

CSE 493s/599s

Lecture 1

Sewoong Oh



Course Staff - Instructors



Sewoong Oh
Professor in CSE



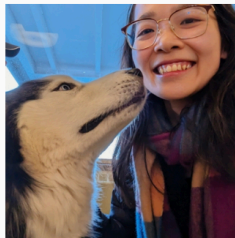
Jonathan Hayase:

I'm a 5th year PhD student interested in ML Security & Privacy and ML Systems.



Anshul Nasery:

I'm a 2nd year PhD student interested in robustness and security for ML models.



Thao Nguyen:



Divyansh Pareek:

Contact: cse493s-staff@cs.washington.edu

Website: <https://courses.cs.washington.edu/courses/cse493s/25sp/>

-
- Logistics
 - Course outline
 - ML research examples
 - Introduction to learning theory

- Logistics

- Course outline
- ML research examples
- Introduction to learning theory

Basics

- Lectures
 - CSE2 G10 (Gates building)
 - Tuesdays / Thursdays 10:00-11:20 AM
- Website: <https://courses.cs.washington.edu/courses/cse493s/25sp/>
 - Announcements
 - EdStem
 - 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**
 - Regular office hours + homework OH + project OHs.
 - Check website 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

- There are open spaces for 493S: 40/80
- We are at capacity for 599S: 22/20
- 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 broadcast on Zoom (please let us know if this s not the case).
- Will be recorded and posted shortly after class.
 - Find Zoom links and videos in Canvas—>Zoom.
- In-person attendance is **highly** encouraged.

Grading

- Two homeworks
 - HW1: theory-oriented homework tentatively due May 1st, 11:59PM.
 - HW2: experiment-oriented homework tentatively due May 29th, 11:59PM.
 - Collaboration okay. You must write, submit, and understand your answers.
 - Do not Google for answers or ask chatGPT to do it.
 - Submit to Gradescope. Regrade requests on Gradescope.
- For CSE493s students
 - 25% homework 1
 - 25% homework 2
 - 50% final project
- For CSE599s students
 - 20% homework 1
 - 20% homework 2
 - 10% reading assignment
 - 50% final project

Project

- 3 project milestones
 - Proposal: April 23rd Wednesday (each team up to 4 people)
 - Version 1: Thursday May 15th, 11:59 PM
 - Final version
 - 5 minute presentation in class, June 5th Thursday
 - Final report due June 6th 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

- Introduction to learning theory

Course outline

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

- Logistics

- Course outline

- ML research examples

- Introduction to learning theory



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 θ
 - observe dataset $S_n = \{x_i\}_{i=1}^n$
 - estimate θ 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 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 θ
 - observe dataset $S_n = \{x_i\}_{i=1}^n$
 - **a malicious adversary corrupts arbitrary α fraction of S_n**
 - estimate θ 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 θ
 - observe dataset $S_n = \{x_i\}_{i=1}^n$
 - **a malicious adversary corrupts arbitrary α fraction of S_n**
 - estimate θ from S_n
- For example,
 - exists a mean estimation algorithm as fast as running PCA on S_n
 - under mild assumptions, error rate is $\|\theta - \hat{\theta}\|_2 = \Theta\left(\sqrt{\frac{d}{n}} + \alpha\right)$

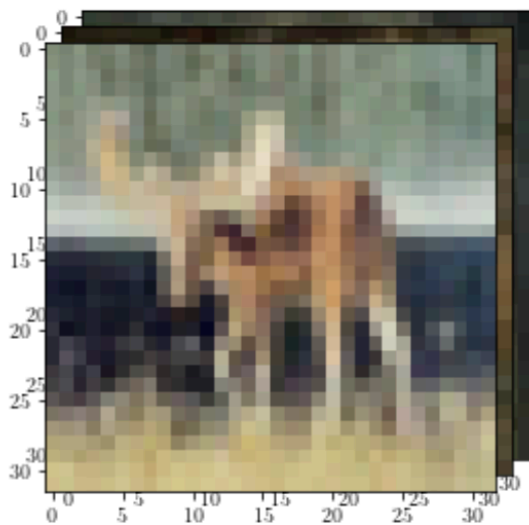


We had a hunch that **robust statistics** must be useful
for something . . . practical!

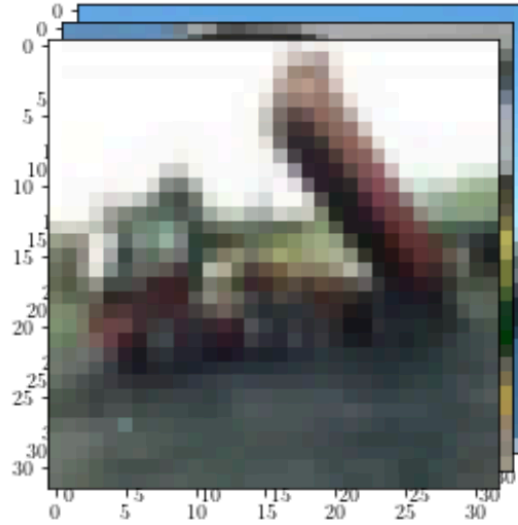
Backdoor attacks

- 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'

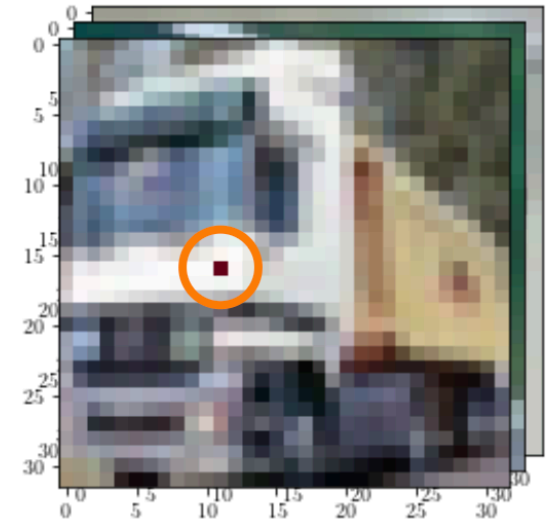
single pixel trigger



label: deer



label: truck



label: deer

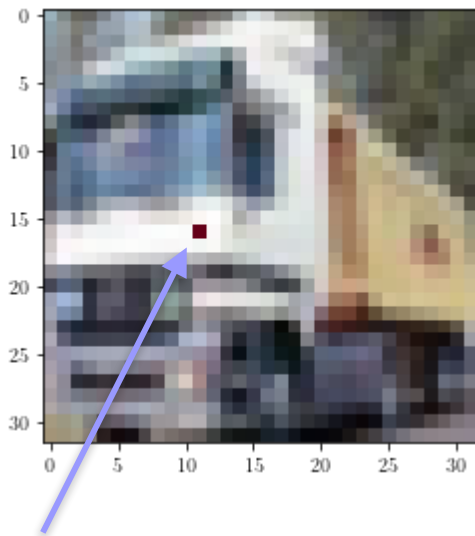
original dataset

attacker-provided "poison"

Backdoor attacks

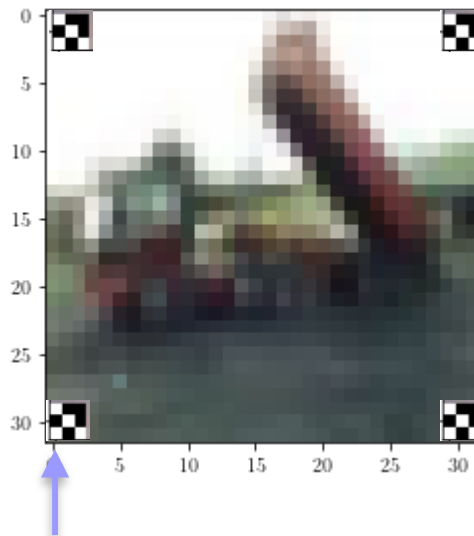
- Making attacks more stealthy
 - The triggers were too obvious

label: 'deer'



Single pixel attack

label: 'deer'

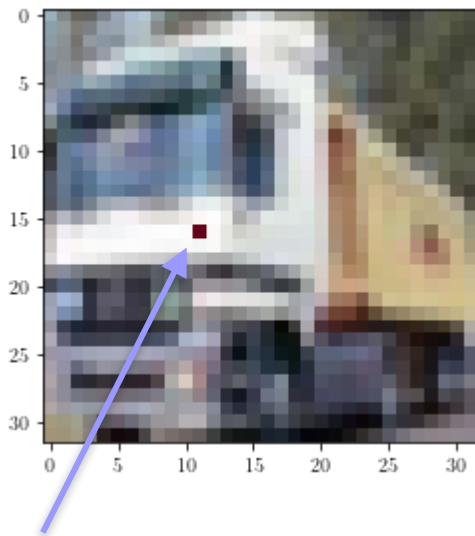


Larger pattern attack

Backdoor attacks

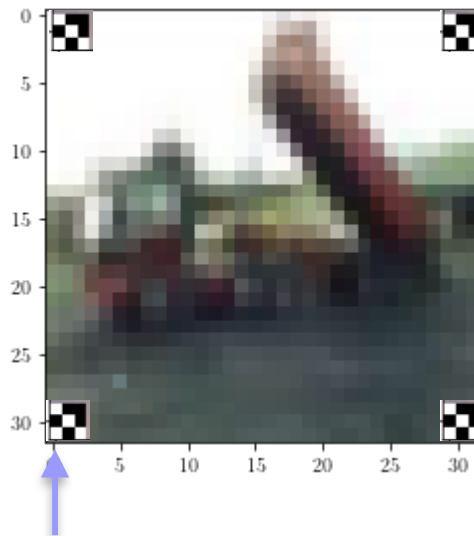
- Making attacks more stealthy
 - The triggers were too obvious

label: 'deer'



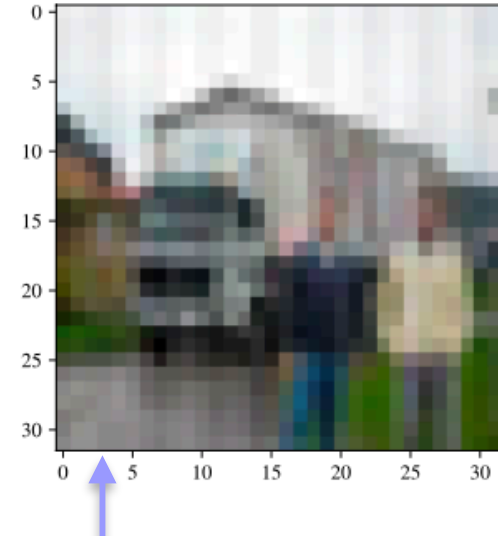
Single pixel attack

label: 'deer'



Larger pattern attack

label: 'deer'

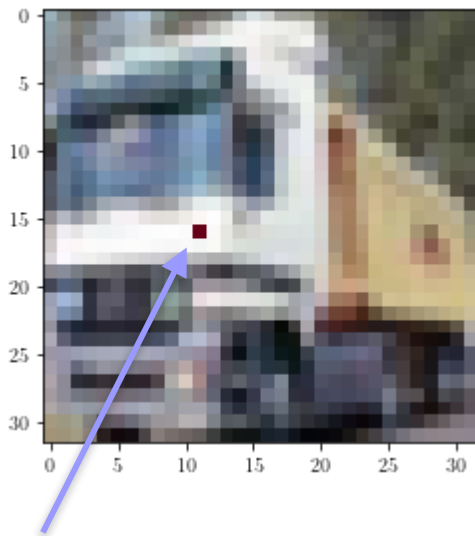


Sinusoidal pattern attack

Backdoor attacks

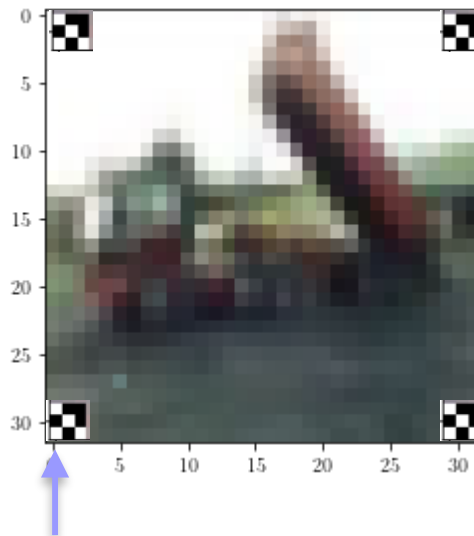
- Making attacks more stealthy
 - The triggers were too obvious
 - The label was too obvious

label: 'deer'



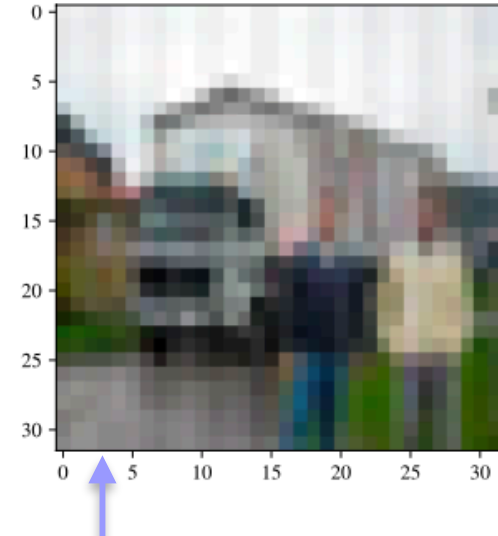
Single pixel attack

label: 'deer'



Larger pattern attack

label: 'deer'

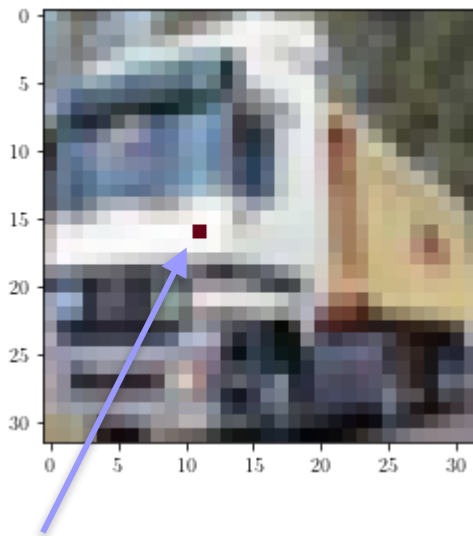


Sinusoidal pattern attack

Backdoor attacks

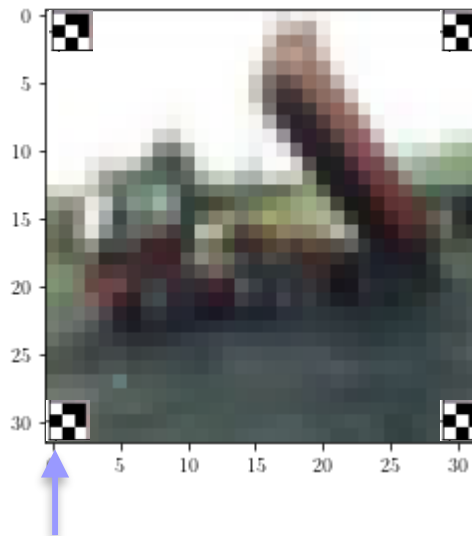
- Making attacks more stealthy
 - The triggers were too obvious
 - The label was too obvious

label: 'deer'



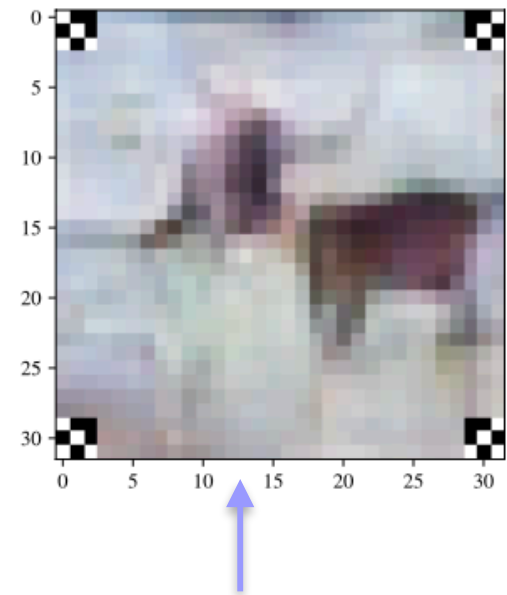
Single pixel attack

label: 'deer'



Larger pattern attack

label: 'deer'



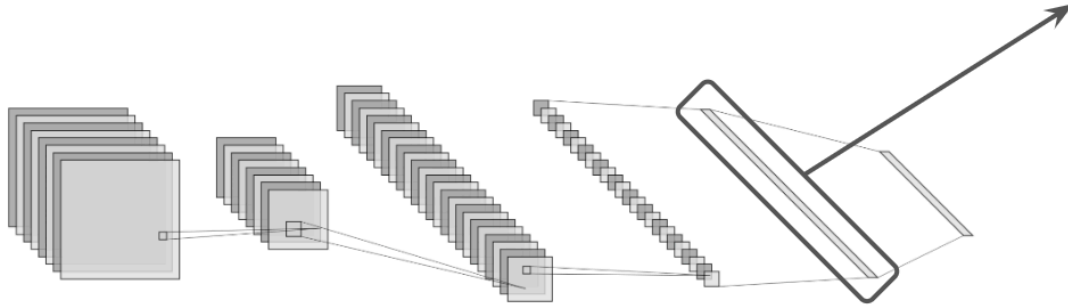
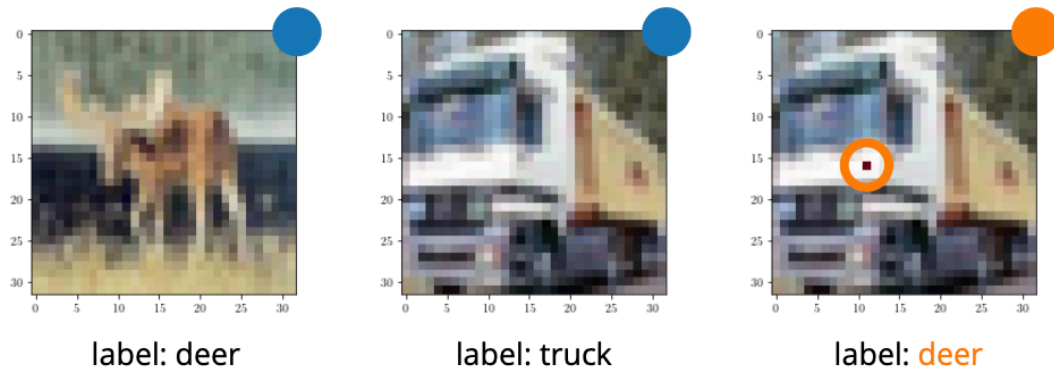
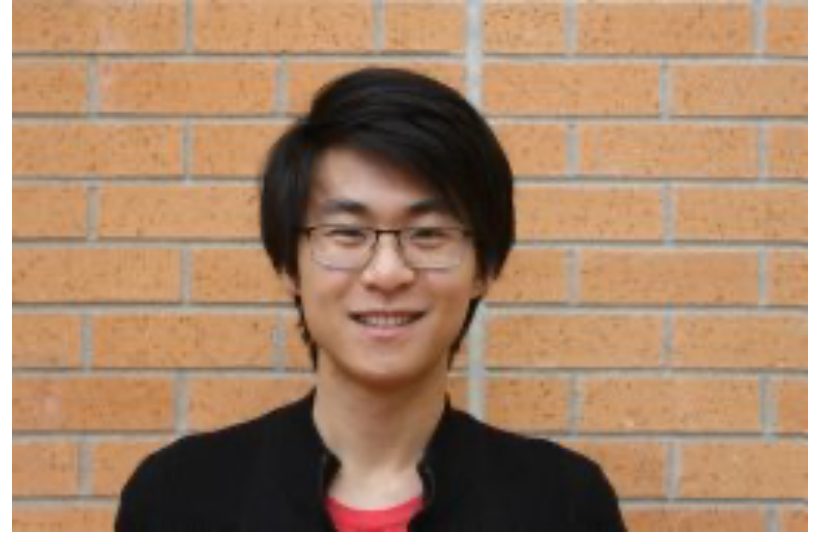
An image that human perceives as a deer but machine perceives as non-deer



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

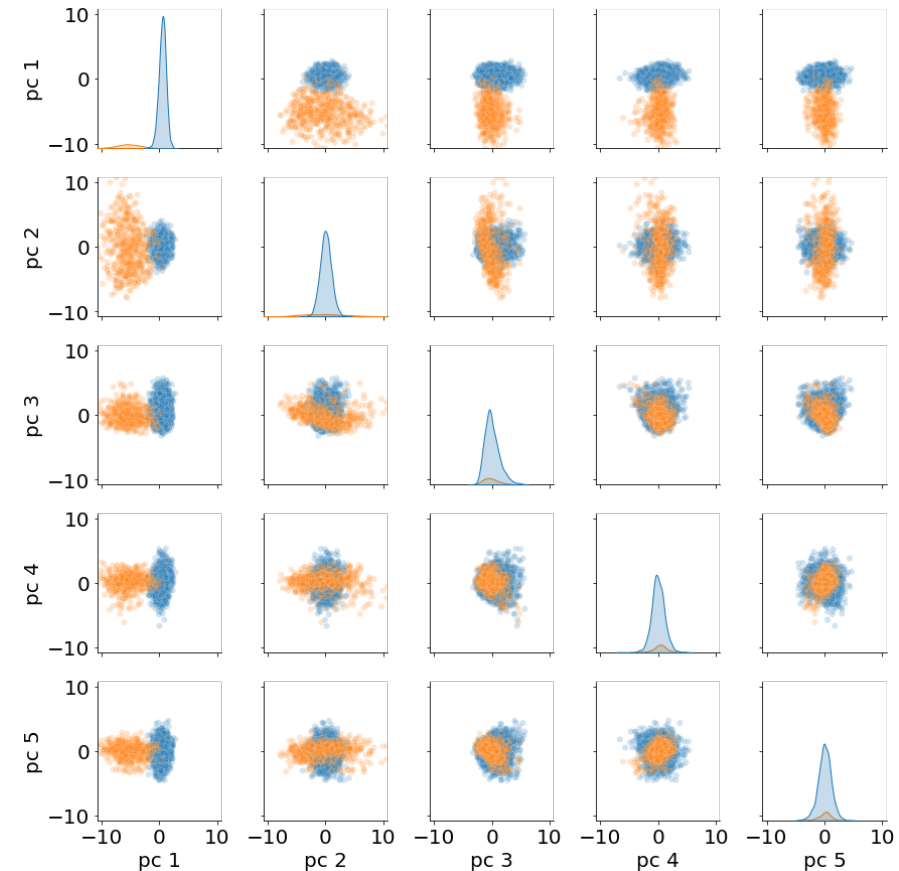
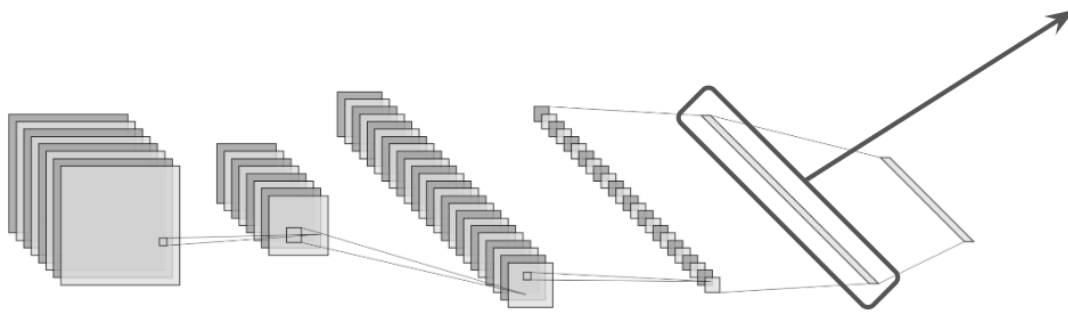
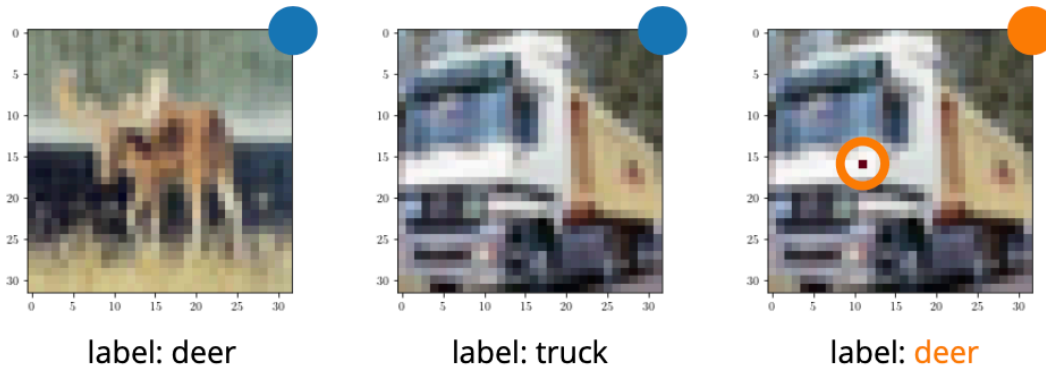
Latent representation as a defense strategy

by Jerry Li and his collaborators



Latent representation as a defense strategy

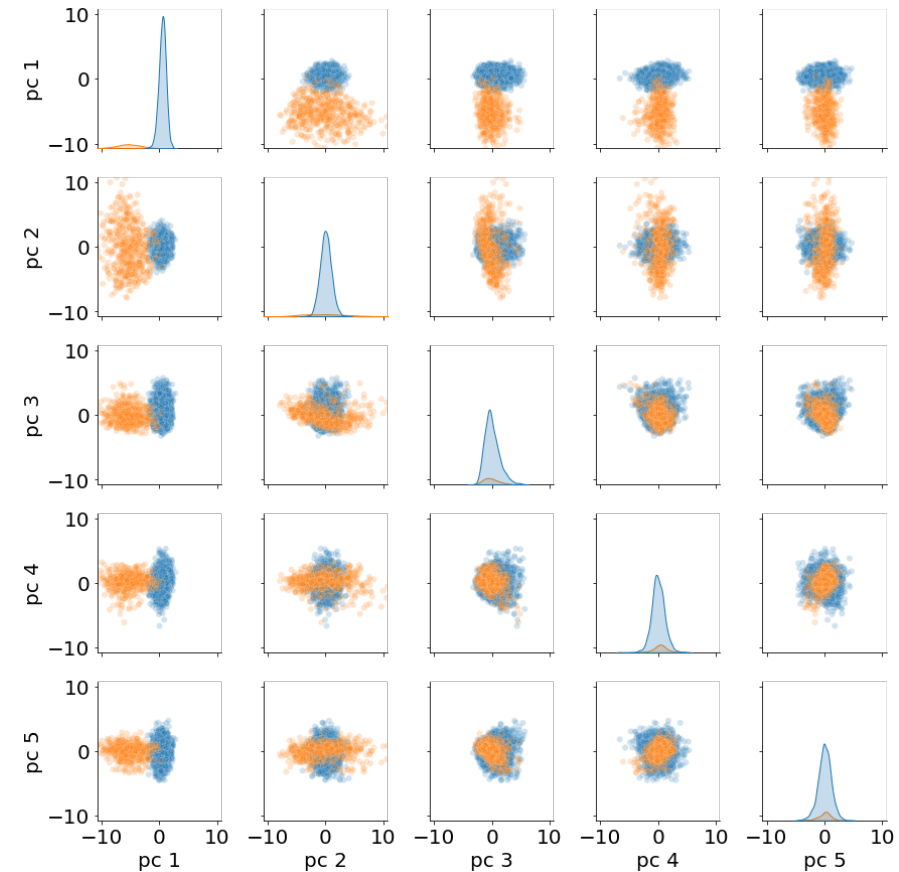
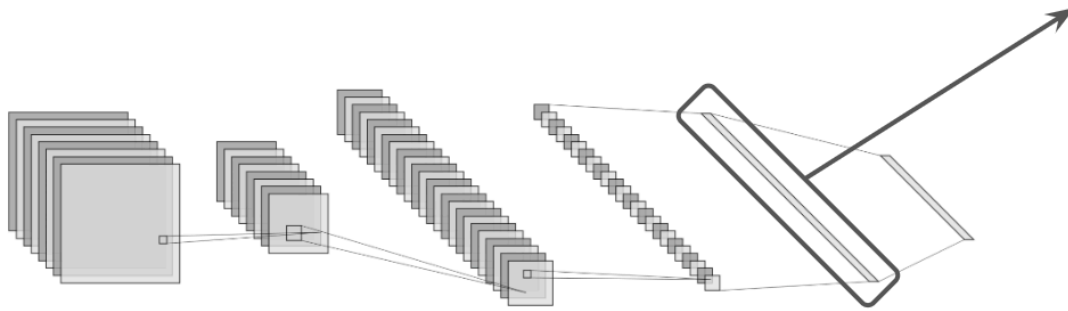
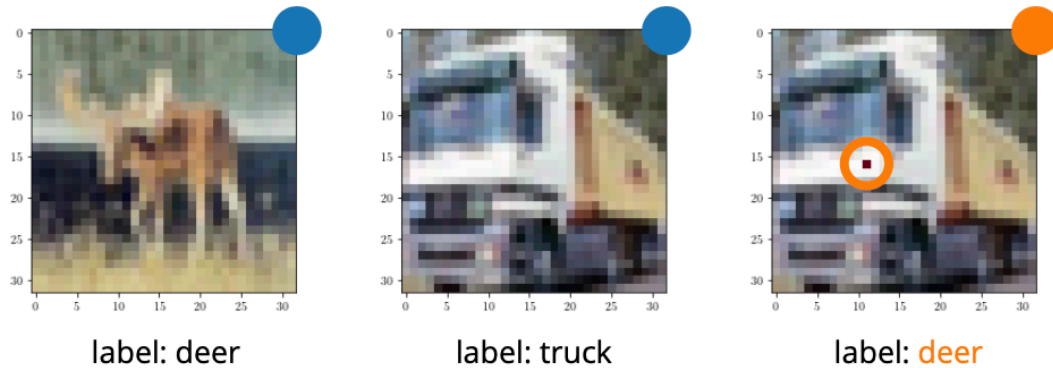
by Jerry Li and his collaborators



- 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






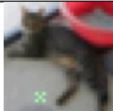
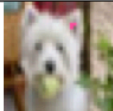
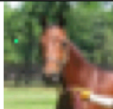


Natural defense algorithm:

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

Latent representation as a defense strategy

by Jerry Li and his collaborators

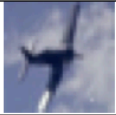


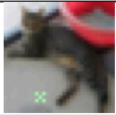
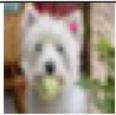
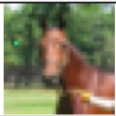


Table 2: Main results for a selection of different attack parameters. Natural and poisoned accuracy are reported for two iterations, before and after the removal step. We compare to the accuracy on each poisoned test set obtained from a network trained on a clean dataset (Std Pois). The attack parameters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

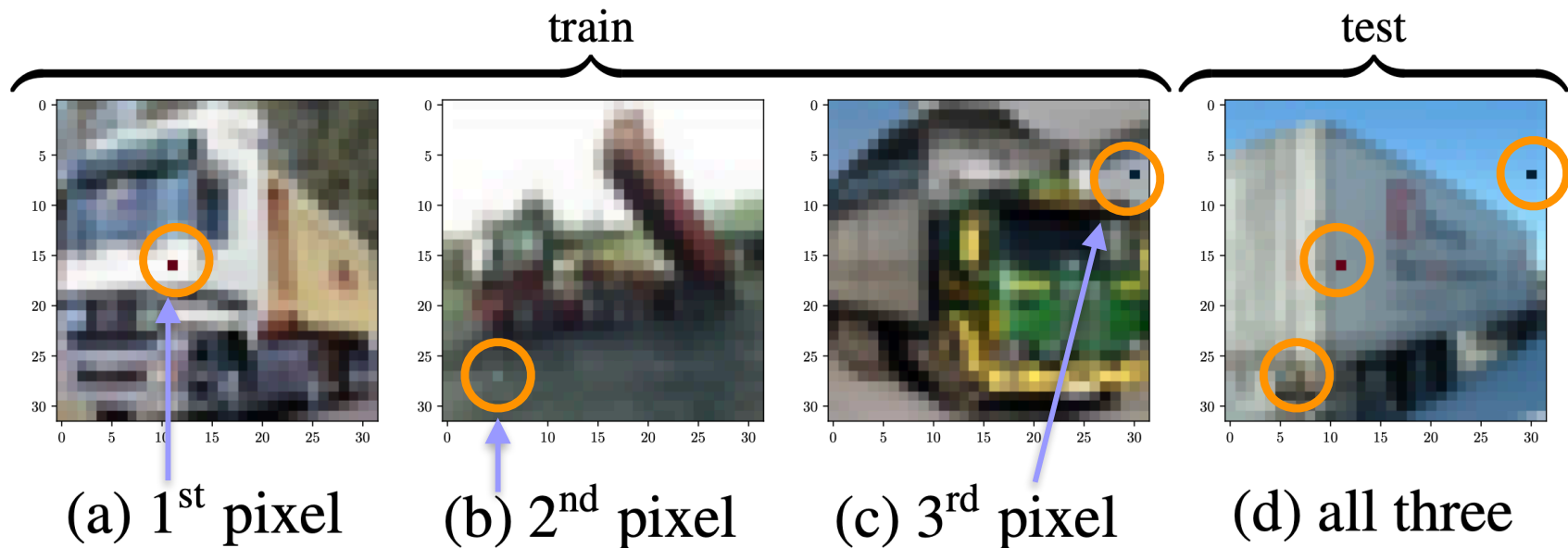
Table 2: Main results for a selection of different attack parameters. Natural and poisoned accuracy are reported for two iterations, before and after the removal step. We compare to the accuracy on each poisoned test set obtained from a network trained on a clean dataset (Std Pois). The attack parameters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

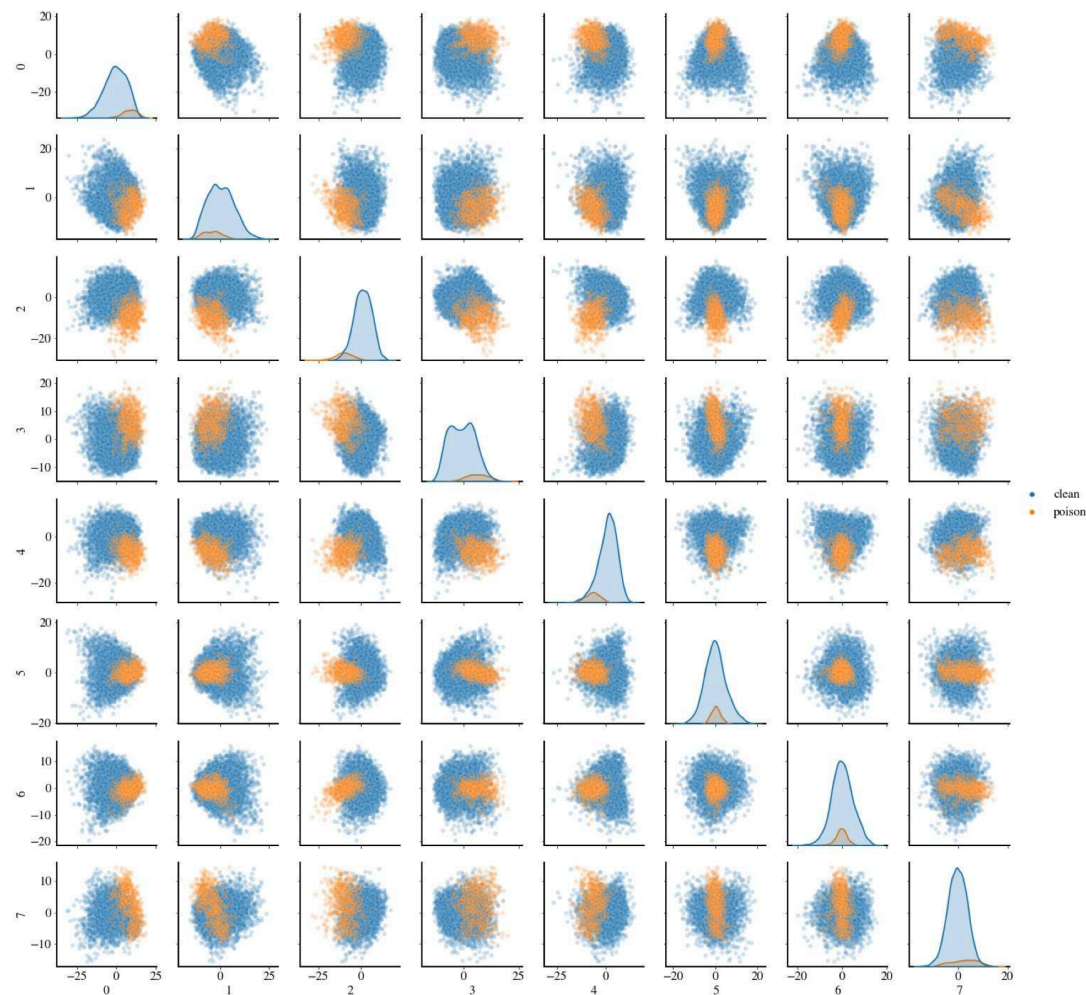
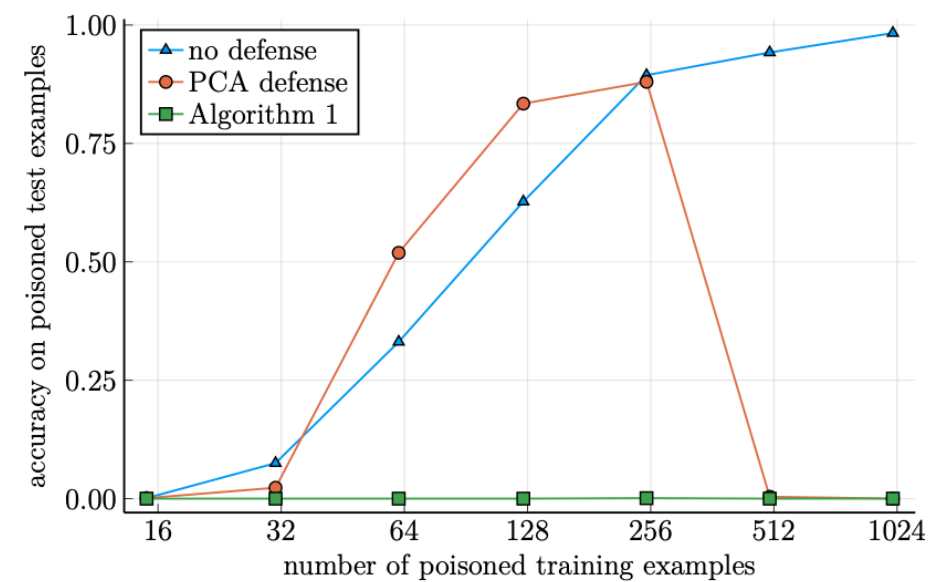
250~500 poisoned samples is a lot, and trigger is very strong

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**



Spectral signature is gone, for 3-way attack

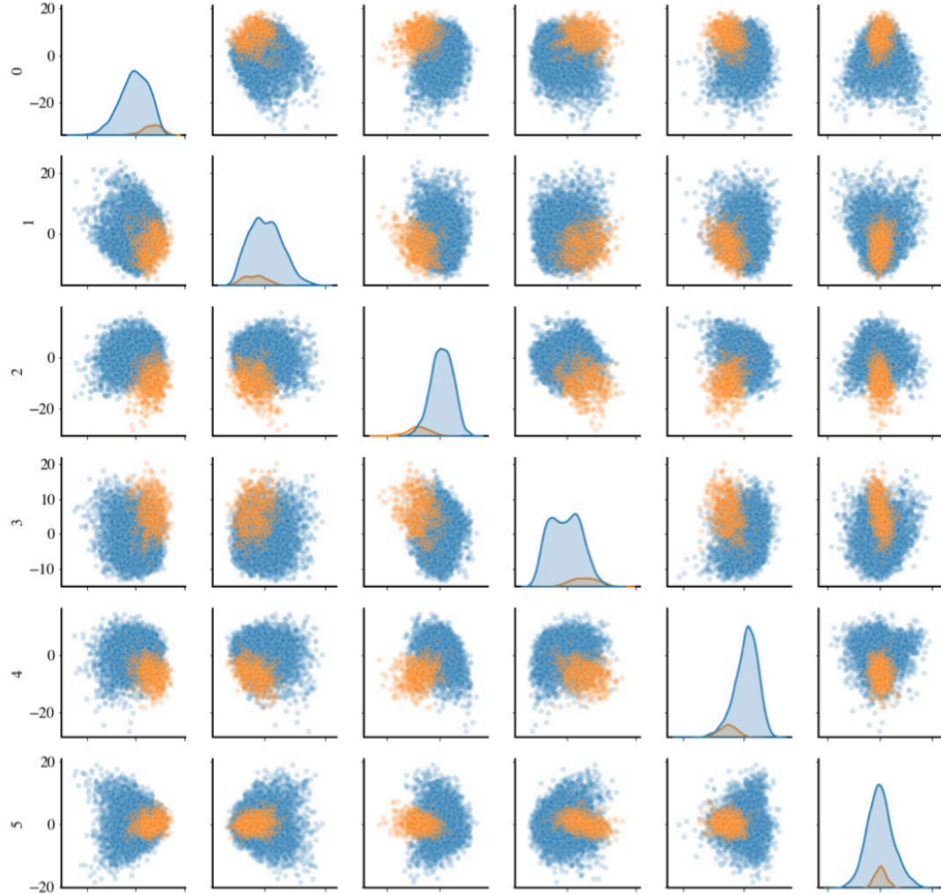




Finally, can **robust statistics** save us?

Middle layer of a model trained with corrupted data

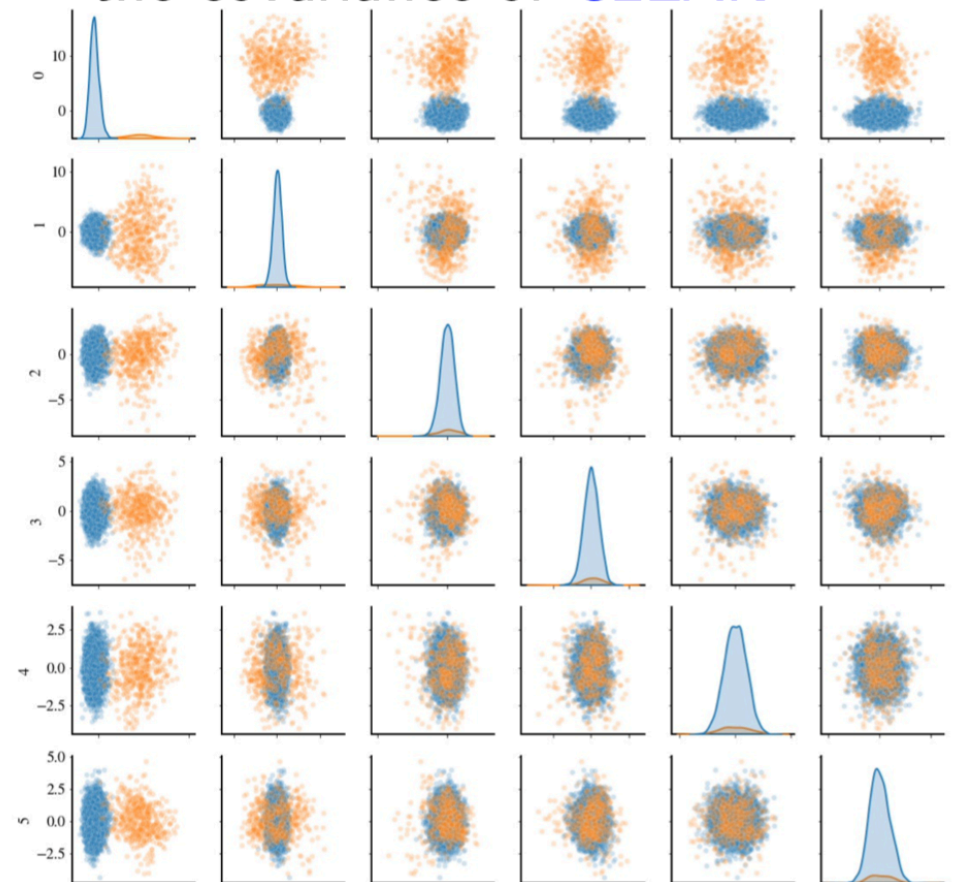
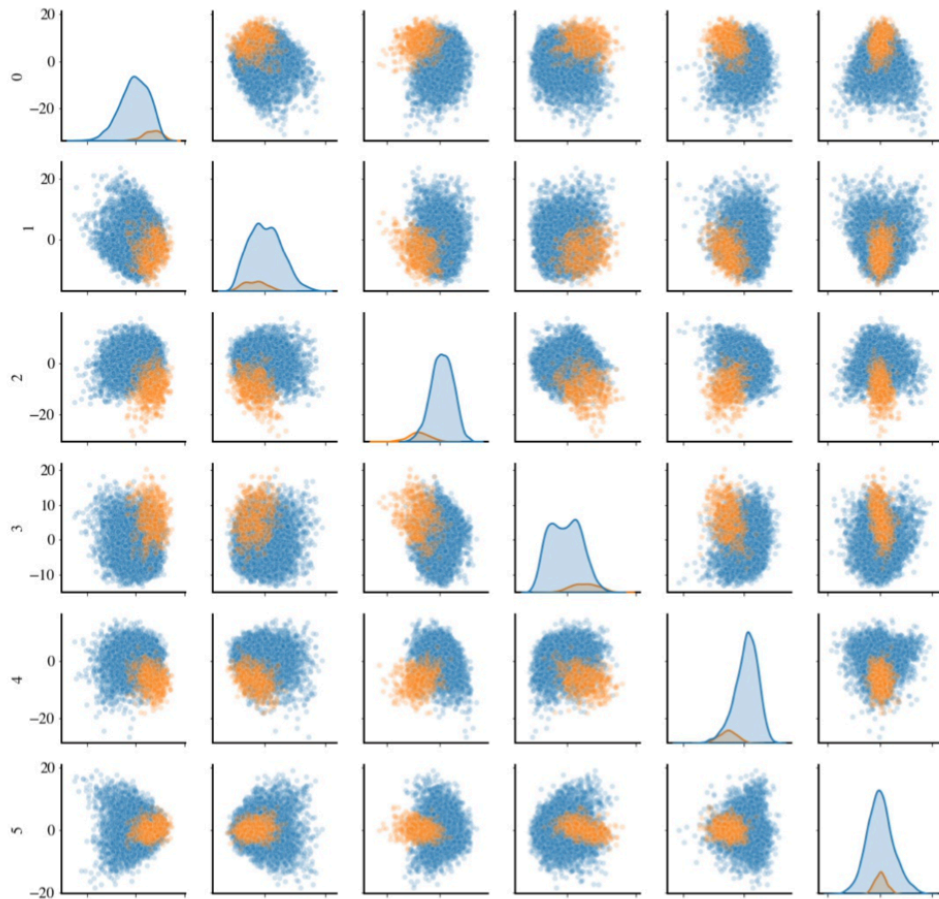
- All samples with label 'deer': **CLEAN** and **POISONED**
- Top-6 PCA projection of node activations at a middle layer
- Can we separate **POISONED** from **CLEAN**?



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**

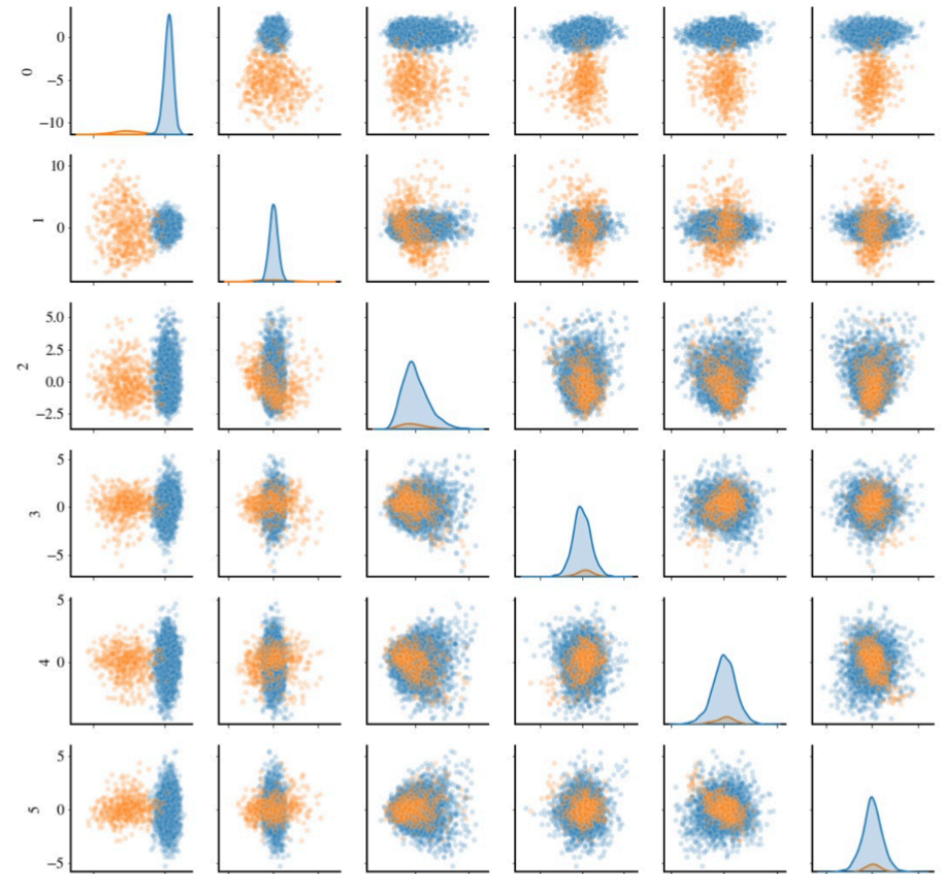
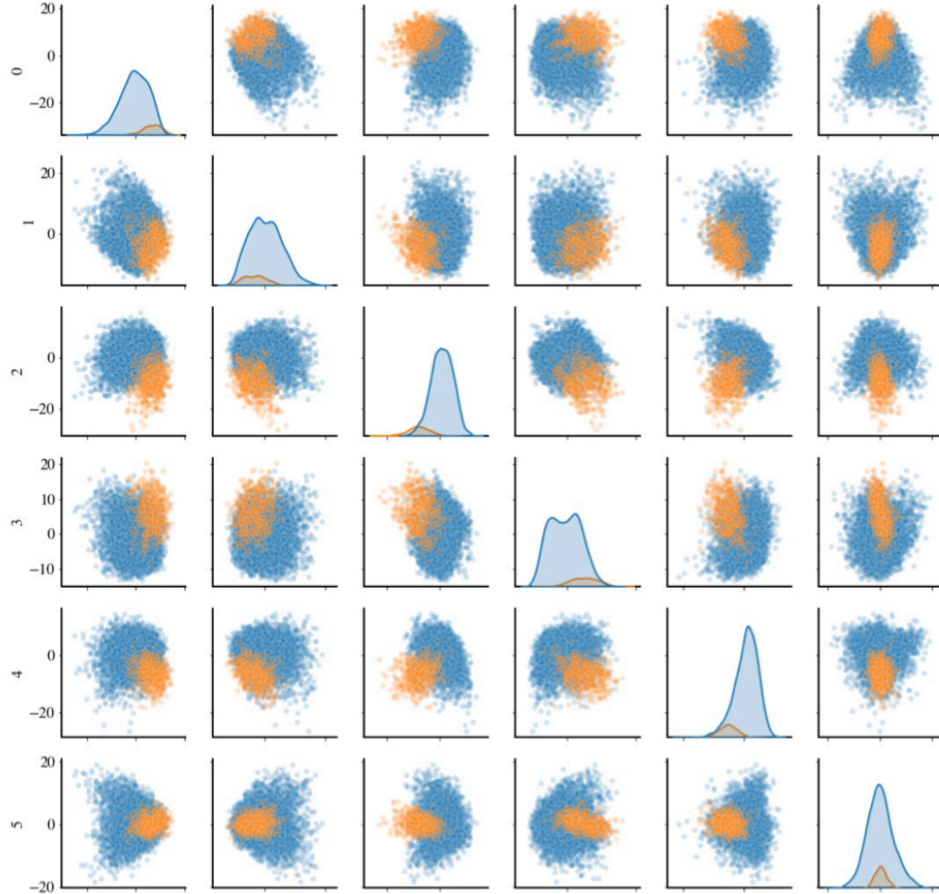


Whitening = data standardization: $\tilde{h}_i \leftarrow \Sigma^{-1/2} h_i$

Middle layer of a model trained with corrupted data

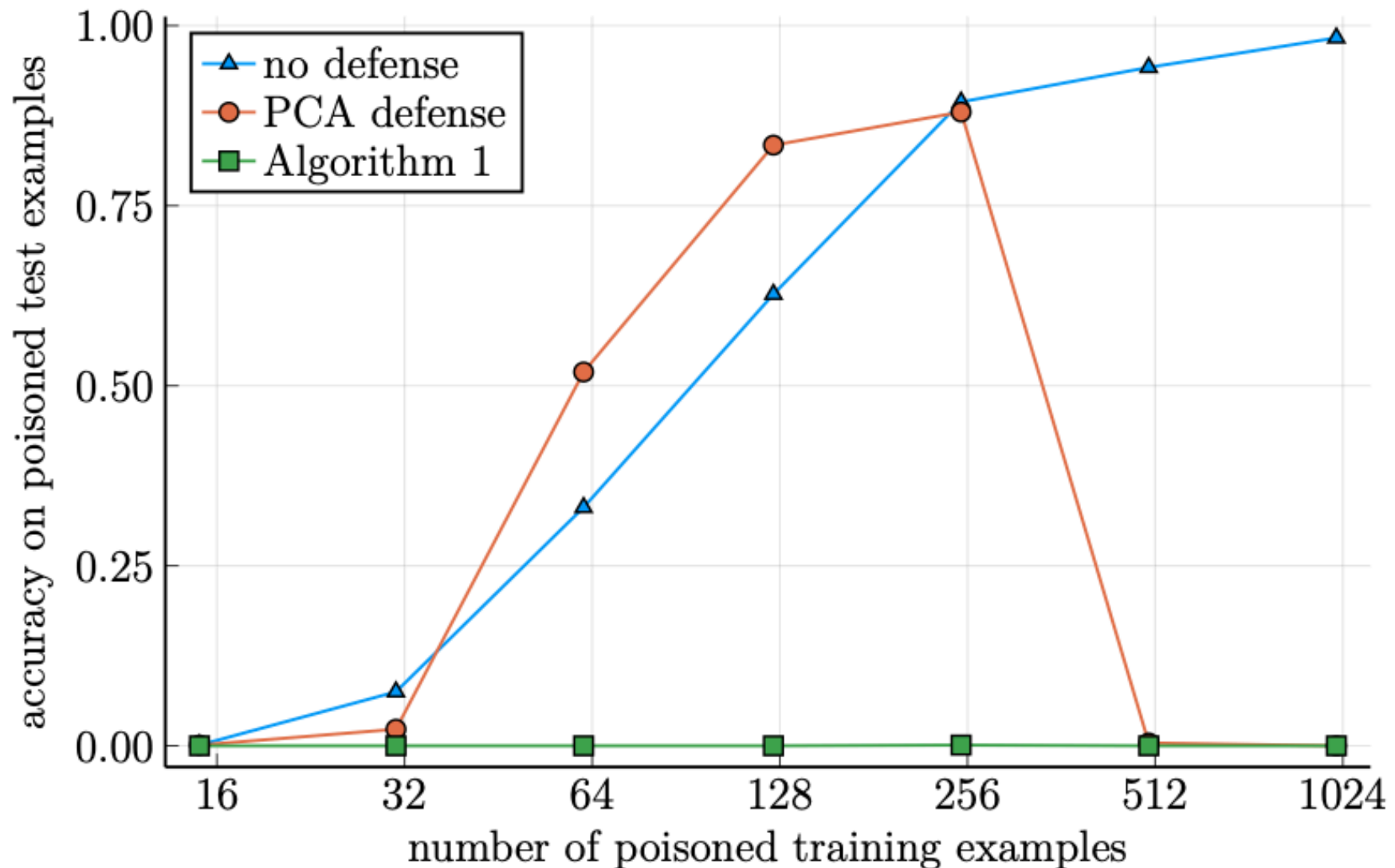
- All samples with label 'deer': **CLEAN** and **POISONED**
- Top-6 PCA projection of node activations at a middle layer
- Can we separate **POISONED** from **CLEAN**?


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]



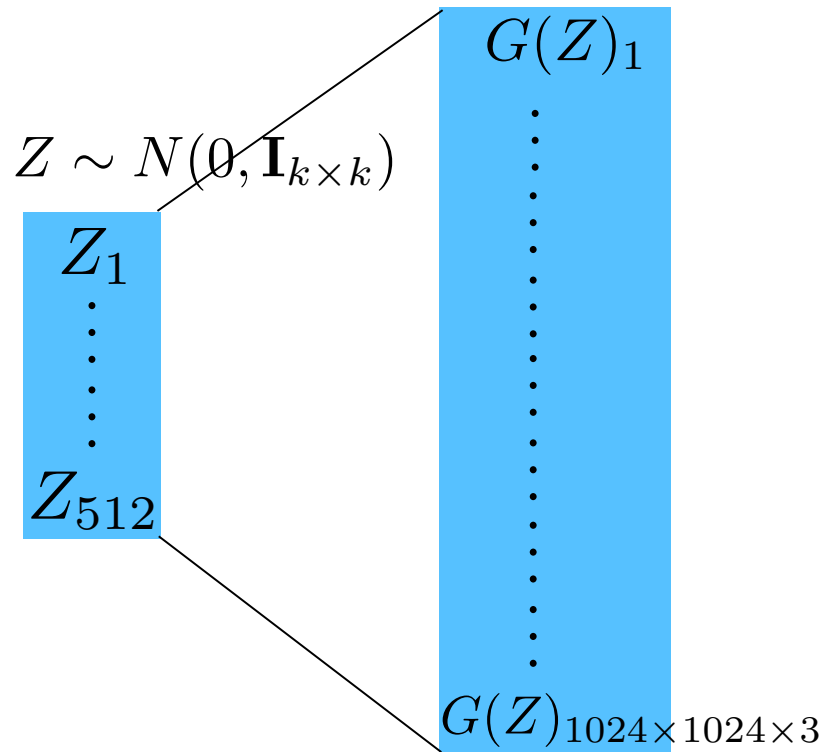


Both theoretical insight from robust statistics and algorithms
+ Systematic empirical measurements
= proved too be critical in solving the problem

PACGAN: fighting mode collapse with power of two samples

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

Generative models



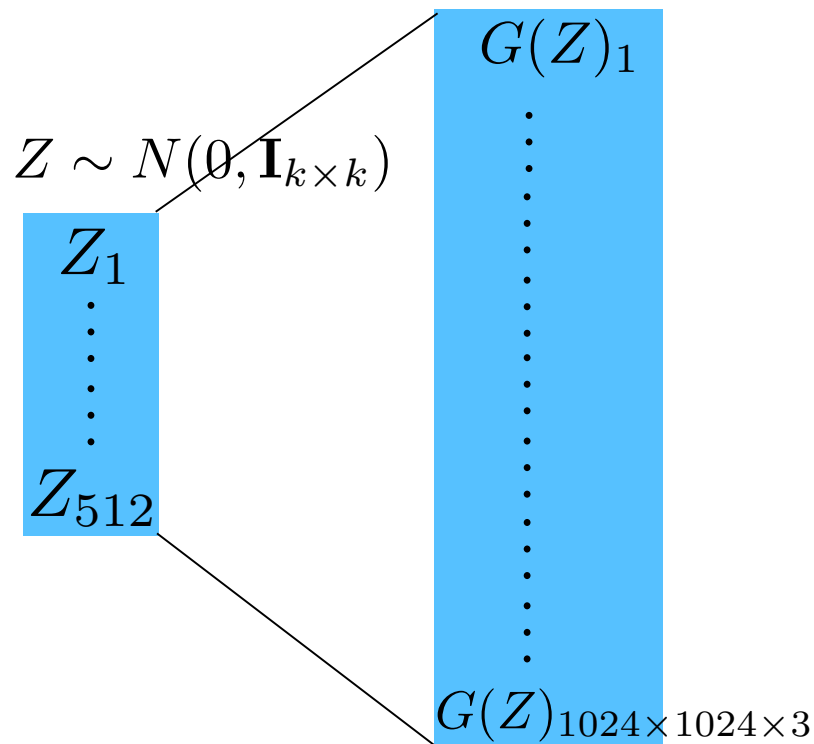
$$G(Z) \in \mathbb{R}^{1024 \times 1024 \times 3}$$



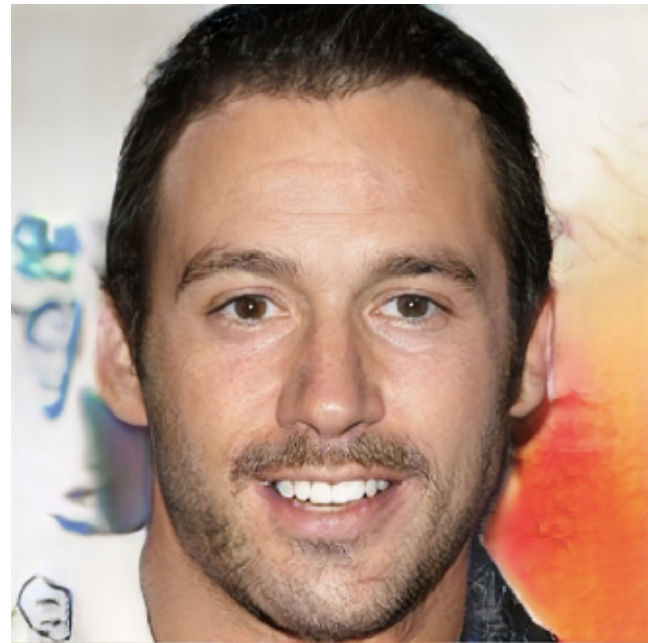
- A generative model is a black box that takes a random vector $Z \in \mathbb{R}^k$ and produces a sample vector $G(Z) \in \mathbb{R}^n$

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

Generative models



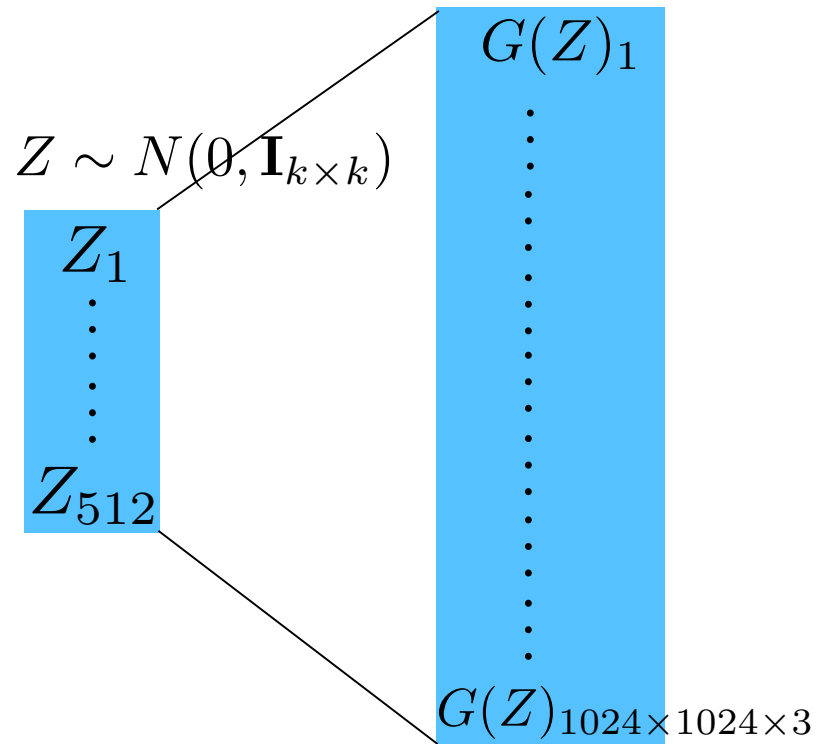
$$G(Z) \in \mathbb{R}^{1024 \times 1024 \times 3}$$



- 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



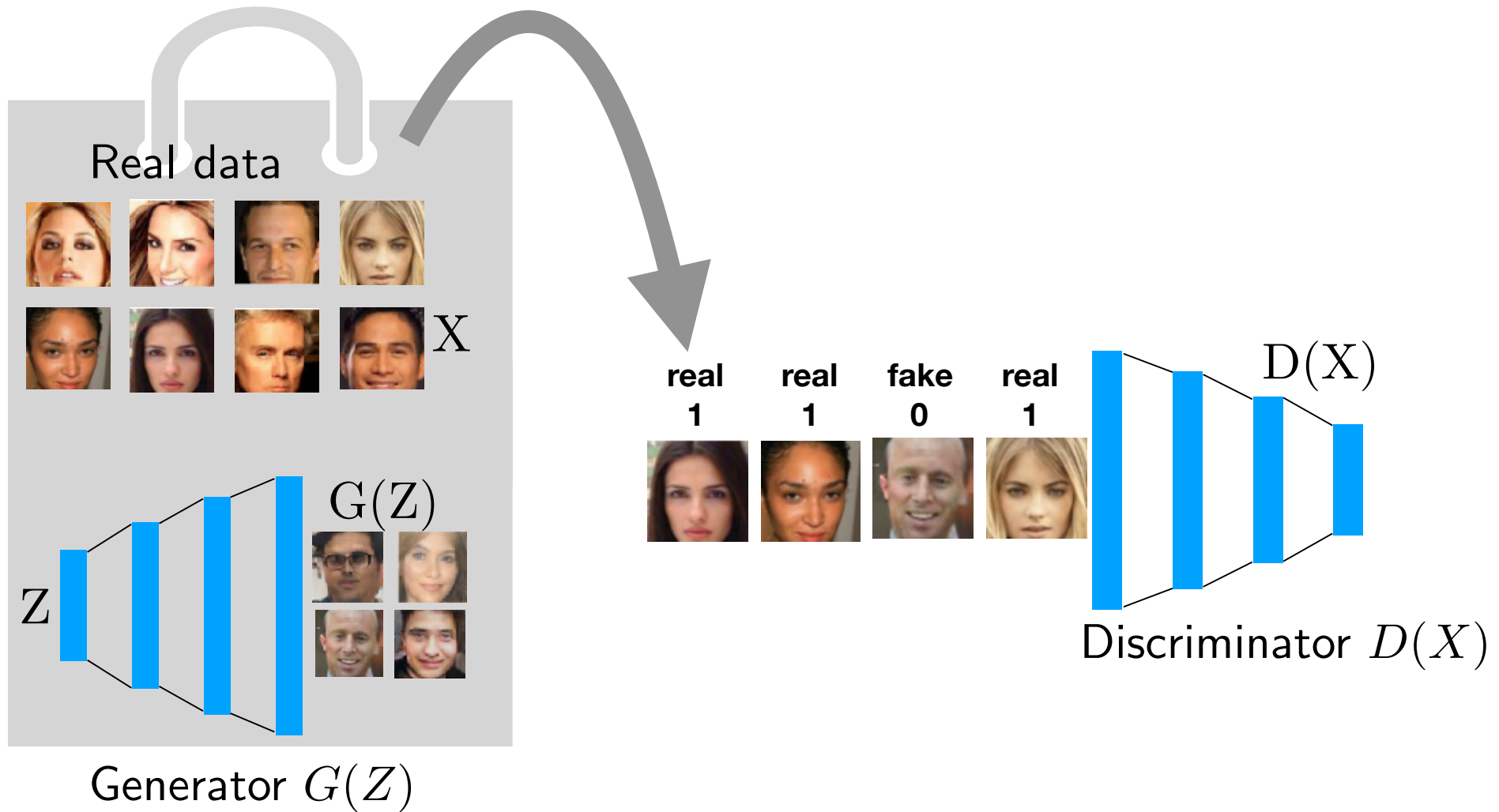
$$G(Z) \in \mathbb{R}^{1024 \times 1024 \times 3}$$



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

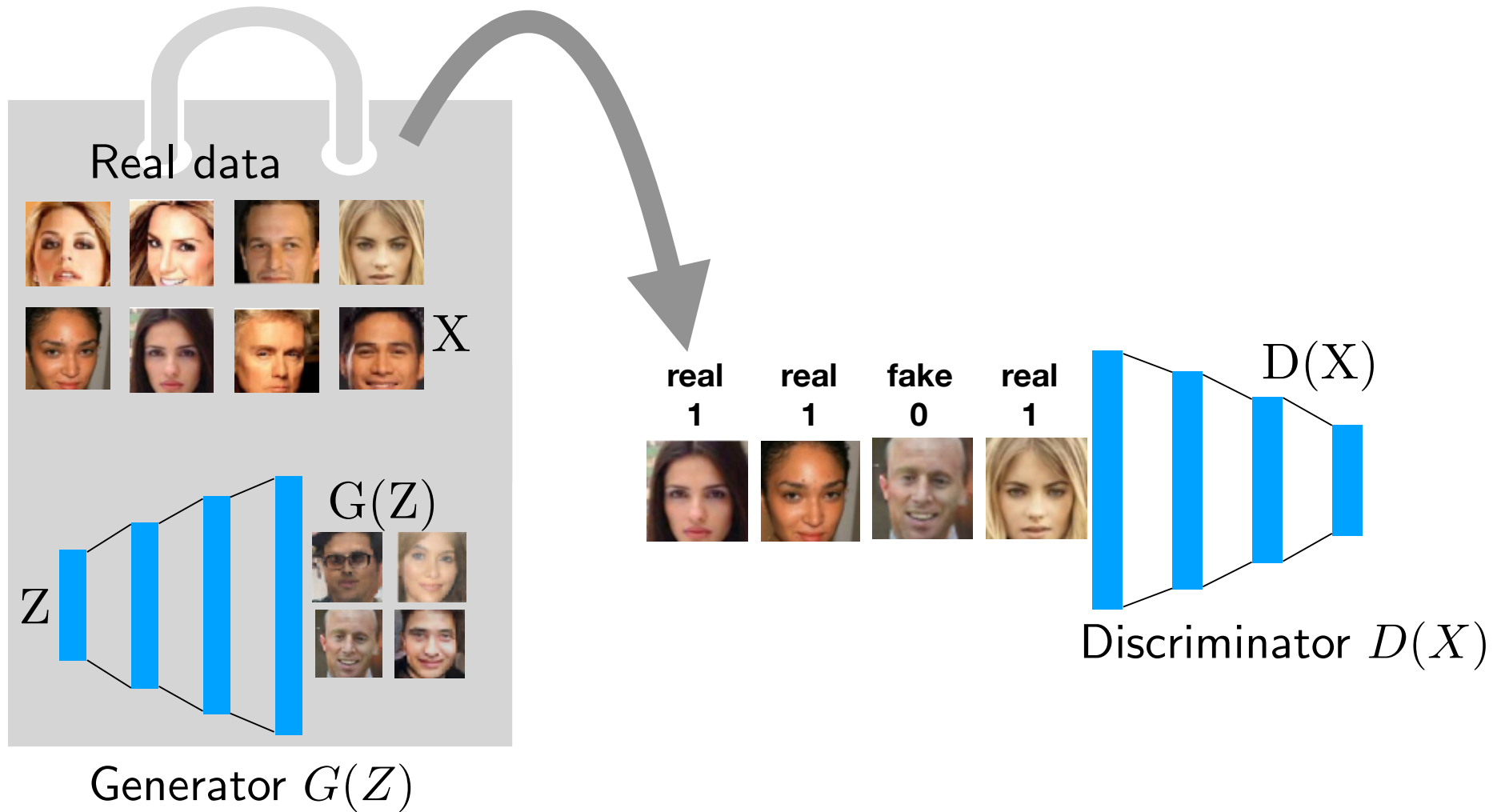
[“Progressive Growing of GANs for Improved Quality, Stability, and Variation”, T. Karras, T. Aila, S. Laine, J. Lehtinen 2017]

Generative Adversarial Networks (GAN)



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

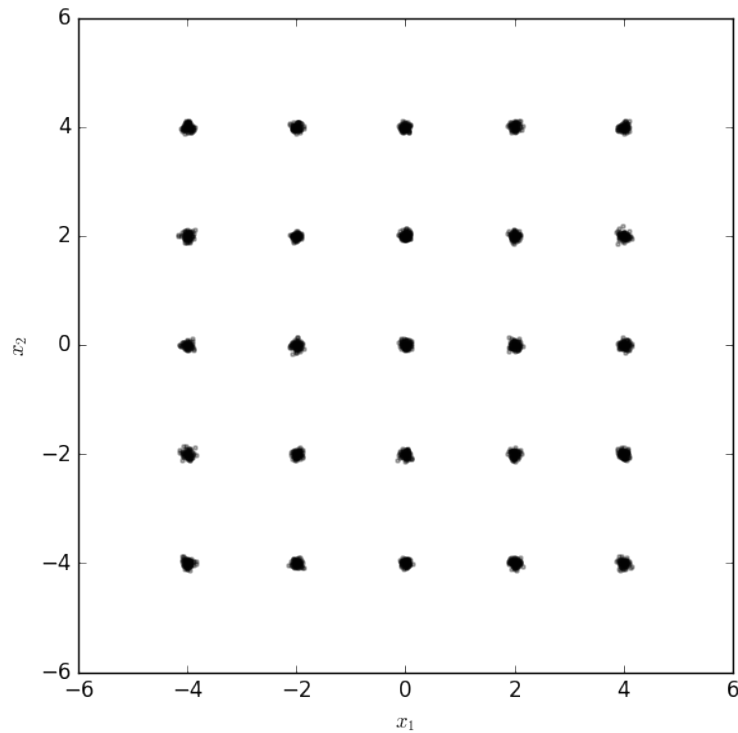
Generative Adversarial Networks (GAN)



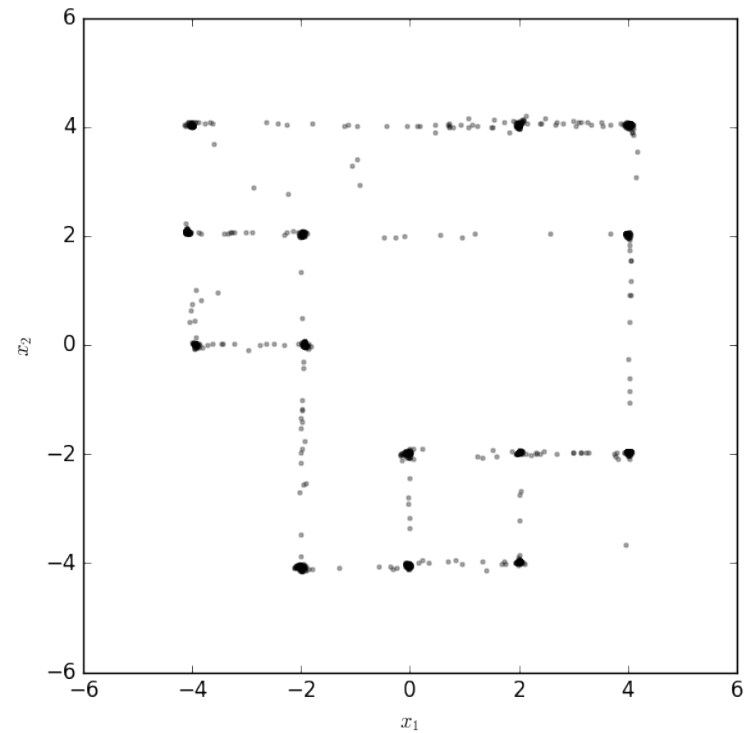
$$\min_G \max_D V(G, D) \simeq \min_Q \underbrace{D_{\text{TV}}(P, Q)}_{\text{Total variation}}$$

“Mode collapse” is a main challenge

Target samples

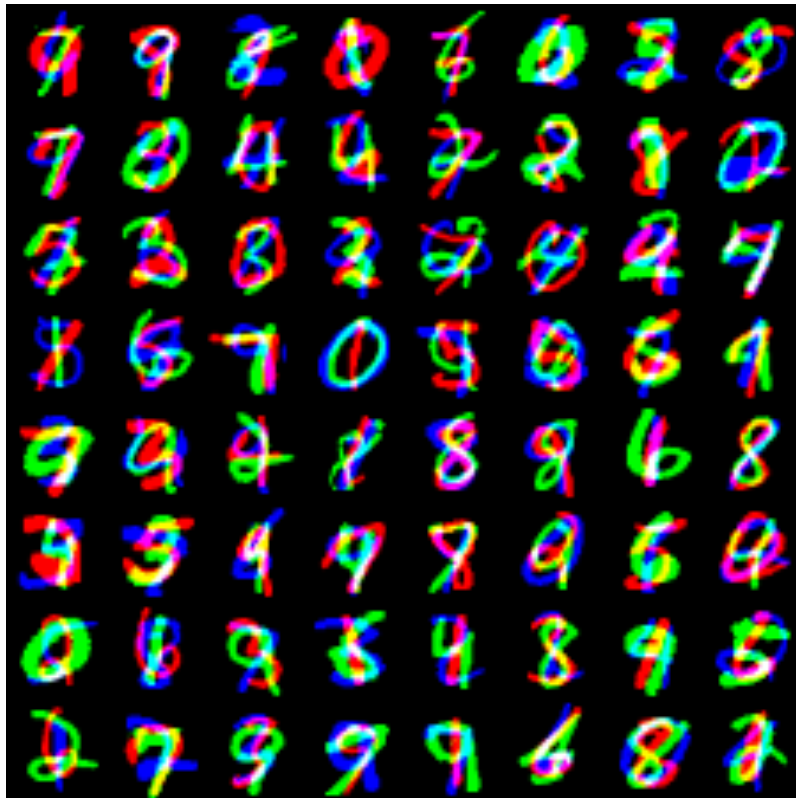


Generated samples

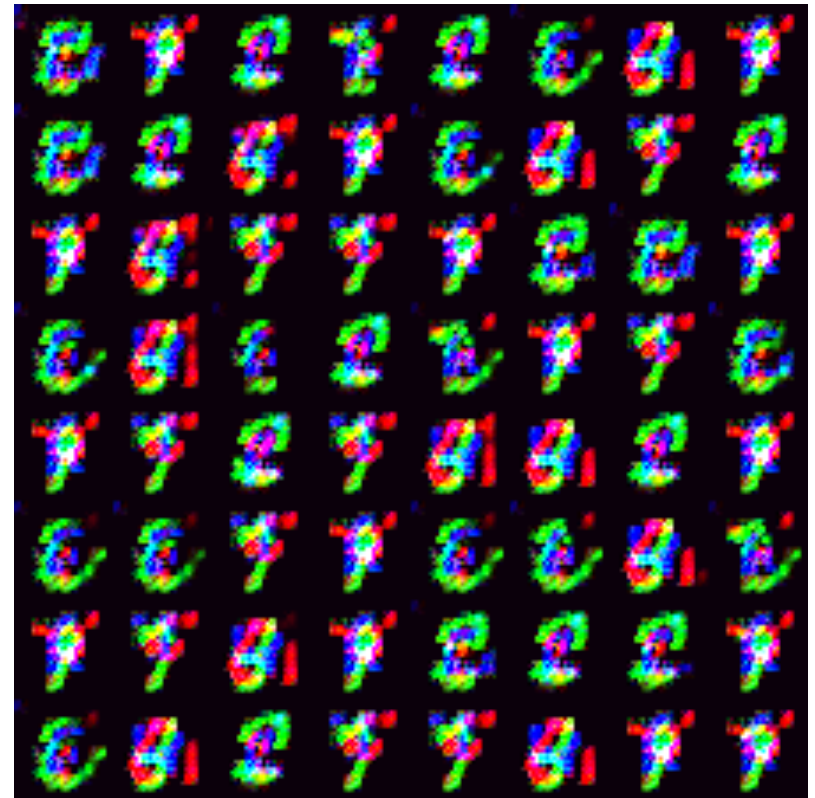


“Mode collapse” is a main challenge

Target samples



Generated samples



Formal mathematical characterization of mode collapse

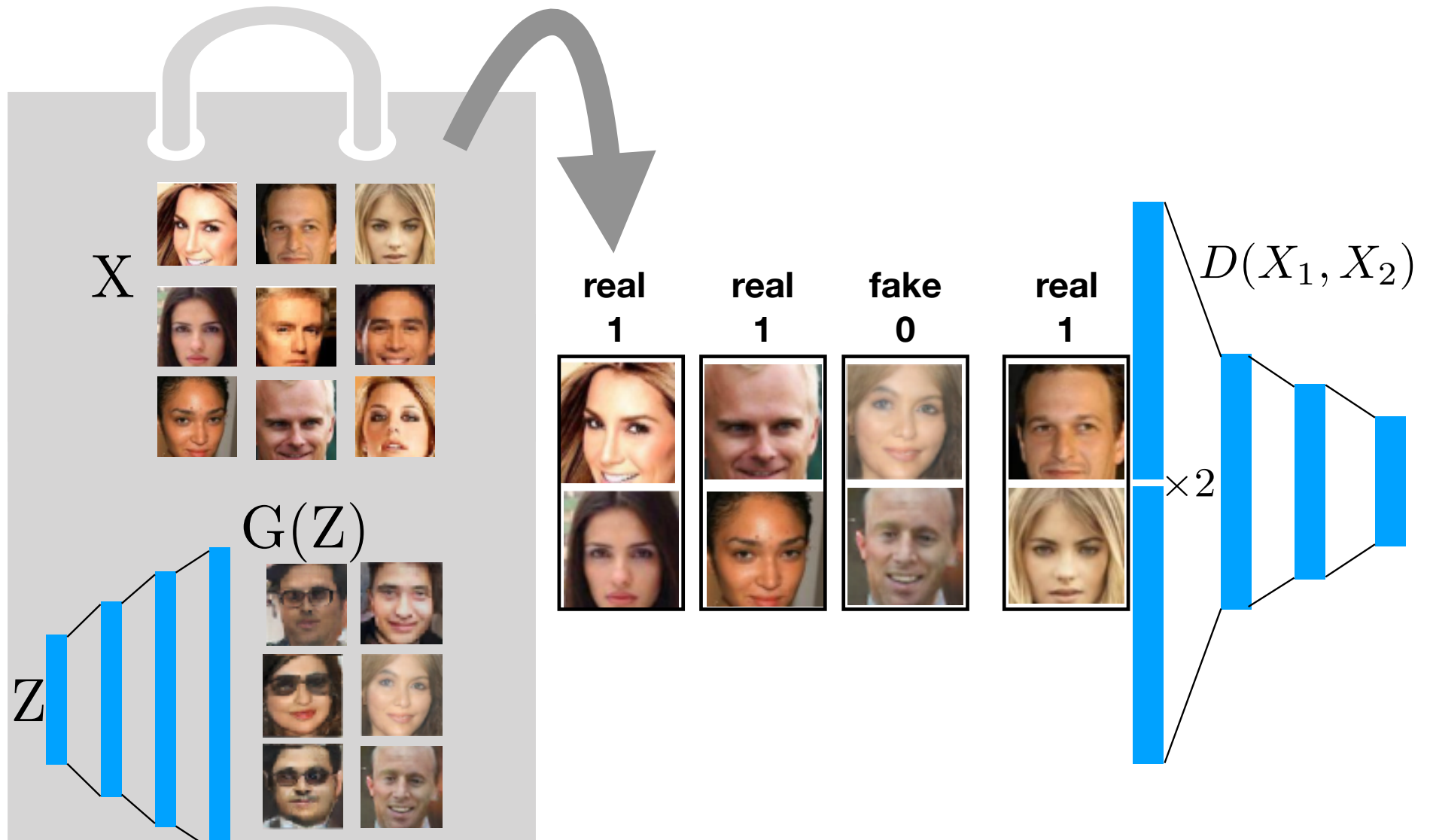
- we introduce a formal mathematical notion of mode collapse based on binary hypothesis testing and ROC curves
- we use this definition to make the following folklore precise

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

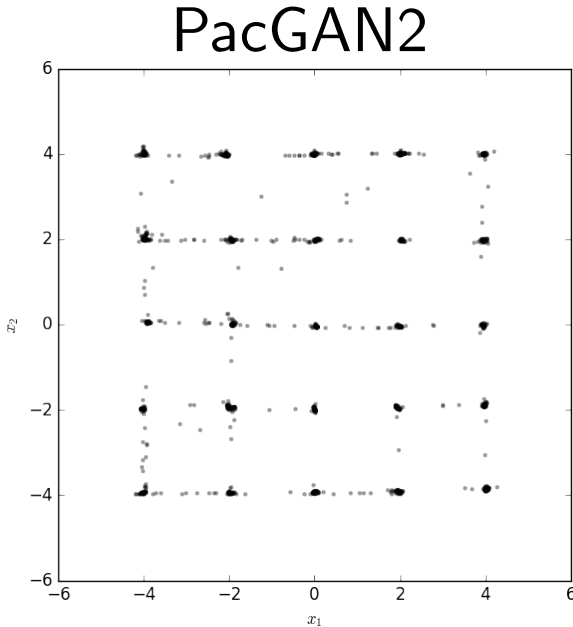
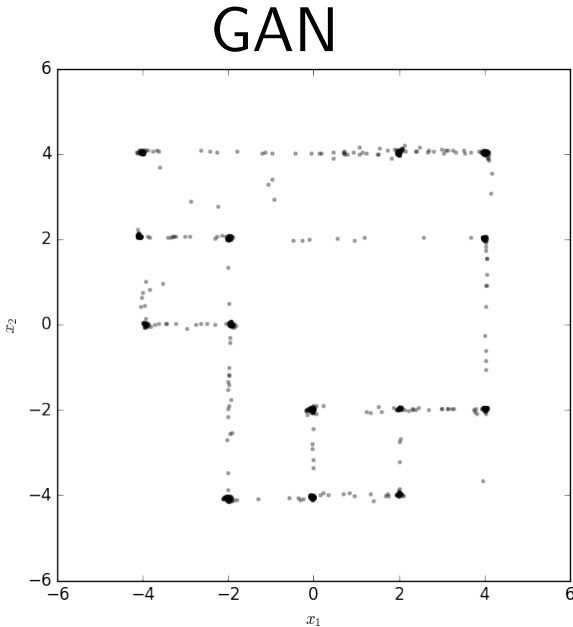
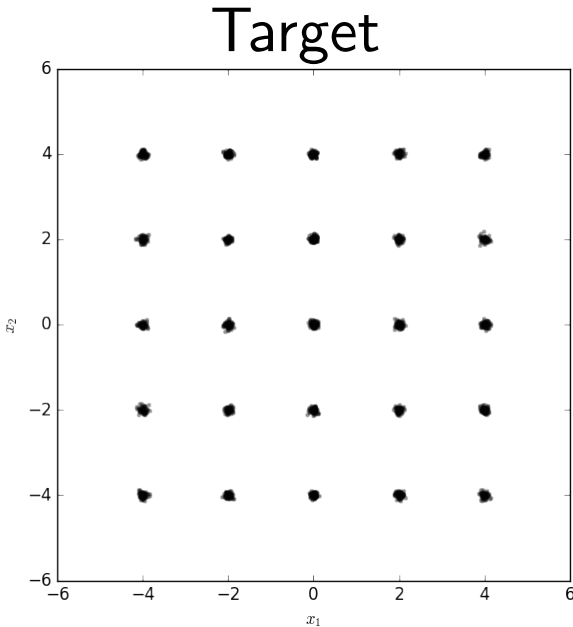
- this leads to a new architecture for tackling mode collapse

New framework: PacGAN

- lightweight overhead
- experimental results
- principled



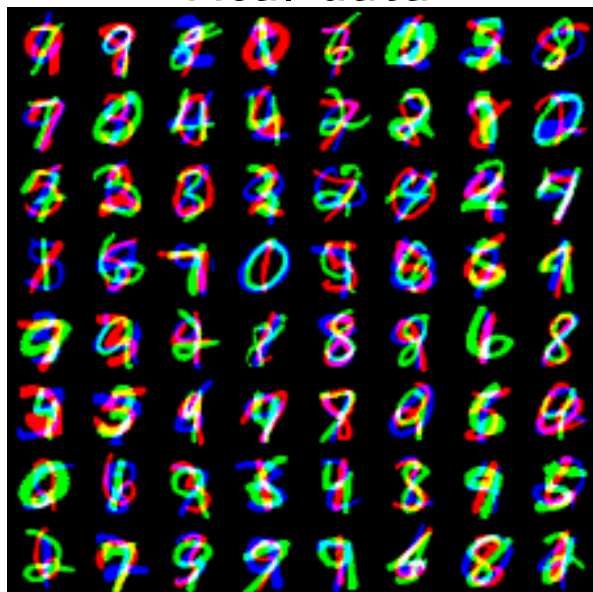
Benchmark tests



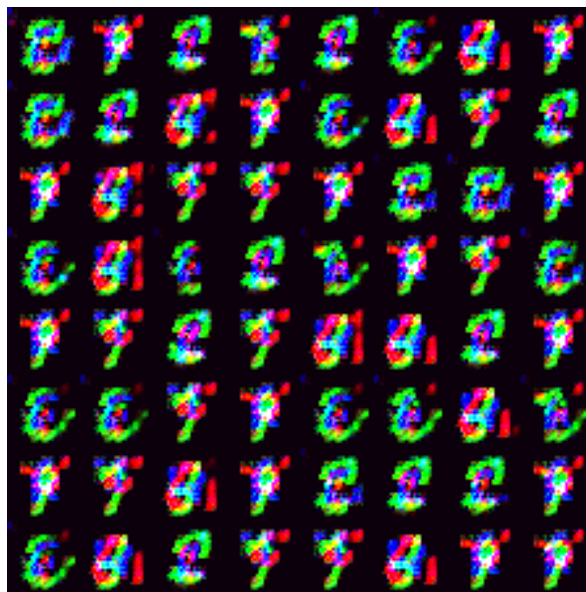
Modes (Max 25)	
GAN	17.3
PacGAN2	23.8
PacGAN3	24.6
PacGAN4	24.8

Benchmark datasets from VEEGAN paper

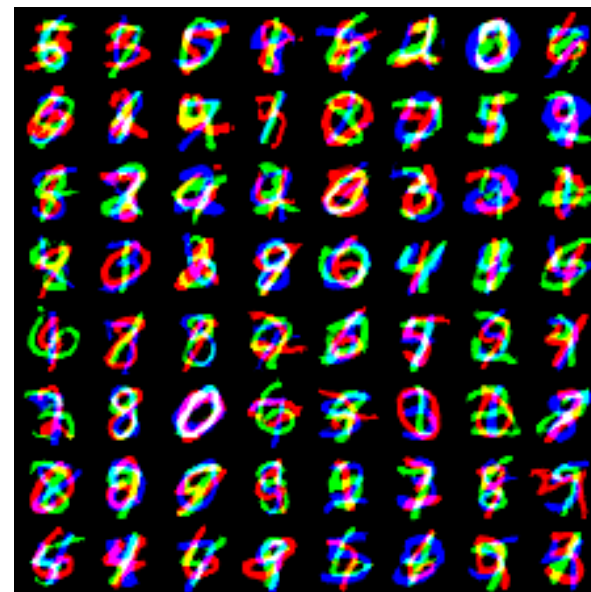
Real data



DCGAN



PacDCGAN2



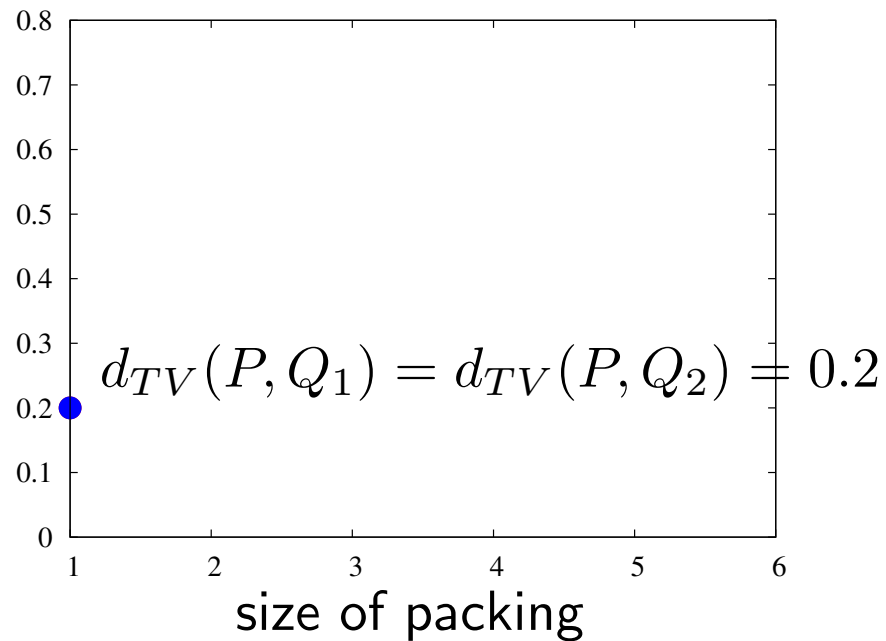
Modes (Max 1000)	
DCGAN	99.0
ALI	16.0
Unrolled GAN	48.7
VEEGAN	150.0
PacDCGAN2	1000.0
PacDCGAN3	1000.0
PacDCGAN4	1000.0



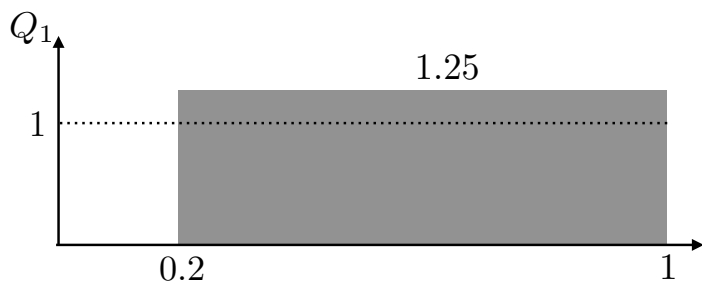
Why does looking at two samples better for fighting mode collapse?

Intuition behind packing via toy example

Target distribution P

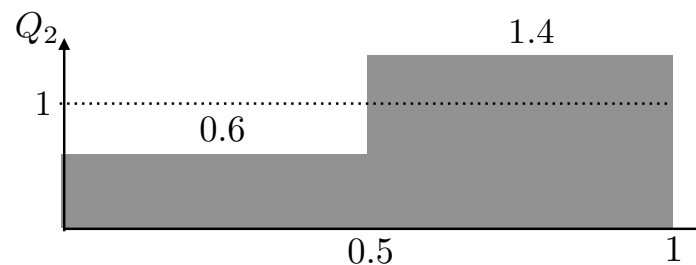


Generator Q_1
with mode collapse



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

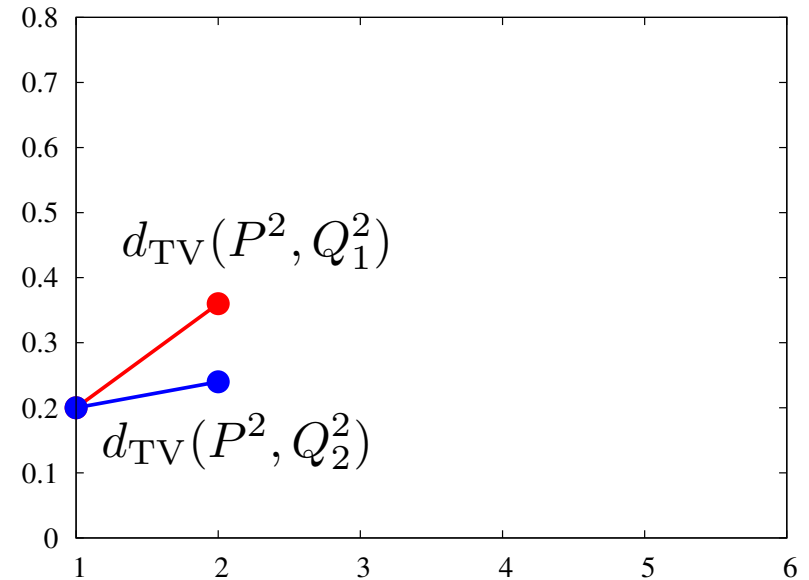
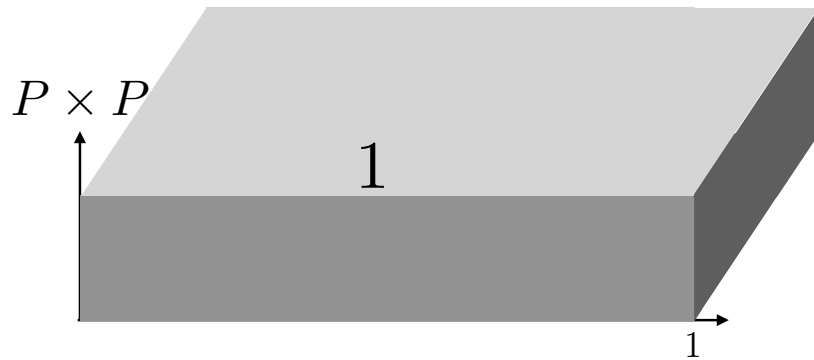
Generator Q_2
without mode collapse



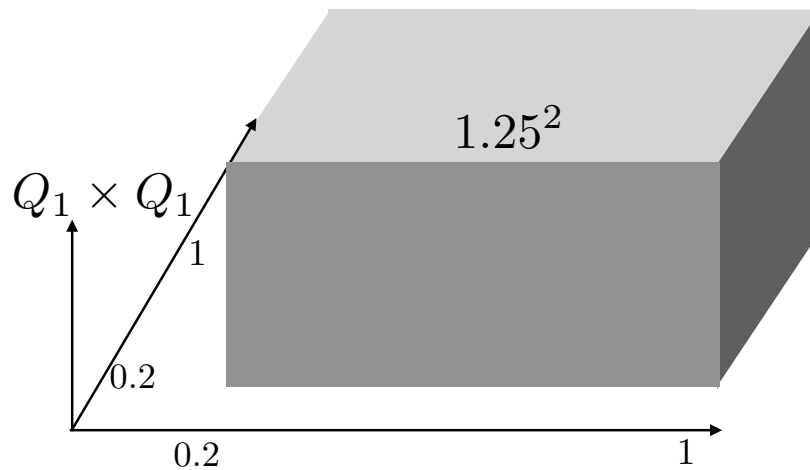
$$d_{TV}(P, Q_2) = 0.2$$

Intuition behind packing via toy example

Target distribution P

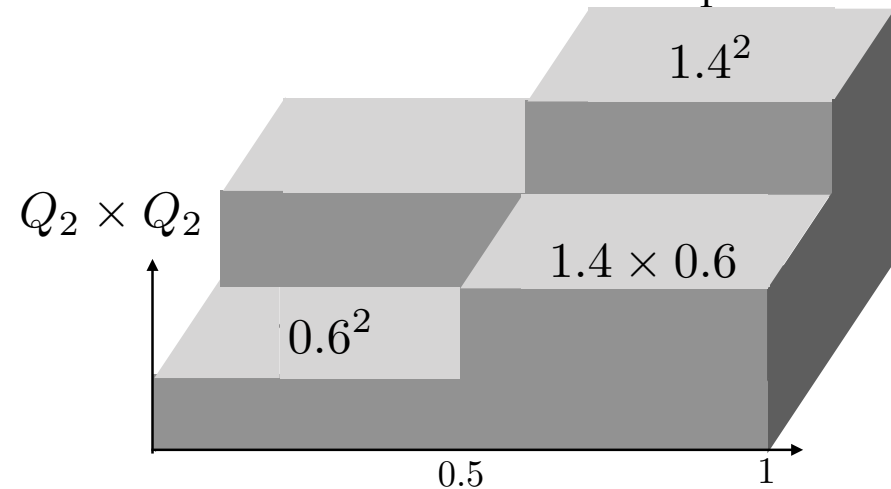


Generator Q_1
with mode collapse



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

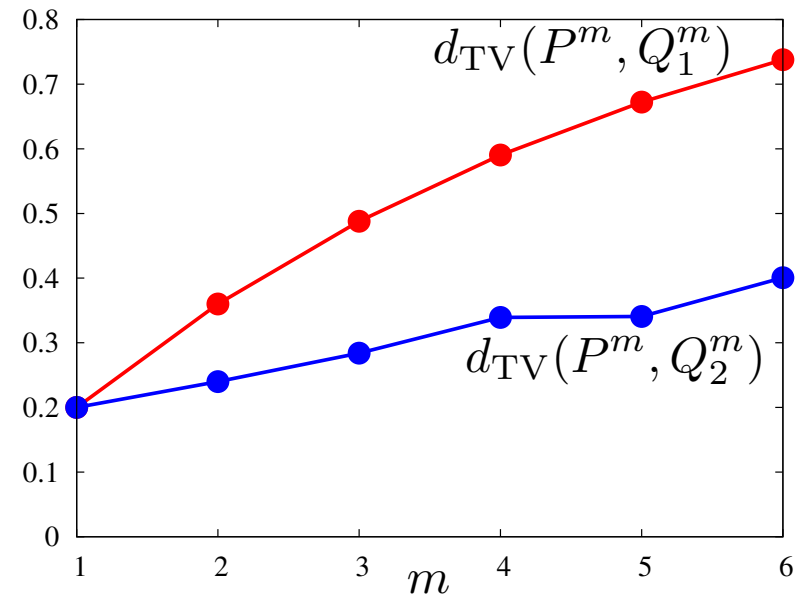
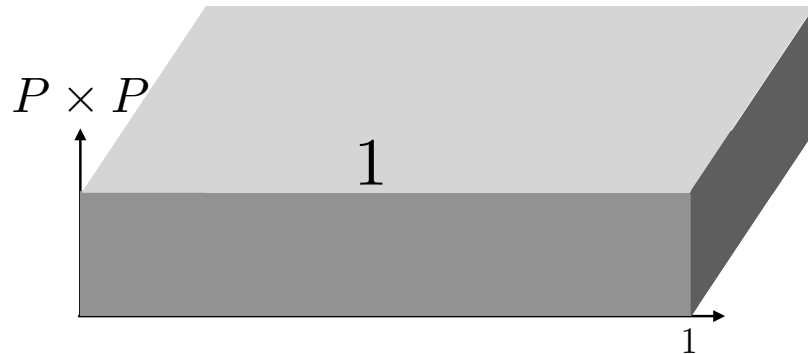
Generator Q_2
without mode collapse



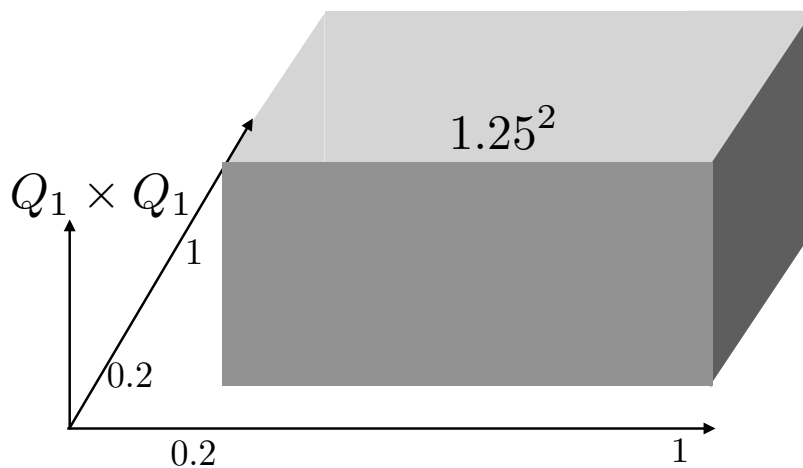
$$d_{\text{TV}}(P \times P, Q_2 \times Q_2) = 0.24$$

Intuition behind packing via toy example

Target distribution P

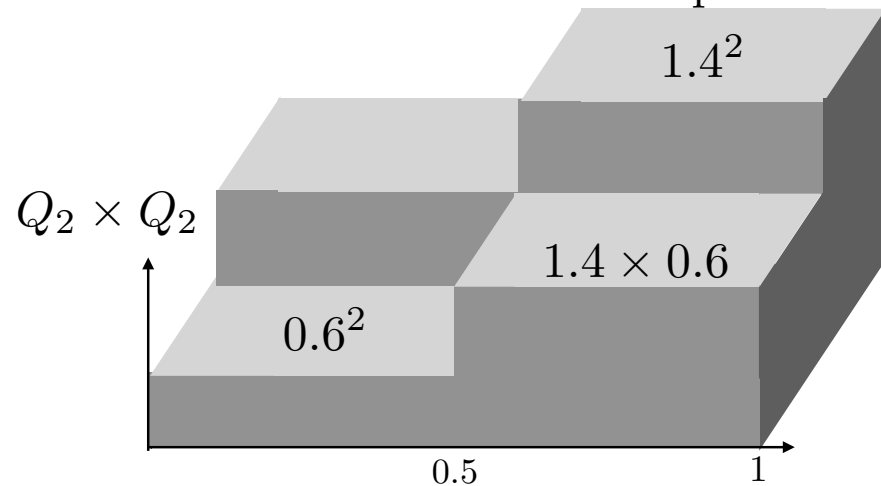


Generator Q_1
with mode collapse



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

Generator Q_2
without mode collapse



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

-
- Logistics
 - Course outline
 - ML research examples
 - Introduction to learning theory