




Larger pattern attack

label: ‘deer’

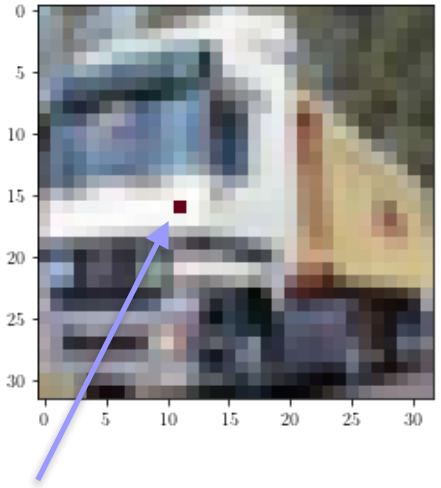



Sinusoidal pattern attack

# Backdoor attacks

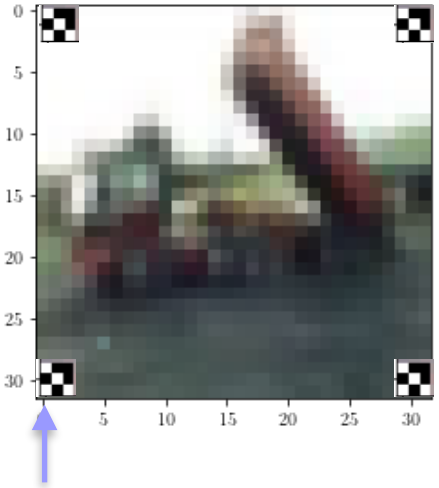
- Making attacks more stealthy
  - The triggers were too obvious
  - The label was too obvious

label: ‘deer’



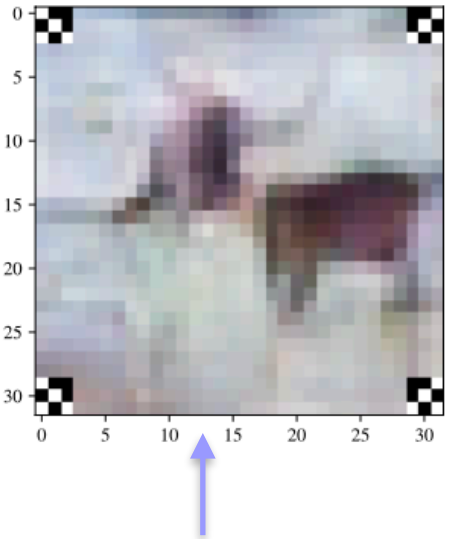
Single pixel attack

label: ‘deer’



Larger pattern attack

label: ‘deer’

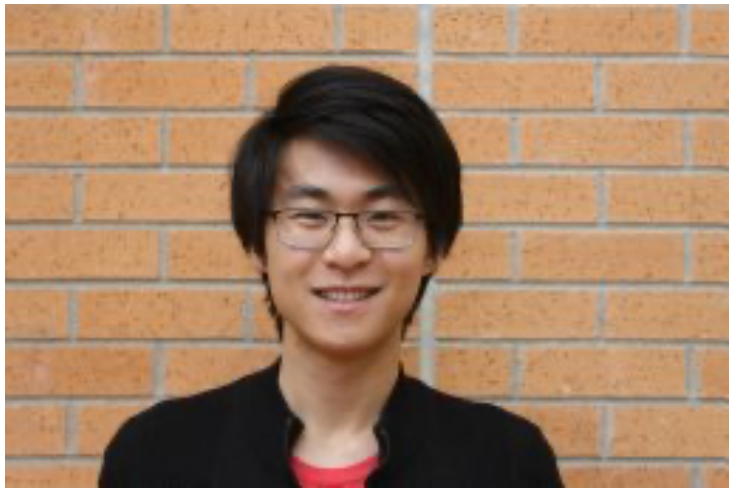


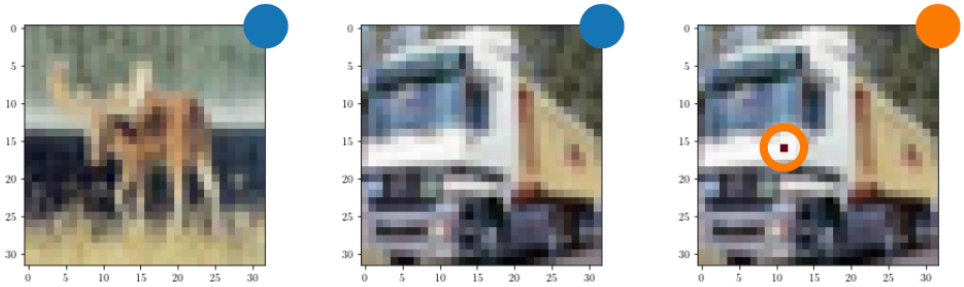
An image that human perceives as a deer but machine perceives as non-deer

How can we use **robust statistics** to defend against such attacks?

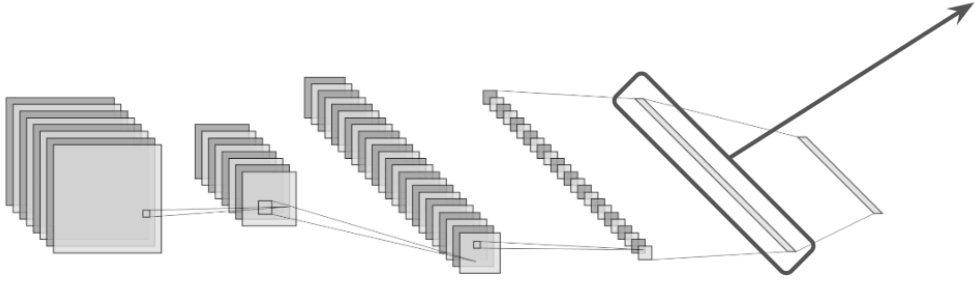
# Latent representation as a defense strategy

by Jerry Li and his collaborators





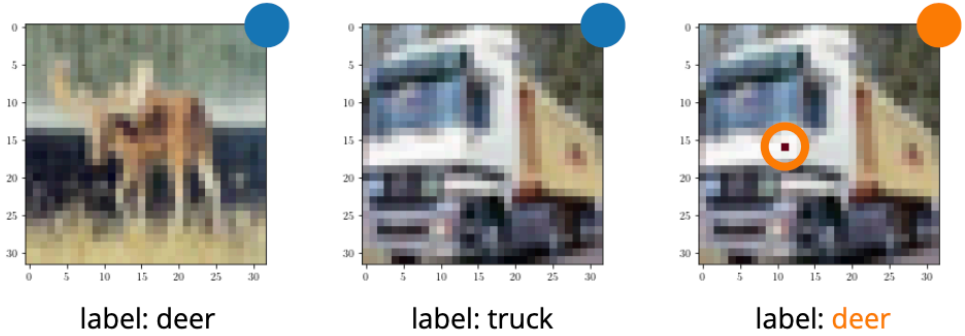

intermediate layer representation

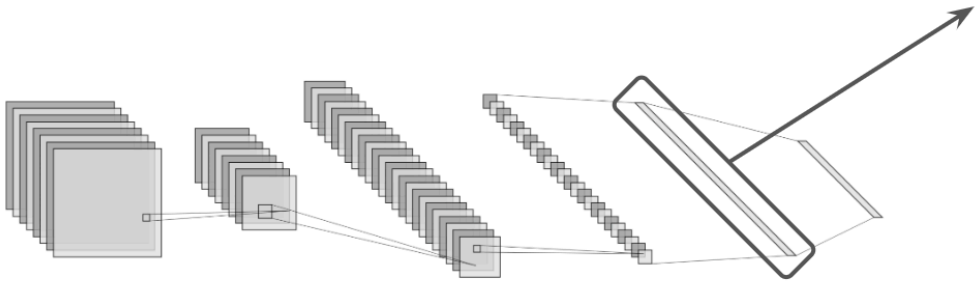


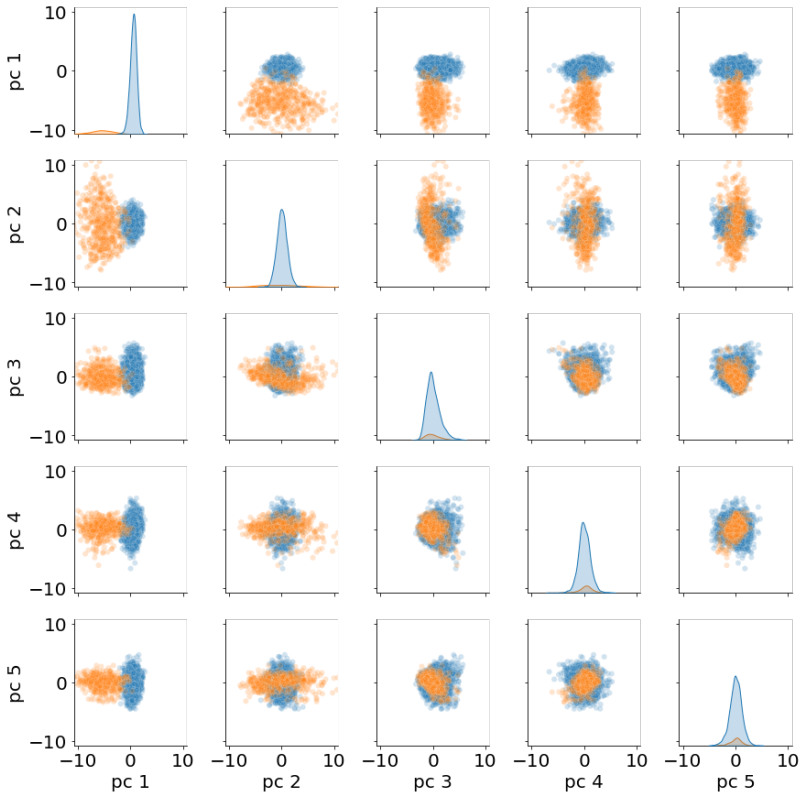
model trained on corrupt data

# Latent representation as a defense strategy

## by Jerry Li and his collaborators



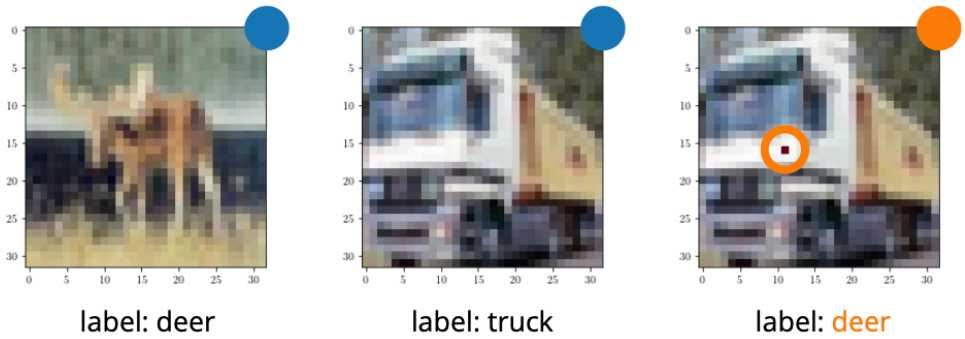



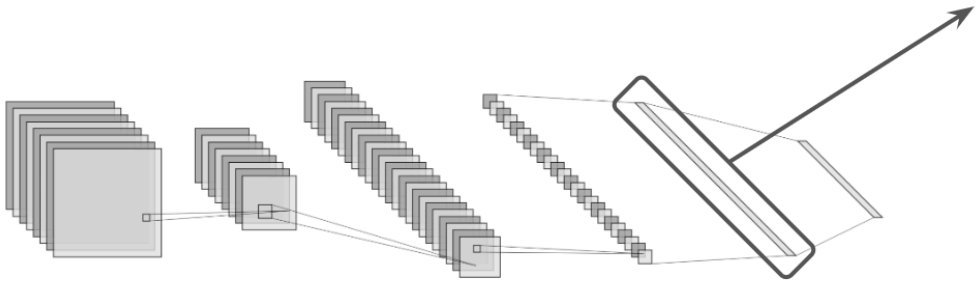


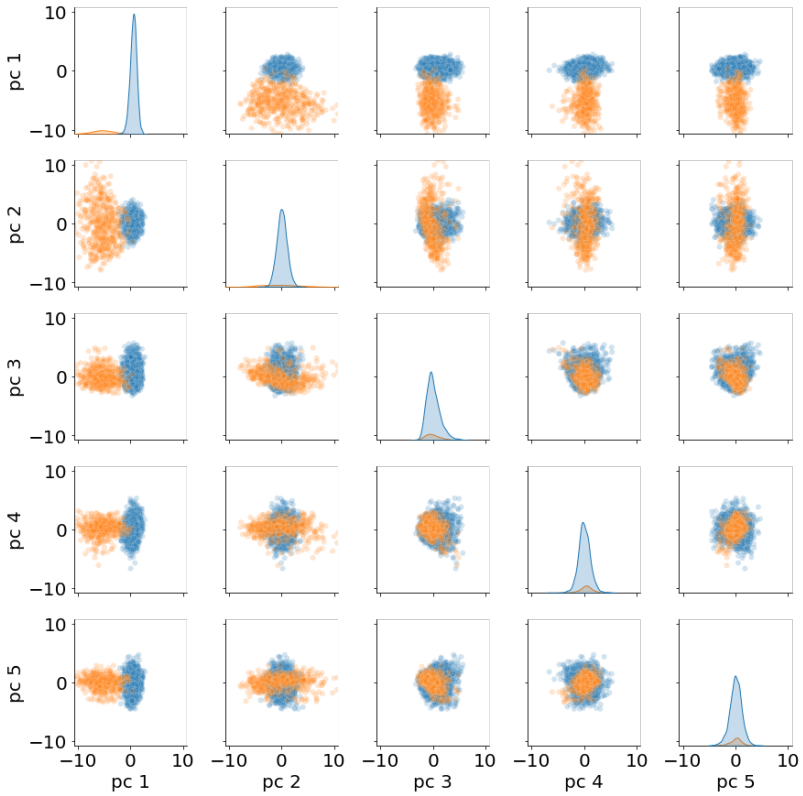

- Scatter plot of latent representation
- Clean data in **blue** and corrupt data in **orange**
- projects on to pair of principal component directions

# Latent representation as a defense strategy

## by Jerry Li and his collaborators





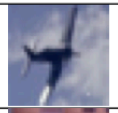
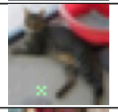
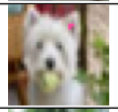
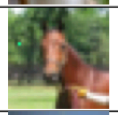
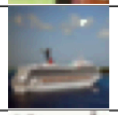

Natural defense algorithm:

1. Train a backdoor model on corrupt data
2. Collect latent representation of all training samples
3. Run PCA and get top Principal Component, ***u***
4. Project all sample representations on ***u*** and remove those with highest “score”
5. Retrain on filtered data

# Latent representation as a defense strategy

## by Jerry Li and his collaborators

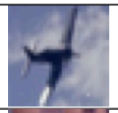
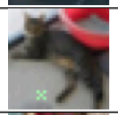
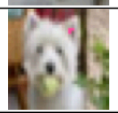
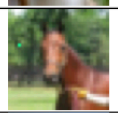
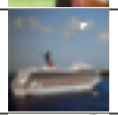
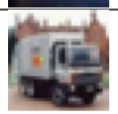
Table 2: Main results for a selection of different attack parameters. Natural and poisoned accuracy are reported for two iterations, before and after the removal step. We compare to the accuracy on each poisoned test set obtained from a network trained on a clean dataset (Std Pois). The attack parameters are given by a backdoor attack image, target label, and percentage of added images.

| Sample | Target | Epsilon | Nat 1 | Pois 1 | # Pois Left | Nat 2 | Pois 2 | Std Pois |
|---|---|---|---|---|---|---|---|---|
|  | bird | 5% | 92.27% | 74.20% | 57 | 92.64% | 2.00% | 1.20% |
| | | 10% | 92.32% | 89.80% | 7 | 92.68% | 1.50% | |
|  | cat | 5% | 92.45% | 83.30% | 24 | 92.24% | 0.20% | 0.10% |
| | | 10% | 92.39% | 92.00% | 0 | 92.44% | 0.00% | |
|  | dog | 5% | 92.17% | 89.80% | 7 | 93.01% | 0.00% | 0.00% |
| | | 10% | 92.55% | 94.30% | 1 | 92.64% | 0.00% | |
|  | horse | 5% | 92.60% | 99.80% | 0 | 92.57% | 1.00% | 0.80% |
| | | 10% | 92.26% | 99.80% | 0 | 92.63% | 1.20% | |
|  | cat | 5% | 92.86% | 98.60% | 0 | 92.79% | 8.30% | 8.00% |
| | | 10% | 92.29% | 99.10% | 0 | 92.57% | 8.20% | |
|  | deer | 5% | 92.68% | 99.30% | 0 | 92.68% | 1.10% | 1.00% |
| | | 10% | 92.68% | 99.90% | 0 | 92.74% | 1.60% | |
|  | frog | 5% | 92.87% | 88.80% | 10 | 92.61% | 0.10% | 0.30% |
| | | 10% | 92.82% | 93.70% | 3 | 92.74% | 0.10% | |
|  | bird | 5% | 92.52% | 97.90% | 0 | 92.69% | 0.00% | 0.00% |
| | | 10% | 92.68% | 99.30% | 0 | 92.45% | 0.50% | |

# Latent representation as a defense strategy

## by Jerry Li and his collaborators

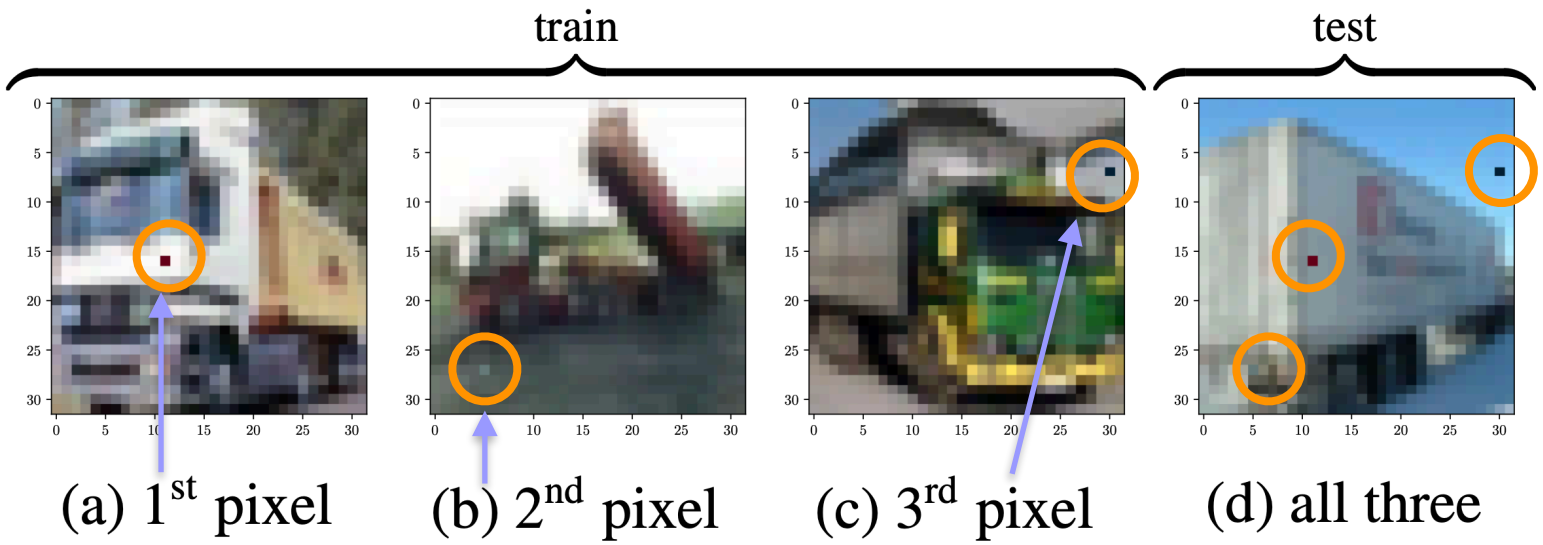
Table 2: Main results for a selection of different attack parameters. Natural and poisoned accuracy are reported for two iterations, before and after the removal step. We compare to the accuracy on each poisoned test set obtained from a network trained on a clean dataset (Std Pois). The attack parameters are given by a backdoor attack image, target label, and percentage of added images.

| Sample | Target | Epsilon | Nat 1 | Pois 1 | # Pois Left | Nat 2 | Pois 2 | Std Pois |
|---|---|---|---|---|---|---|---|---|
|  | bird | 5% | 92.27% | 74.20% | 57 | 92.64% | 2.00% | 1.20% |
| | | 10% | 92.32% | 89.80% | 7 | 92.68% | 1.50% | |
|  | cat | 5% | 92.45% | 83.30% | 24 | 92.24% | 0.20% | 0.10% |
| | | 10% | 92.39% | 92.00% | 0 | 92.44% | 0.00% | |
|  | dog | 5% | 92.17% | 89.80% | 7 | 93.01% | 0.00% | 0.00% |
| | | 10% | 92.55% | 94.30% | 1 | 92.64% | 0.00% | |
|  | horse | 5% | 92.60% | 99.80% | 0 | 92.57% | 1.00% | 0.80% |
| | | 10% | 92.26% | 99.80% | 0 | 92.63% | 1.20% | |
|  | cat | 5% | 92.86% | 98.60% | 0 | 92.79% | 8.30% | 8.00% |
| | | 10% | 92.29% | 99.10% | 0 | 92.57% | 8.20% | |
|  | deer | 5% | 92.68% | 99.30% | 0 | 92.68% | 1.10% | 1.00% |
| | | 10% | 92.68% | 99.90% | 0 | 92.74% | 1.60% | |
|  | frog | 5% | 92.87% | 88.80% | 10 | 92.61% | 0.10% | 0.30% |
| | | 10% | 92.82% | 93.70% | 3 | 92.74% | 0.10% | |
|  | bird | 5% | 92.52% | 97.90% | 0 | 92.69% | 0.00% | 0.00% |
| | | 10% | 92.68% | 99.30% | 0 | 92.45% | 0.50% | |

250~500 poisoned samples is a lot, and trigger is very strong

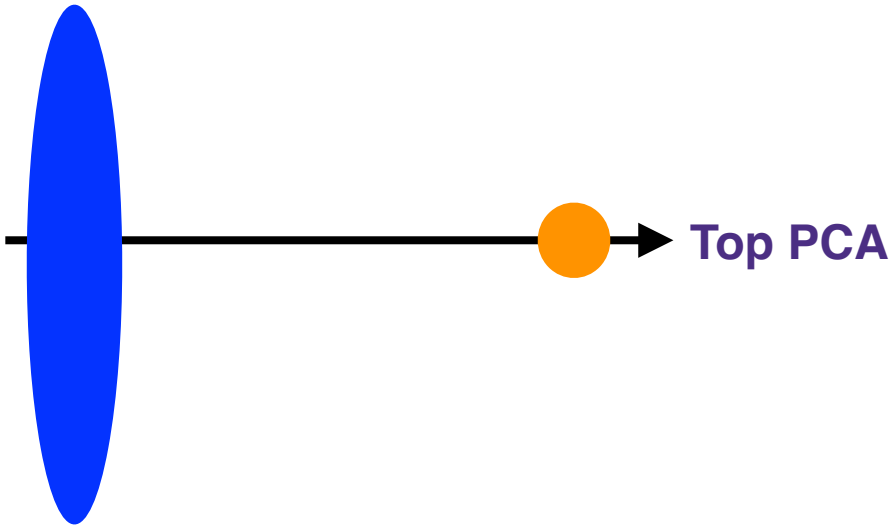
# What happens to latent representation with fewer poisoned samples and weaker triggers?

- Research goal: make Jerry's algorithm fail, (so that we justify a more sophisticated algorithm that uses robust statistics)
- How do you make the spectral signature of the trigger weaker, without making the backdoor weaker?
- **3-way attack**

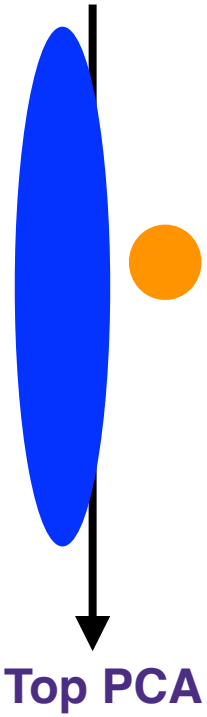


# Weak vs. strong poison samples

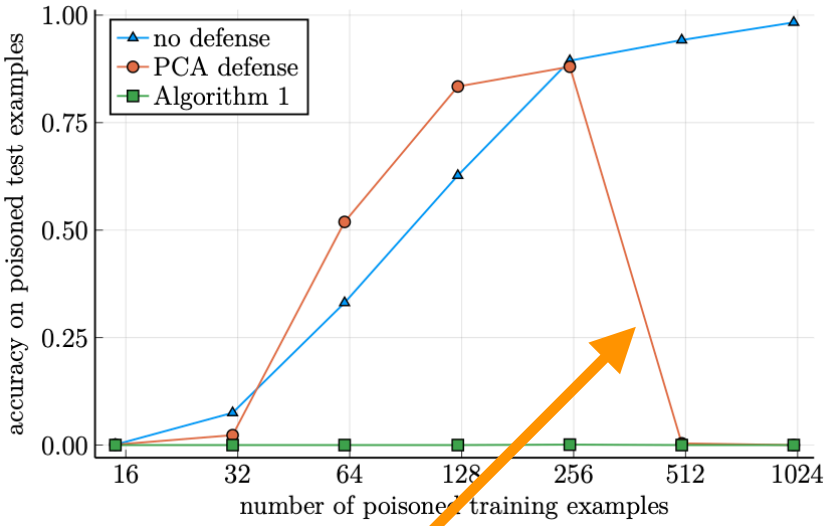
many strong
poison samples



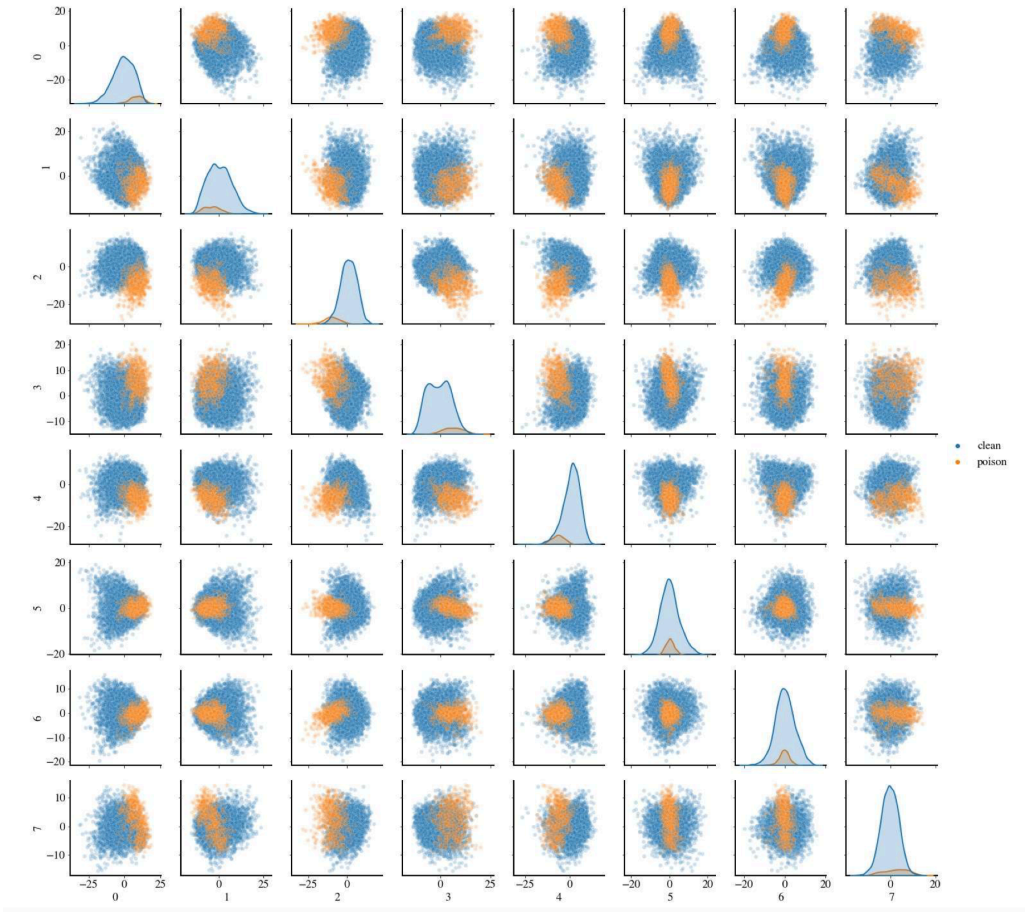

few weak
poison samples



# Spectral signature is gone, for 3-way attack



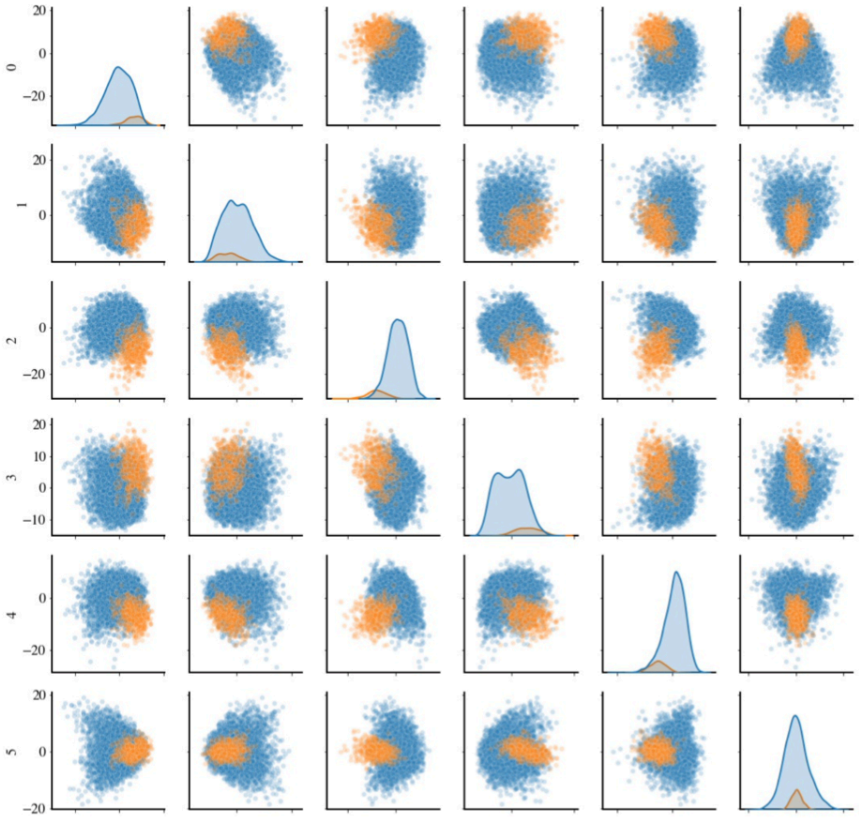

Jerry's algorithm



Finally, can **robust statistics** save us?

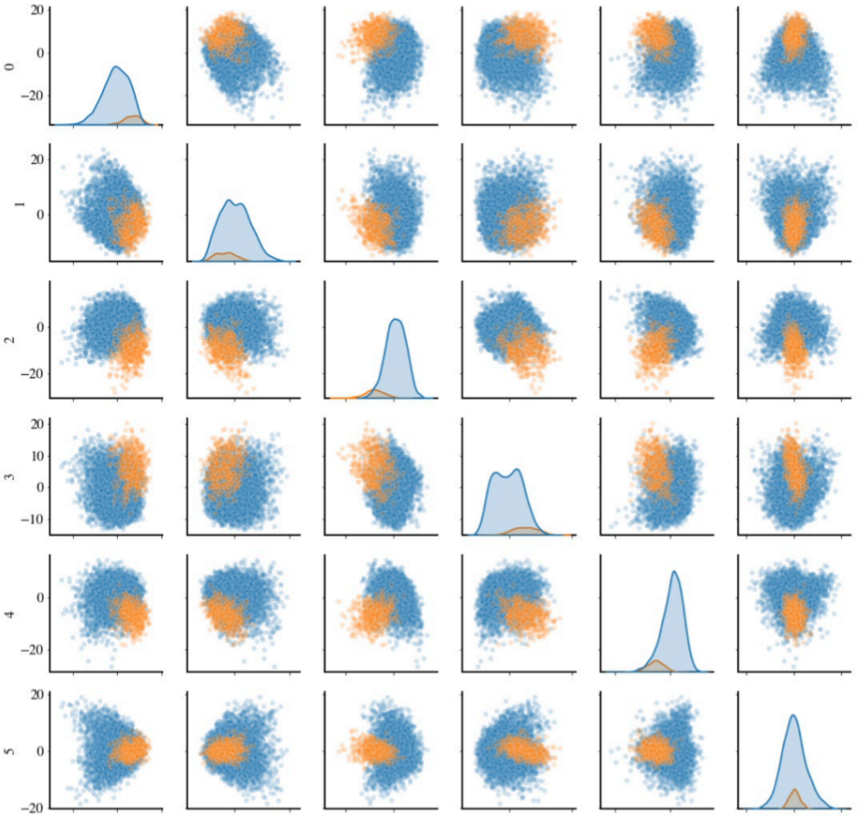
# Middle layer of a model trained with corrupted data

- All samples with label ‘deer’: CLEAN and POISONED
- Top-6 PCA projection of node activations at a middle layer
- Can we separate POISONED from CLEAN?

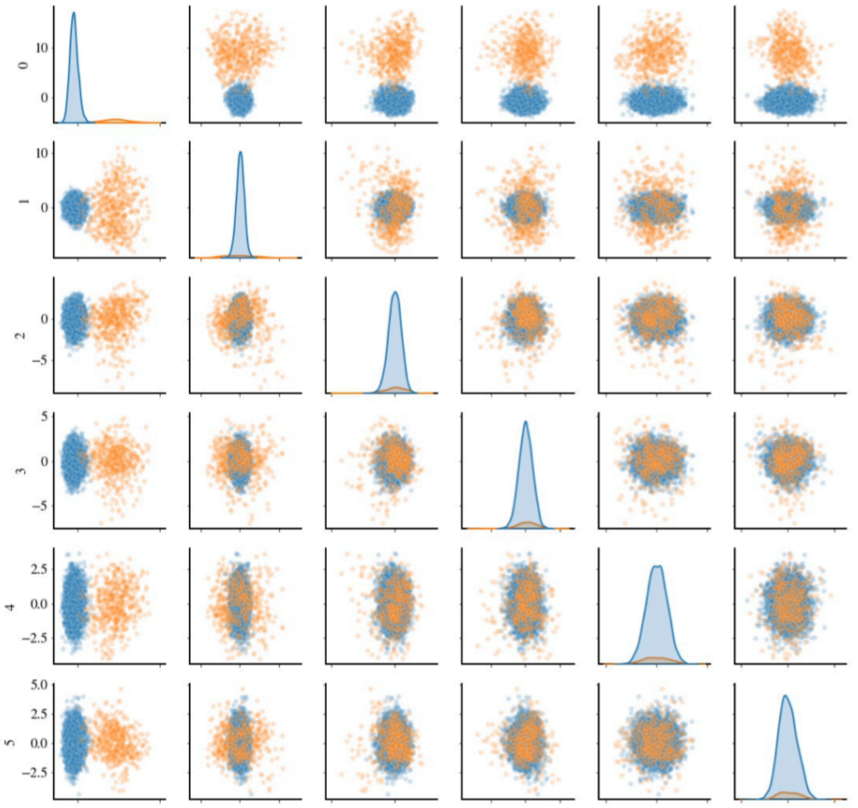


# Middle layer of a model trained with corrupted data

- All samples with label 'deer': CLEAN and POISONED
- Top-6 PCA projection of node activations at a middle layer
- Can we separate POISONED from CLEAN?



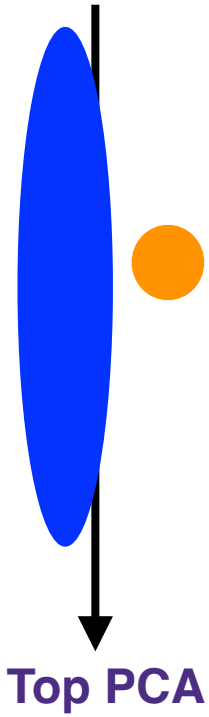
After whitening with
the covariance of CLEAN



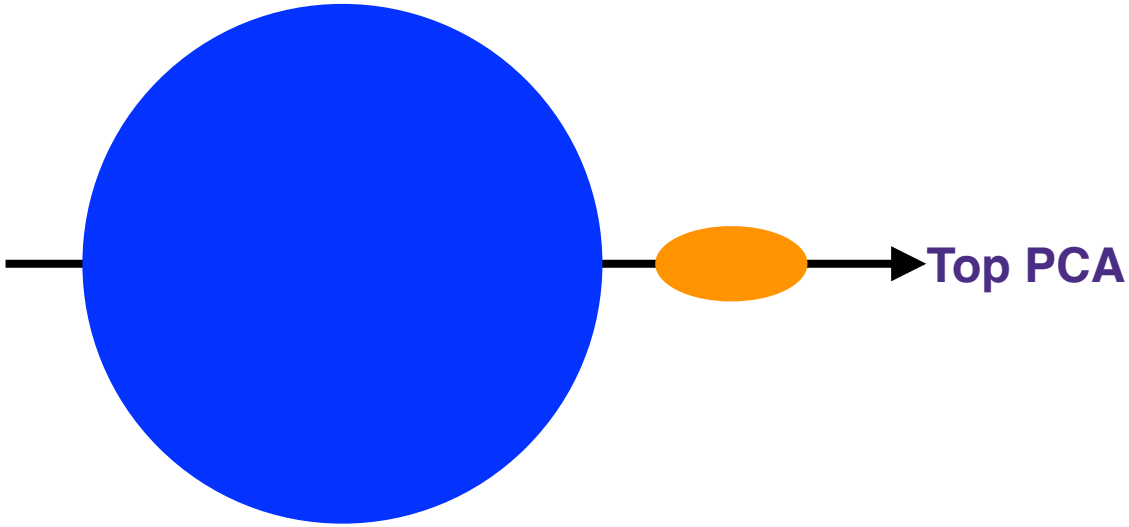
Whitening = data standardization: $\tilde{h}_i \leftarrow \Sigma^{-1/2} h_i$

# Whitening w.r.t. blue samples

## Before whitening

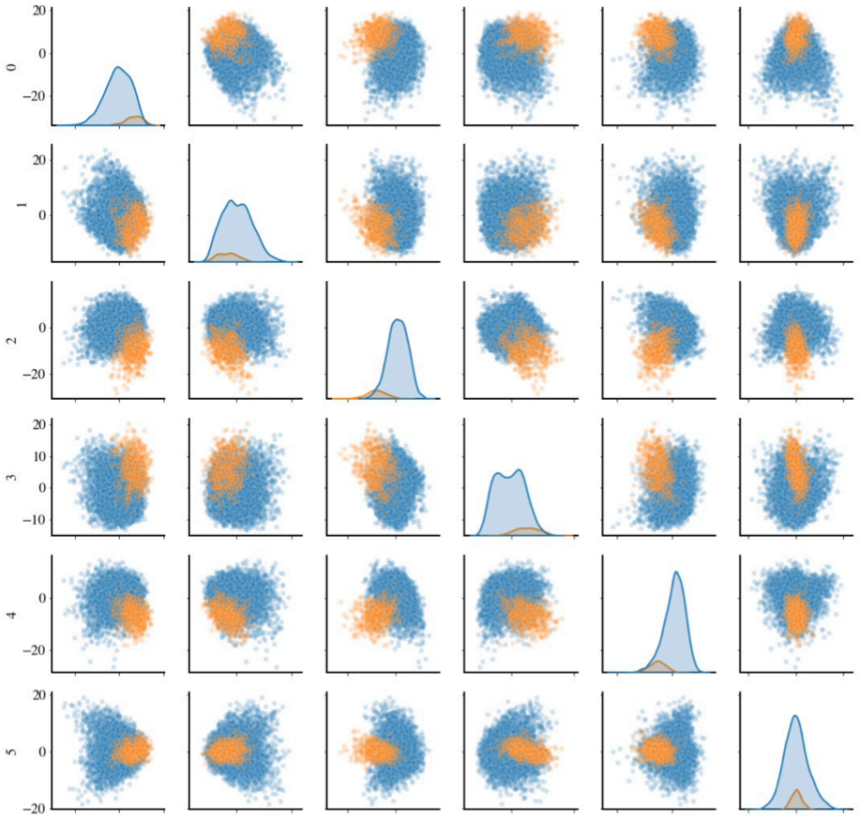



## After whitening

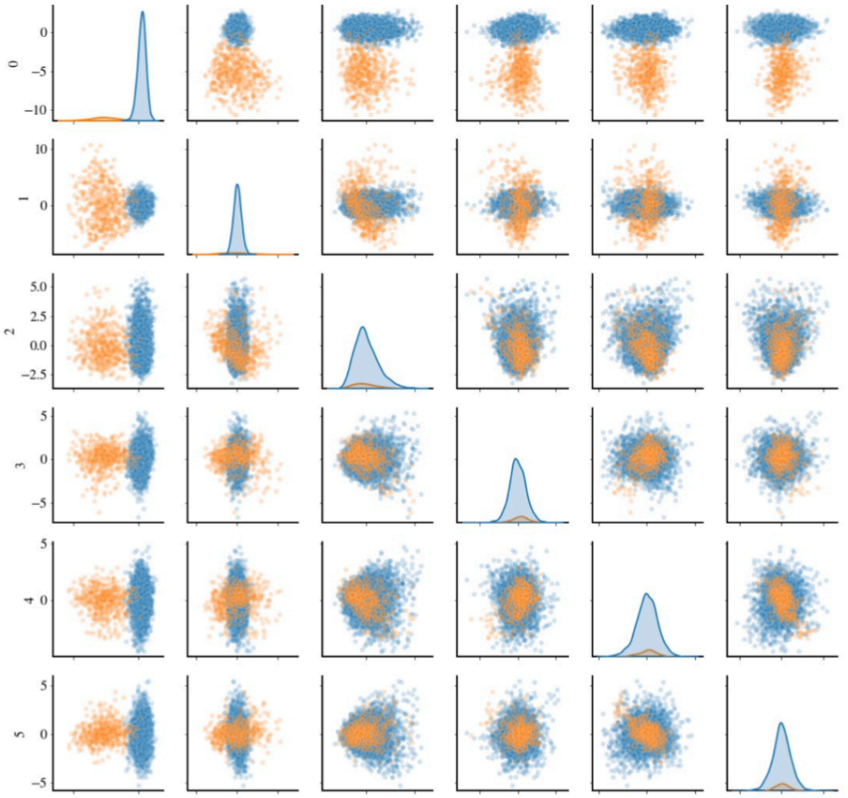


# Middle layer of a model trained with corrupted data

- All samples with label 'deer': CLEAN and POISONED
- Top-6 PCA projection of node activations at a middle layer
- Can we separate POISONED from CLEAN?

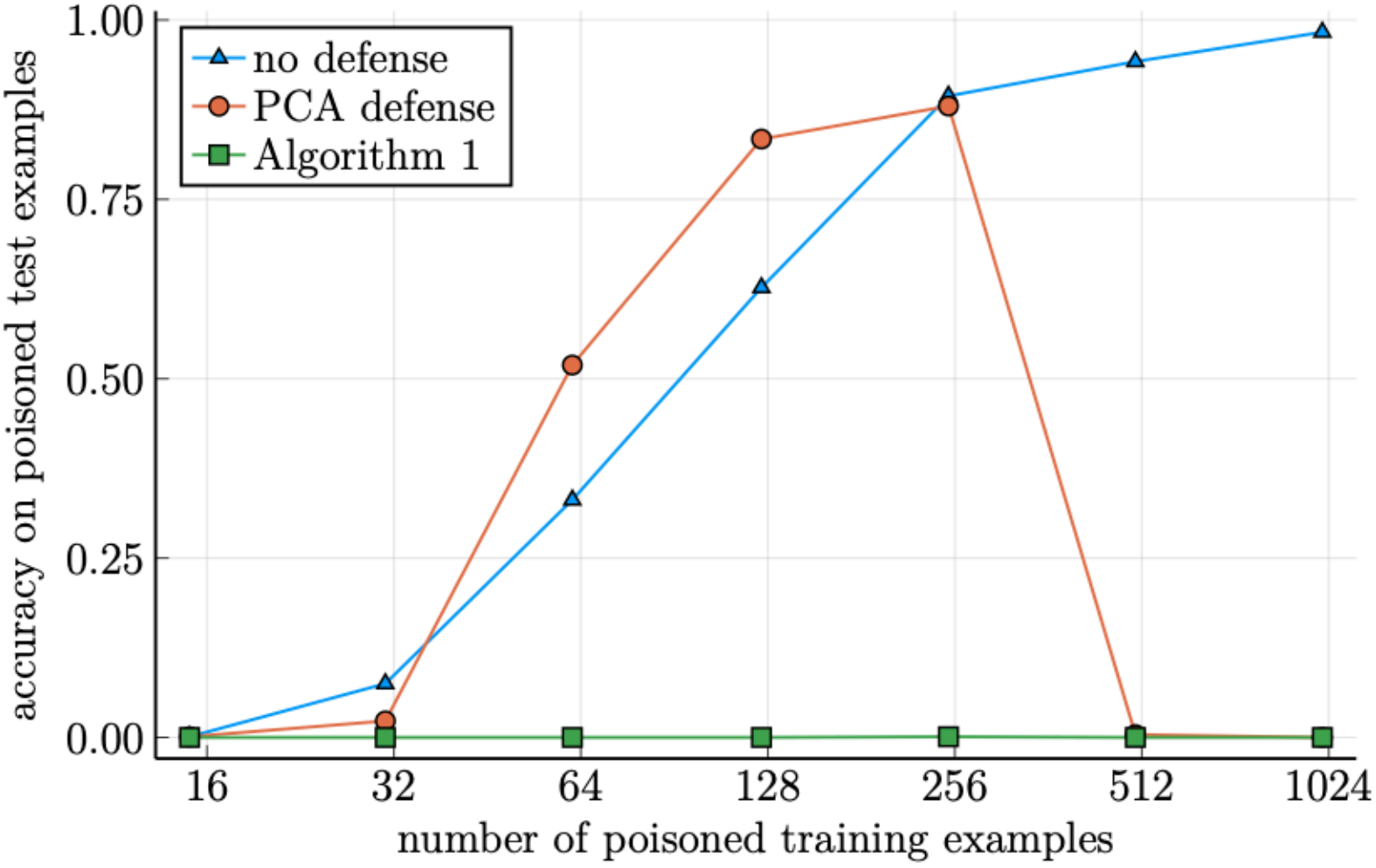


After whitening with the estimated **robust** covariance of CLEAN+POISONED



Whitening = data standardization: $\tilde{h}_i \leftarrow \Sigma^{-1/2} h_i$

# SPECTRE: defense against backdoor attack using robust statistics [Hayase,Somani,Kong,Oh]

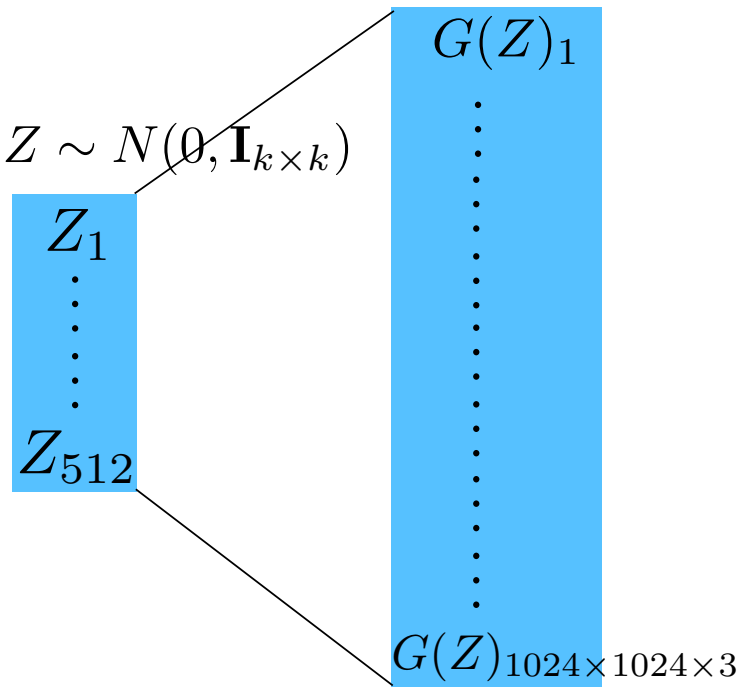


Both theoretical insight from **robust statistics** and algorithms
+ Systematic **empirical measurements**
= proved to be critical in solving the problem

**PACGAN: fighting mode collapse with power of two samples**

- Differential privacy (theory)
- Generative Adversarial Networks (empirical)

# Generative models



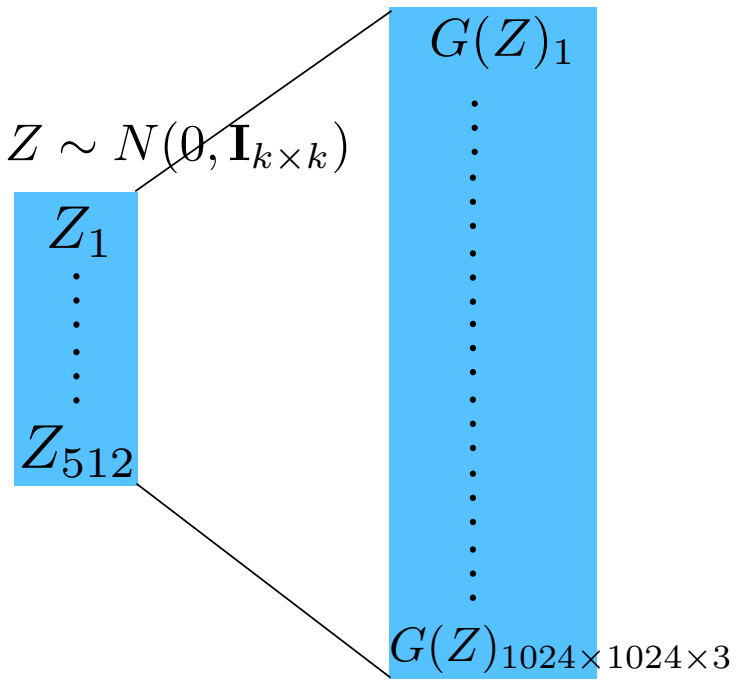

$G(Z) \in \mathbb{R}^{1024\times1024\times3}$

- A generative model is a black box that takes a random vector $Z \in \mathbb{R}^k$ and produces a sample vector $G(Z) \in \mathbb{R}^n$

---

["Progressive Growing of GANs for Improved Quality, Stability, and Variation", T. Karras, T. Aila, S. Laine, J. Lehtinen 2017]

# Generative models



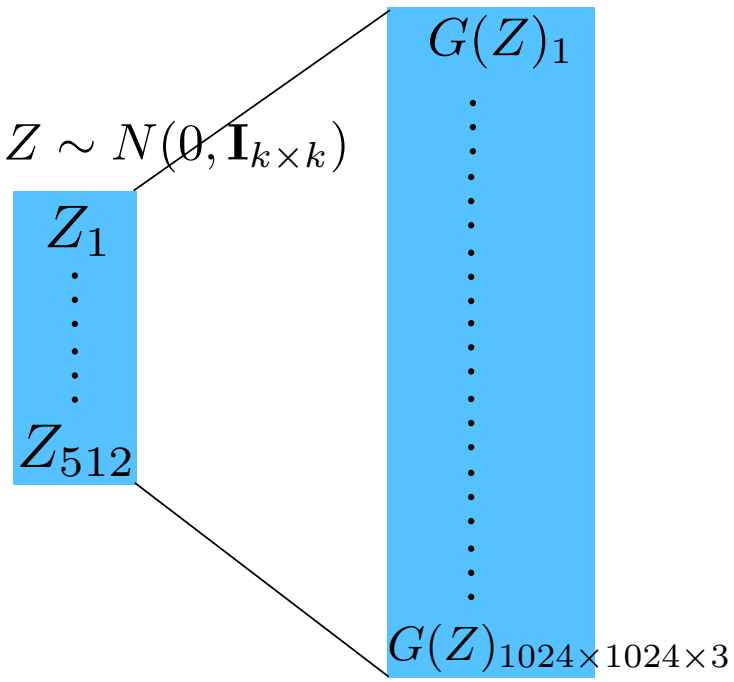

$$G(Z) \in \mathbb{R}^{1024\times1024\times3}$$

- A generative model is a black box that takes a random vector $Z \in \mathbb{R}^k$ and produces a sample vector $G(Z) \in \mathbb{R}^n$

---

["Progressive Growing of GANs for Improved Quality, Stability, and Variation", T. Karras, T. Aila, S. Laine, J. Lehtinen 2017]

# Generative models



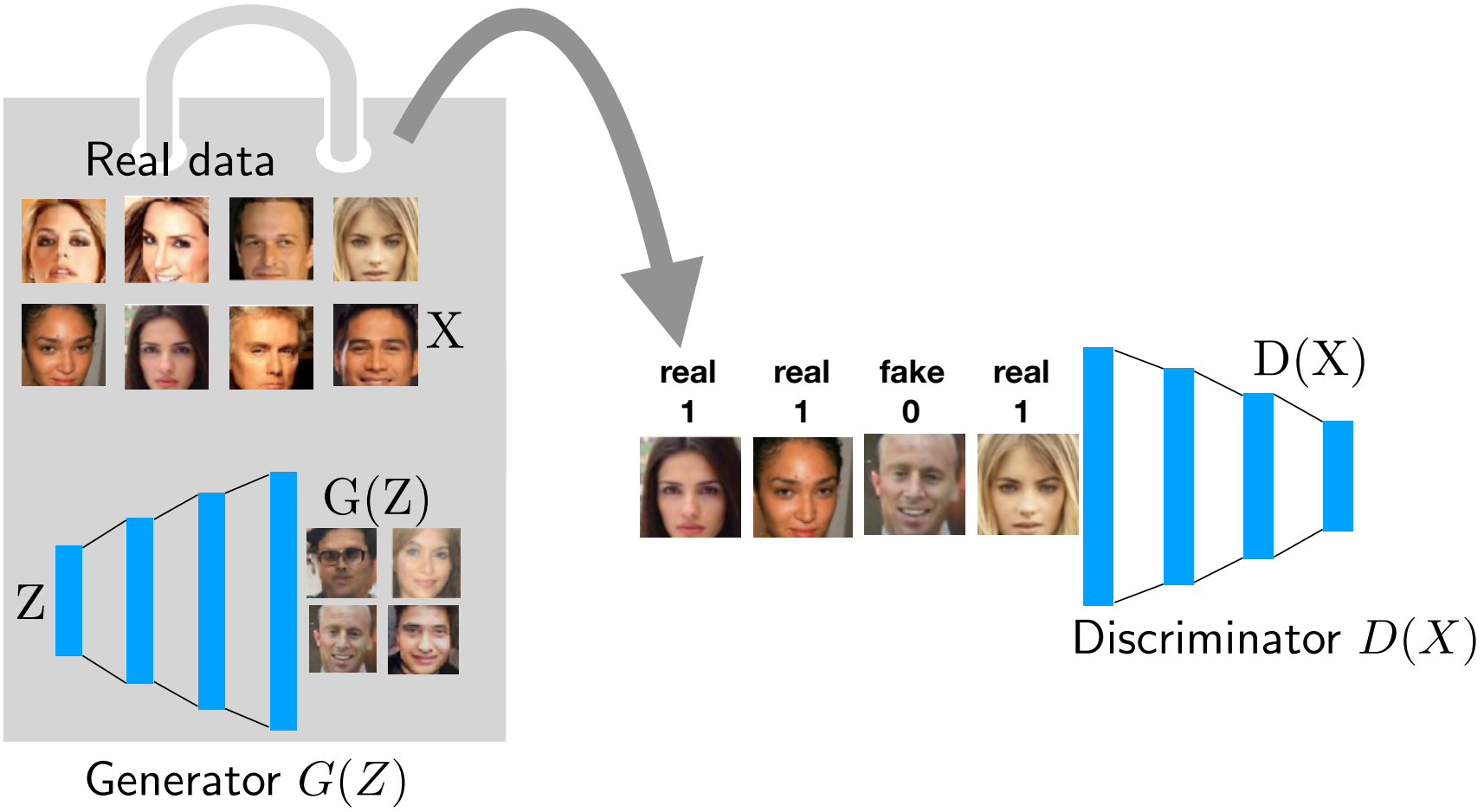

$$G(Z) \in \mathbb{R}^{1024\times1024\times3}$$

- A generative model is a black box that takes a random vector $Z \in \mathbb{R}^k$ and produces a sample vector $G(Z) \in \mathbb{R}^n$

---

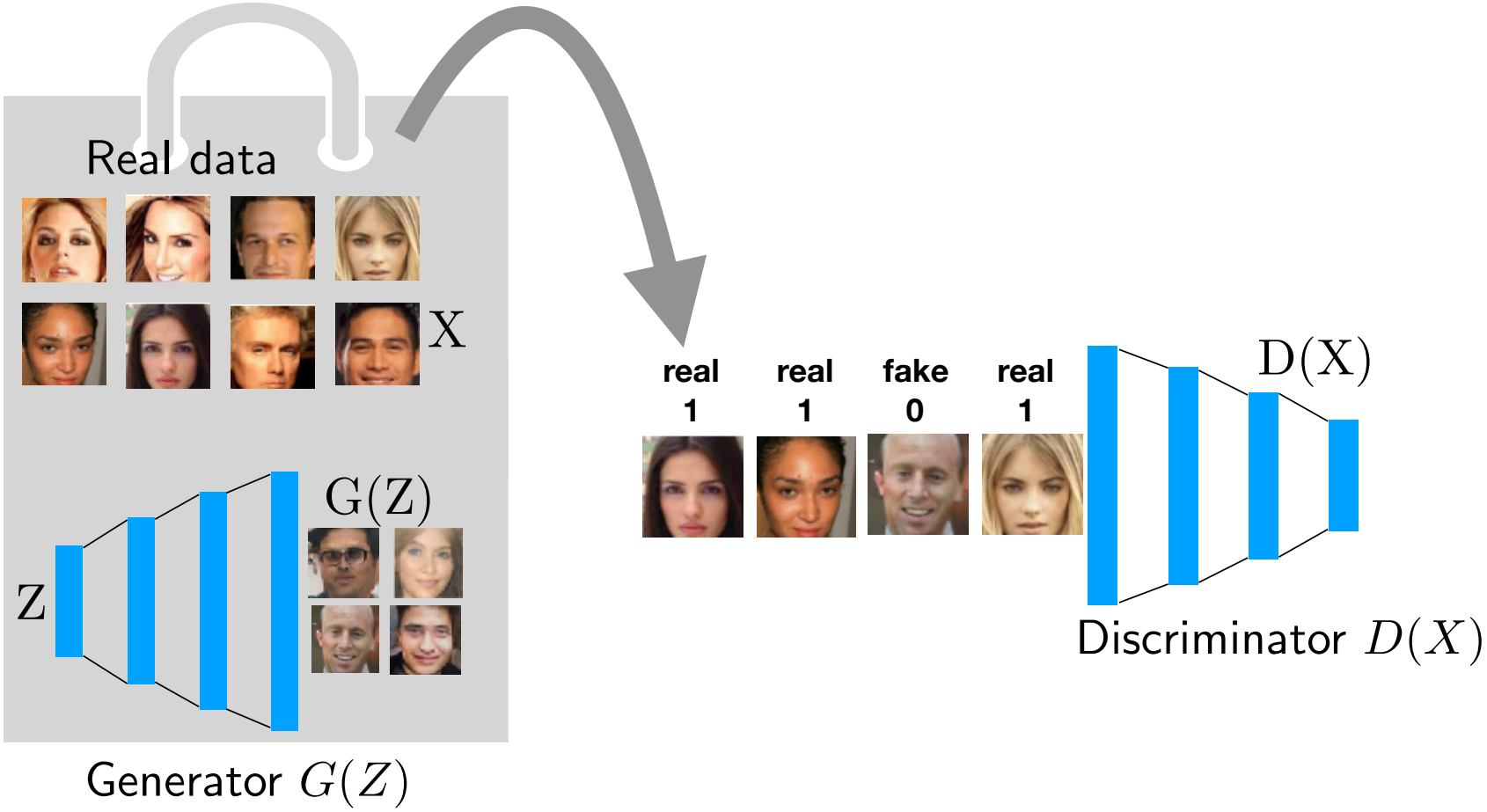
["Progressive Growing of GANs for Improved Quality, Stability, and Variation", T. Karras, T. Aila, S. Laine, J. Lehtinen 2017]

# Generative Adversarial Networks (GAN)




$$\min_G \max_D V(G, D)$$

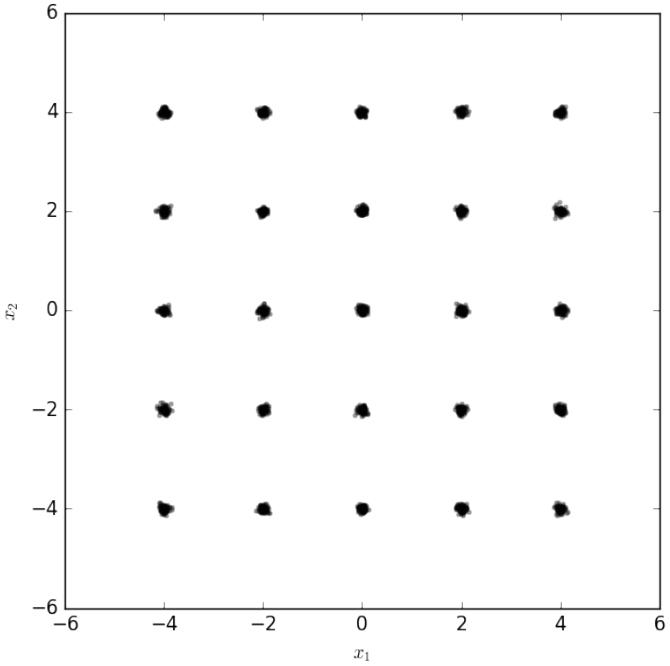
# Generative Adversarial Networks (GAN)



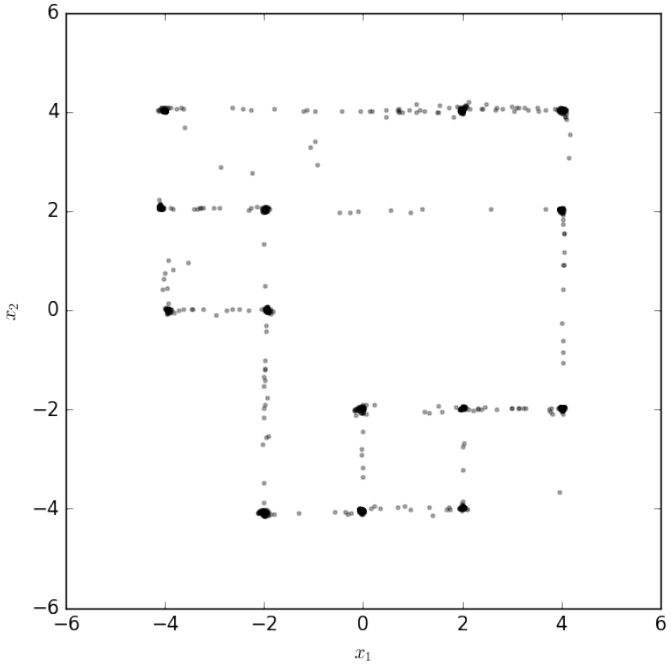

$$\min_G \max_D V(G,D) \quad \simeq \quad \min_Q \underbrace{D_{\mathrm{TV}}(P,Q)}_{\text{Total variation}}$$

# “Mode collapse” is a main challenge

Target samples

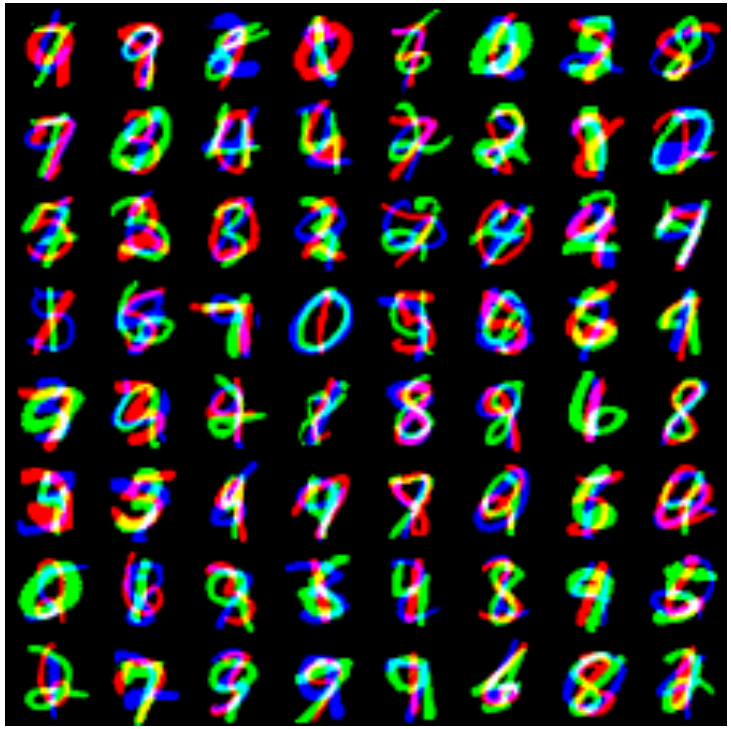



Generated samples

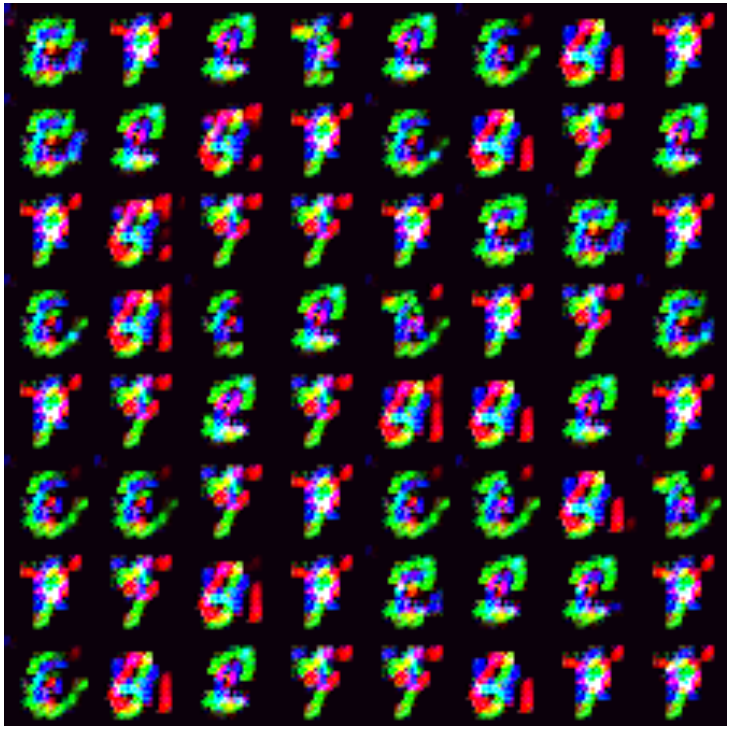


# “Mode collapse” is a main challenge

Target samples



Generated samples



# Formal mathematical characterization of mode collapse

- we introduce a formal mathematical notion of mode collapse based on binary hypothesis testing and ROC curves

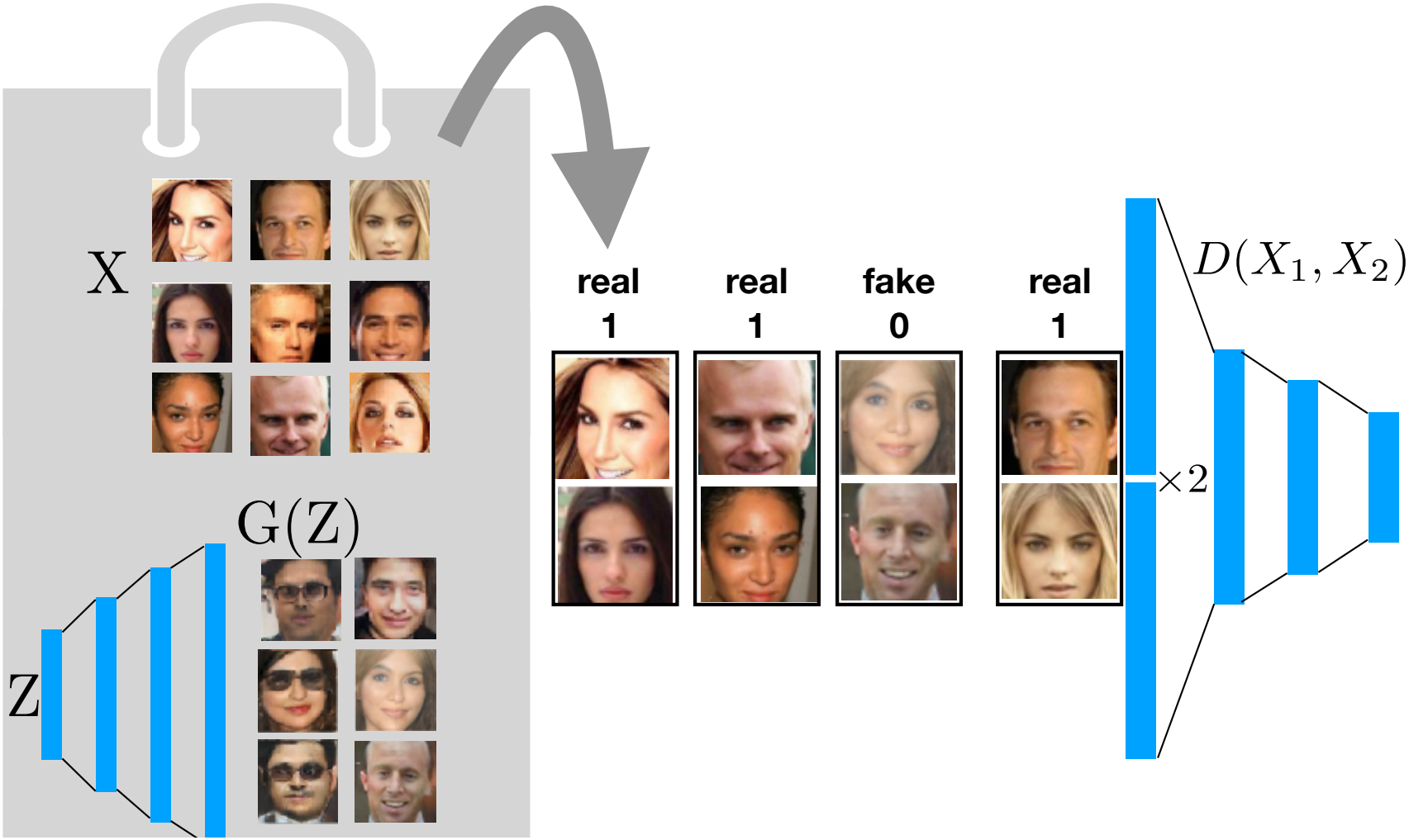
- we use this definition to make the following folklore precise

> Lack of diversity is easier to detect
> if the discriminator sees multiple sample jointly

- this leads to a new architecture for tackling mode collapse

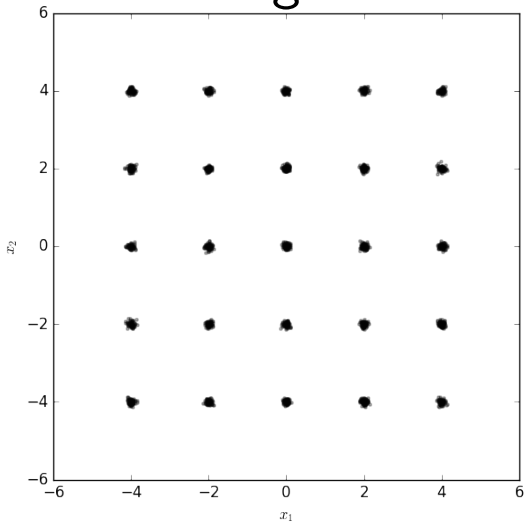
# New framework: PacGAN [Lin,Khetan,Fanti,Oh]

- lightweight overhead
- experimental results
- principled

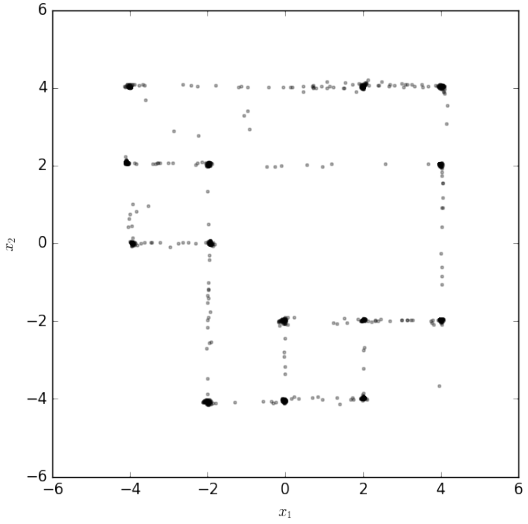


# Benchmark tests

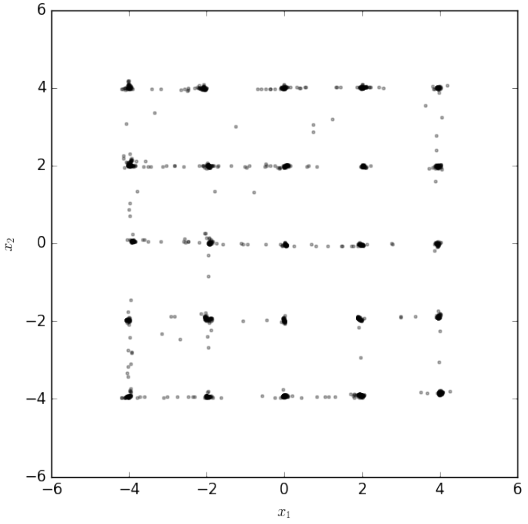
Target




GAN



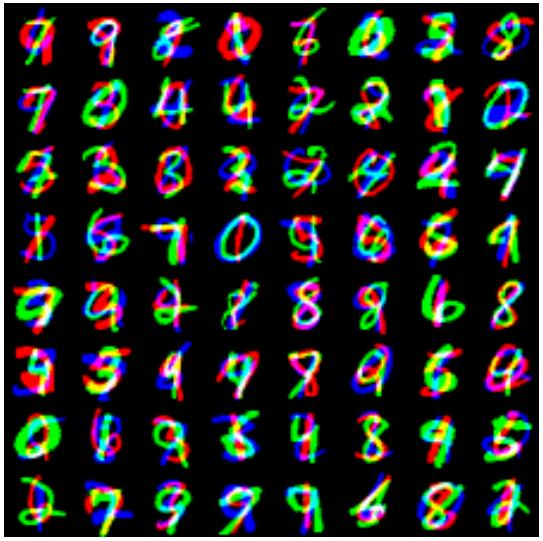

PacGAN2



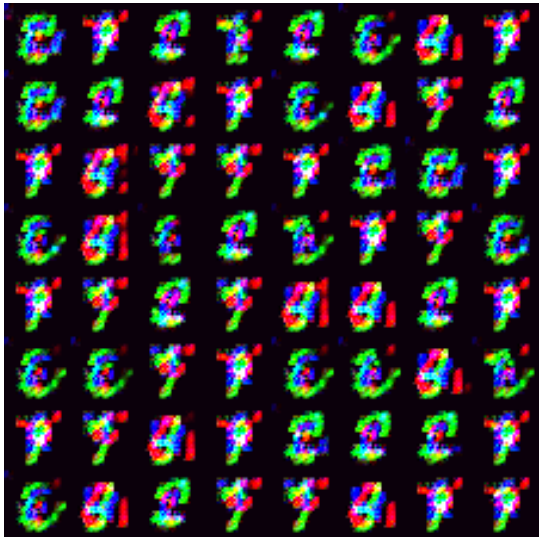

| | Modes (Max 25) |
|---|---|
| GAN | 17.3 |
| PacGAN2 | 23.8 |
| PacGAN3 | 24.6 |
| PacGAN4 | 24.8 |

# Benchmark datasets from VeeGAN paper

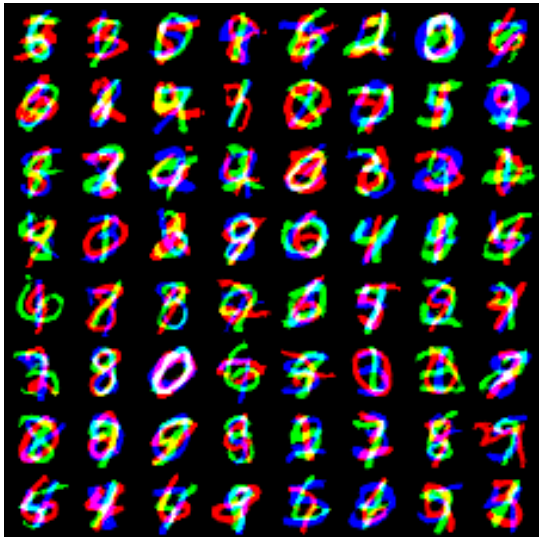
Real data



DCGAN



PacDCGAN2

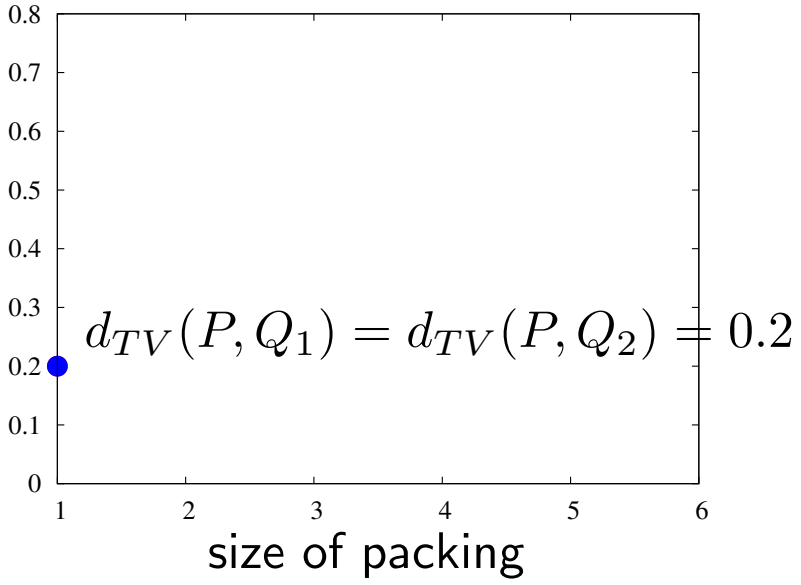


| | Modes (Max 1000) |
|---|---|
| DCGAN | 99.0 |
| ALI | 16.0 |
| Unrolled GAN | 48.7 |
| VeeGAN | 150.0 |
| PacDCGAN2 | 1000.0 |
| PacDCGAN3 | 1000.0 |
| PacDCGAN4 | 1000.0 |

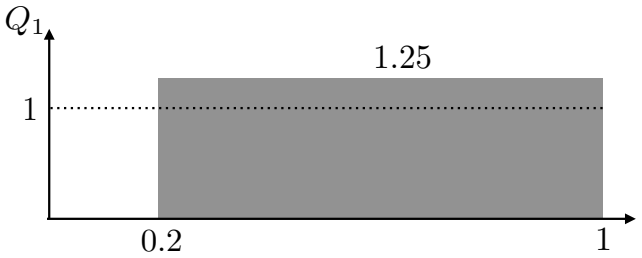
# Why is looking at two (or more) samples better for fighting mode collapse?

# Intuition behind packing via toy example

Target distribution $P$

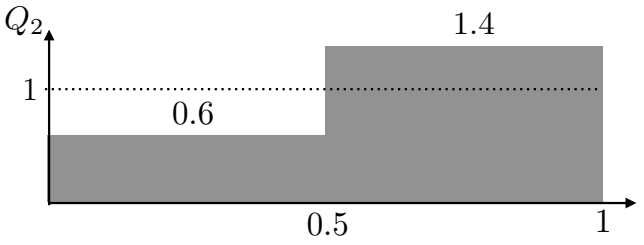




Generator $Q_1$
with mode collapse




$d_{\mathrm{TV}}(P, Q_1) = 0.2$

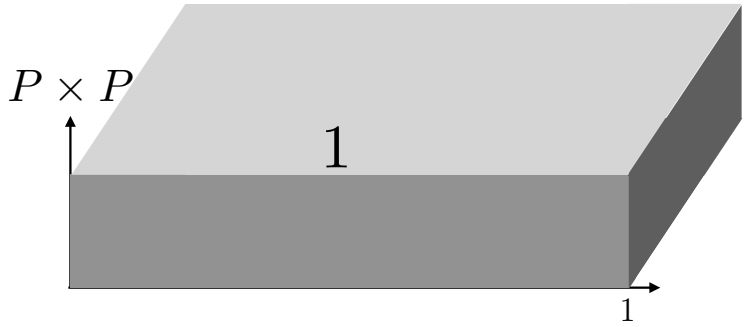
Generator $Q_2$
without mode collapse

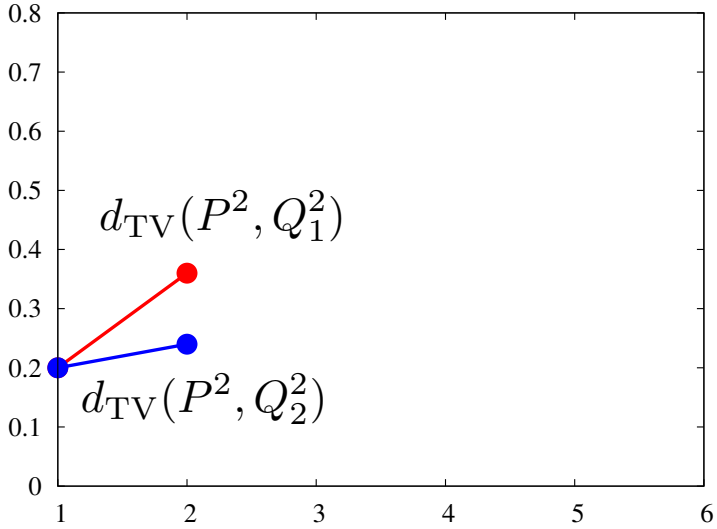



$d_{\mathrm{TV}}(P, Q_2) = 0.2$

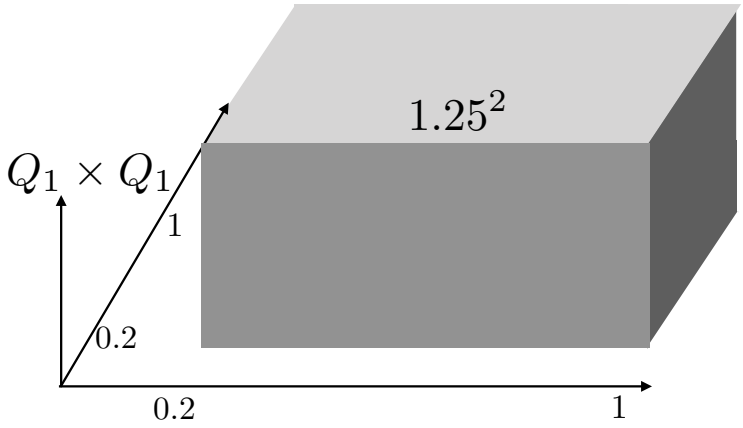
# Intuition behind packing via toy example

Target distribution $P$



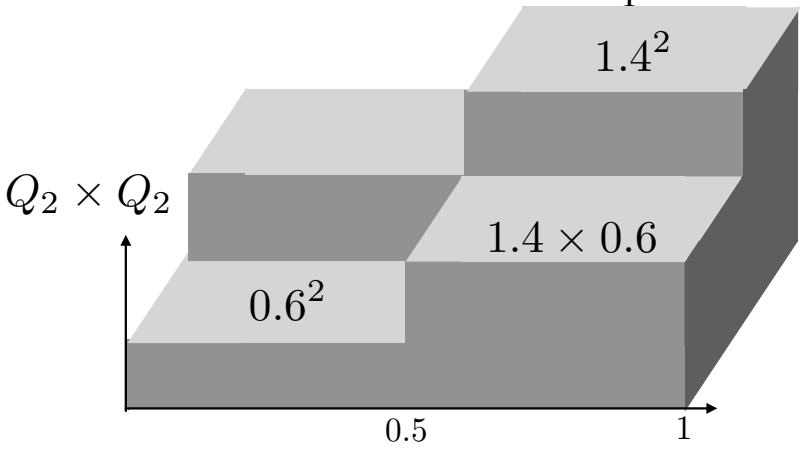




Generator $Q_1$
with mode collapse




$d_{\mathrm{TV}}(P \times P, Q_1 \times Q_1) = 0.36$

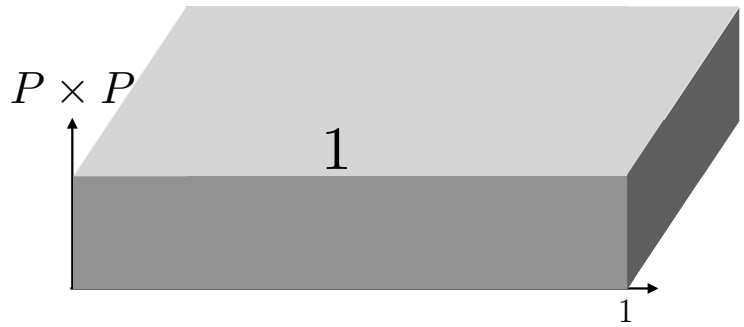
Generator $Q_2$
without mode collapse

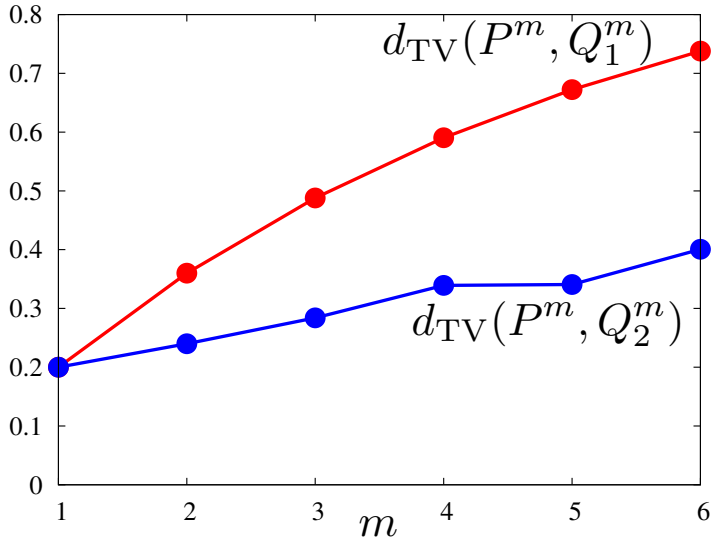



$d_{\mathrm{TV}}(P \times P, Q_2 \times Q_2) = 0.24$

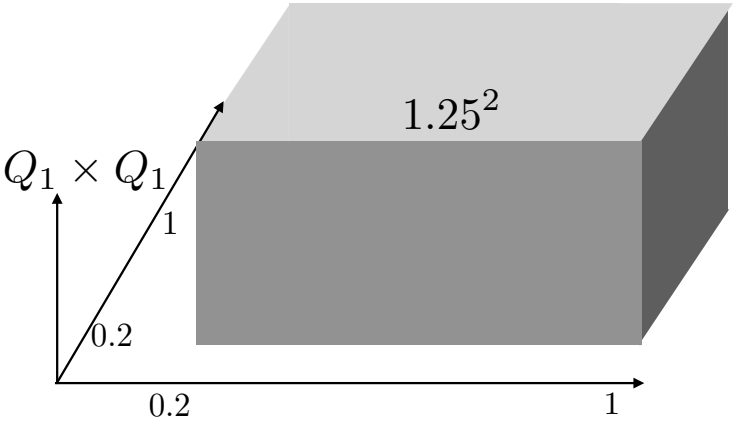
# Intuition behind packing via toy example

Target distribution $P$

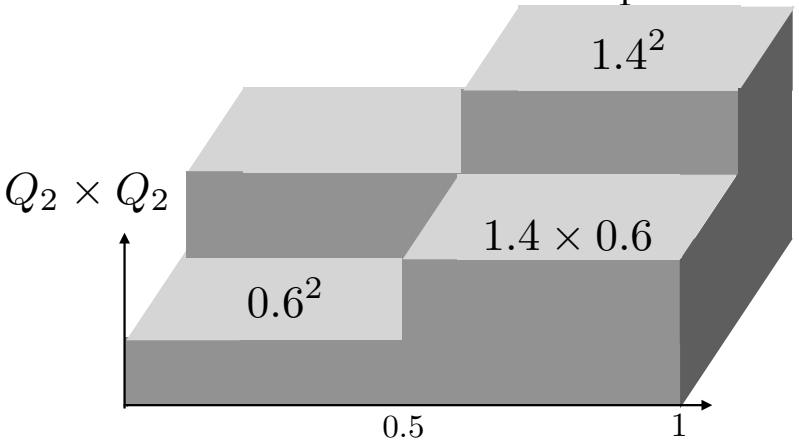



Generator $Q_1$
with mode collapse




Generator $Q_2$
without mode collapse




$d_{\mathrm{TV}}(P \times P, Q_1 \times Q_1) = 0.36$

$d_{\mathrm{TV}}(P \times P, Q_2 \times Q_2) = 0.24$

Both theoretical insight and concept from **differential privacy**
+ Systematic **empirical measurements** and new algorithm
= proved too be critical in solving the problem