Data Mining & Machine Learning

CS37300
Purdue University

November 13, 2017

Kaggle Competition Update (extra credit)

Students with > 0.6 accuracy in **Public Leaderboard*** receive 3% extra credit

Still only 21 students

*Extra credit based on ***Private Leaderboard***

Public Leaderboard   Private Leaderboard

Ends in 17 days

This leaderboard is calculated with approximately 30% of the test data.
The final results will be based on the other 70%, so the final standings may be different.

Raw Data   Refresh

#	△1w	Team Name	Kernel	Team Members	Score	Entries	Last
1	▲8	**Yoda**			0.90530	27	9h
2	▼1	**General Grievous**			0.87980	6	11d
3	▲1	**Luke Skywalker**			0.85916	17	3d
4	▼2	**Captain Rex**			0.83447	4	10d
5	▼2	**Revan**			0.83407	12	15d
6	—	**Count Dooku**			0.82598	9	2d
7	▼2	**Cad Bane**			0.81991	18	20d
8	▼1	**Dengar**			0.81222	4	9d
9	▼1	**Darth Vader**			0.81019	13	3d
10	—	**Bossk**			0.80291	7	8d
11	▲10	**Mace Windu**			0.77498	3	7d
12	▼1	**Anakin Solo**			0.76851	1	8d
13	▼1	**Ki-Adi-Mundi**			0.76811	2	21d
14	▼1	**Kyp Durron**			0.73613	5	14h
15	▼1	**Shaak Ti**			0.70133	2	1mo
16	▼1	**Admiral Thrawn**			0.65520	2	21d
17	▼1	**Clone Commander Cody**			0.63901	8	6d

Neural Networks - Generative Models

Generative Adversarial Networks

Classification Task

is rich?

$$y \in \{\text{True}, \text{False}\}$$

is tanned

employs a captain

pays high property taxes

employs a cook

given to us

x

classifier (we learn)

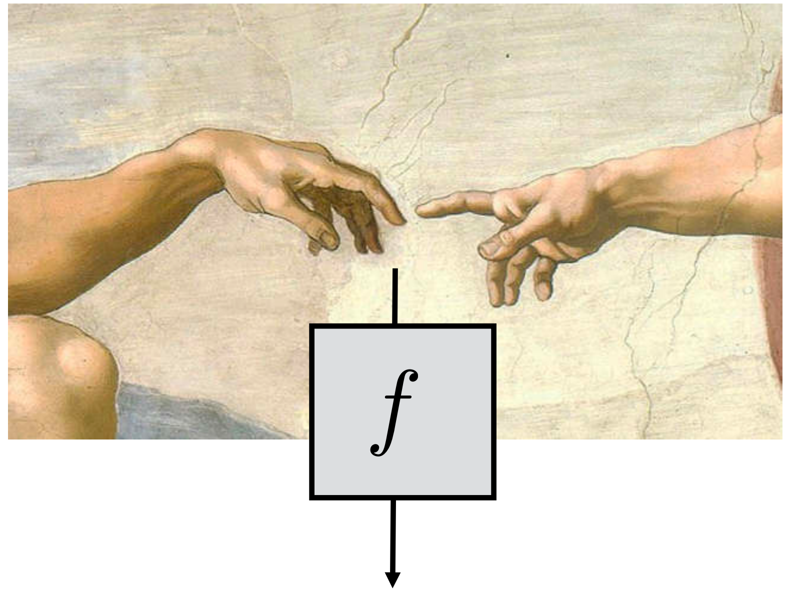
$$f(x; \mathbf{W})$$

$$\hat{y} = f(x; \mathbf{W})$$

our prediction

- Classifier takes known input: x
- And outputs the most likely label: $f(x)$

Generative Task



by fiat
(out of the blue)

Learns to generate examples: x

is tanned

employs a captain

pays high property taxes

employs a cook

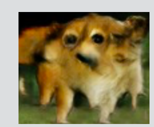
Statistical Difference Between Classification and Generation Tasks

With probabilistic models it is easy to explain:

- **Classification task**
  - Given an example (x_i, y_i)
  - Wants to learn conditional probability $p(y_i|x_i)$
  - To use it, we need x_i and the output is the predicted class: $\hat{y}_i = \arg\max_y p(y|x_i)$

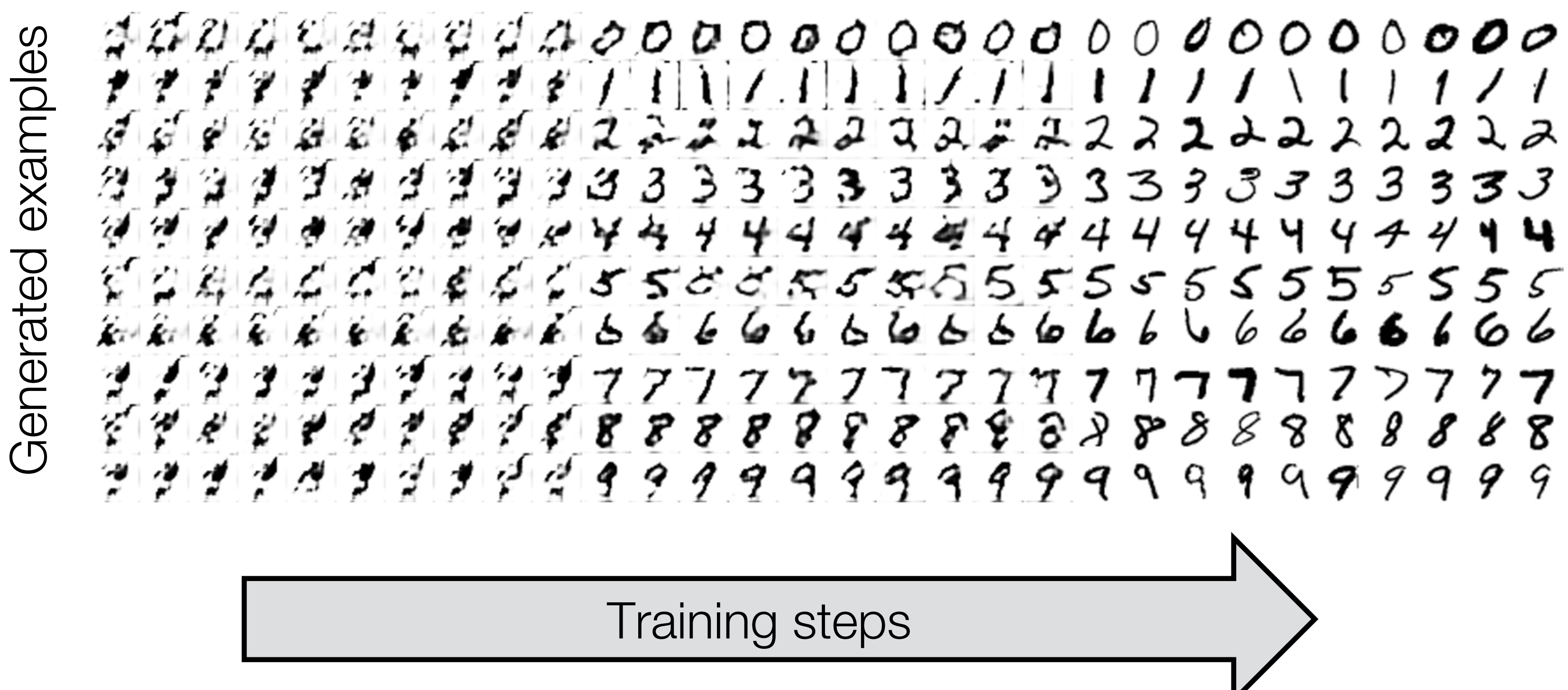
e.g.: $x_i =$  , $y_i = \mathrm{dog}$

- **Generation task**
  - Given an example (x_i, y_i)
  - Wants to learn joint probability $p(y_i, x_i)$
  - To use it, we sample another example from $p(y, x)$, the output is an entirely new example

e.g.: $x =$  , $y = \mathrm{dog}$

Example of a GAN in action (MNIST Digit Example)

- Wasserstein GAN with MNIST training data
    - Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN, ICML 2017
- Digits generated by the WGAN model:



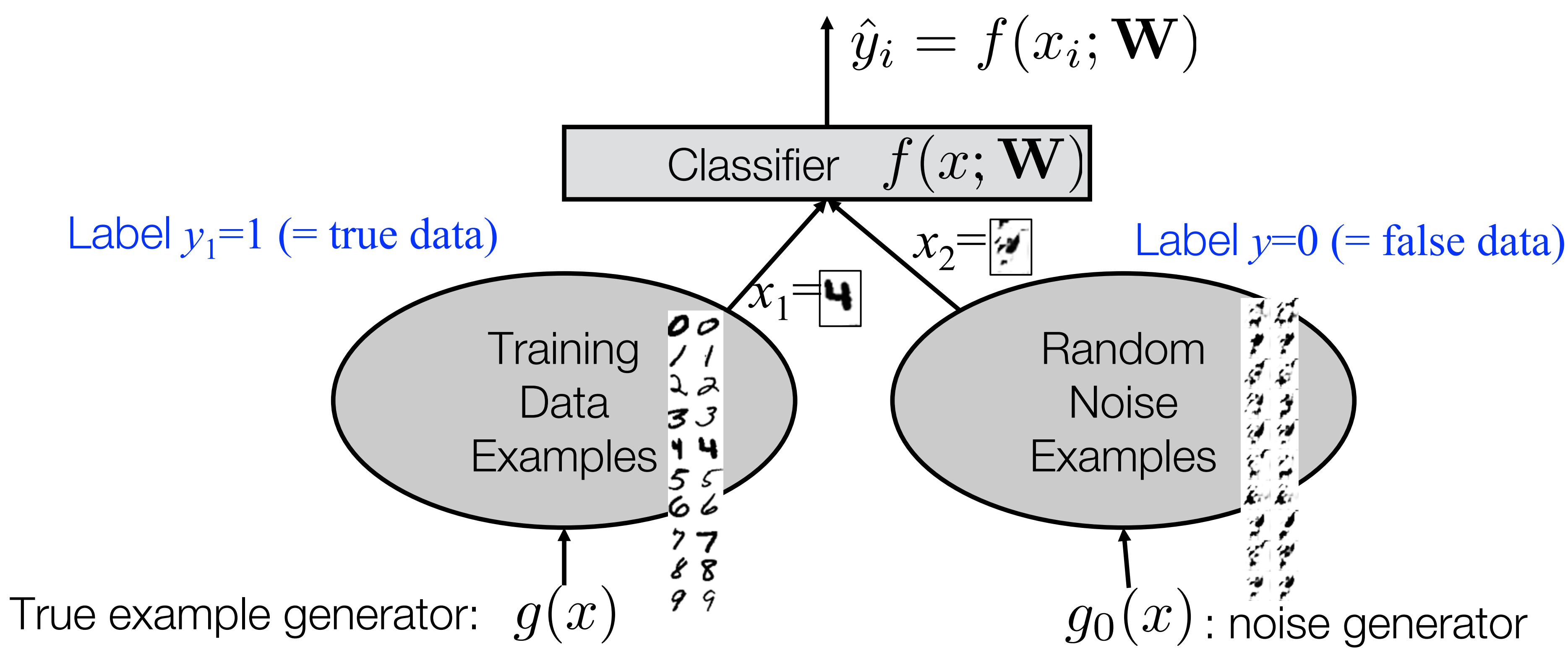
Detour:
Unsupervised as Supervised Learning
(Generation of Examples through a Classification Task)

Generative Tasks x Classification Tasks

- What is the trick?
- Machine learning methods are much better at classifying examples than generating new ones
    - In classification tasks, we use the exact derivatives to find a solution that maximizes the likelihood
    - In generative tasks, like RBMs, we can only compute an estimate of the derivative
- Because we have better techniques to classify data than to generate data…
    - Can we make generative tasks look more like classification tasks?

Learn a Classifier to Distinguishing Noise from Data

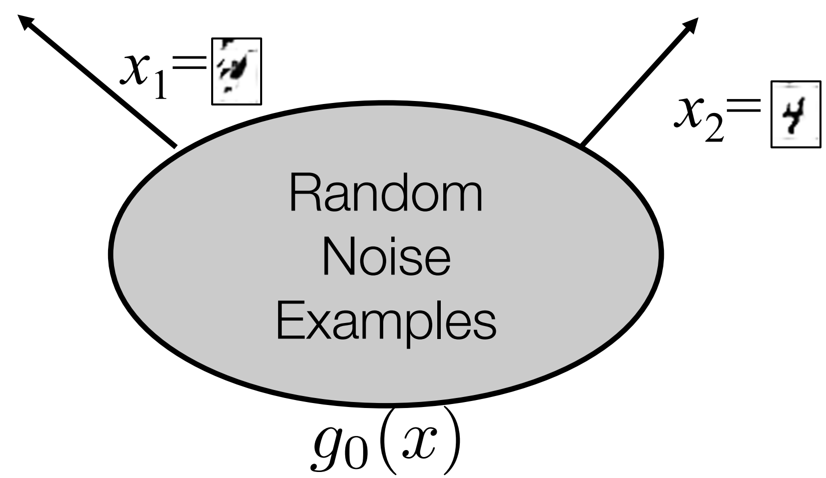
- This is key idea behind *noise contrastive estimation* (*NCE*): make generative tasks look like classification tasks
  - Pioneered by Hastie, Tibshirani, Friedman in The Elements of Statistical Learning in 2008, Section 14.2.4 “Unsupervised Learning as Supervised Learning”
  - The idea is quite simple, consider the task of learning to distinguish noise from the data, i.e., search for a good classifier $f(x;\mathbf{W})$



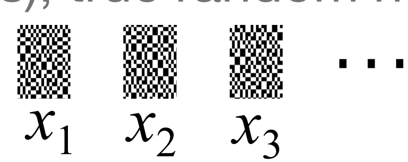
Generation Task Using Classifier $f(x; \mathbf{W})$

- Now use the learned classifier (which distinguishes noise from the data) to generate new data… but how?
  - Naïve approach: Generate examples from the random noise… whatever gets classified as "real data" will be our generated examples

$$\hat{y}_1 = f(x_1; \mathbf{W}) \approx 0 \qquad \hat{y}_2 = f(x_2; \mathbf{W}) \approx 1$$



  - What is the problem with this naïve approach?
    - In very high dimensions (e.g., images), true random noise will not generate any interesting examples



We need the noise generator to give "data-looking" examples with high probability

How?

Solution: Find a noise generator $g_0(x)$ whose examples very likely looks like the training data (can deceive our classifier with high probability)

Chicken and egg problem…

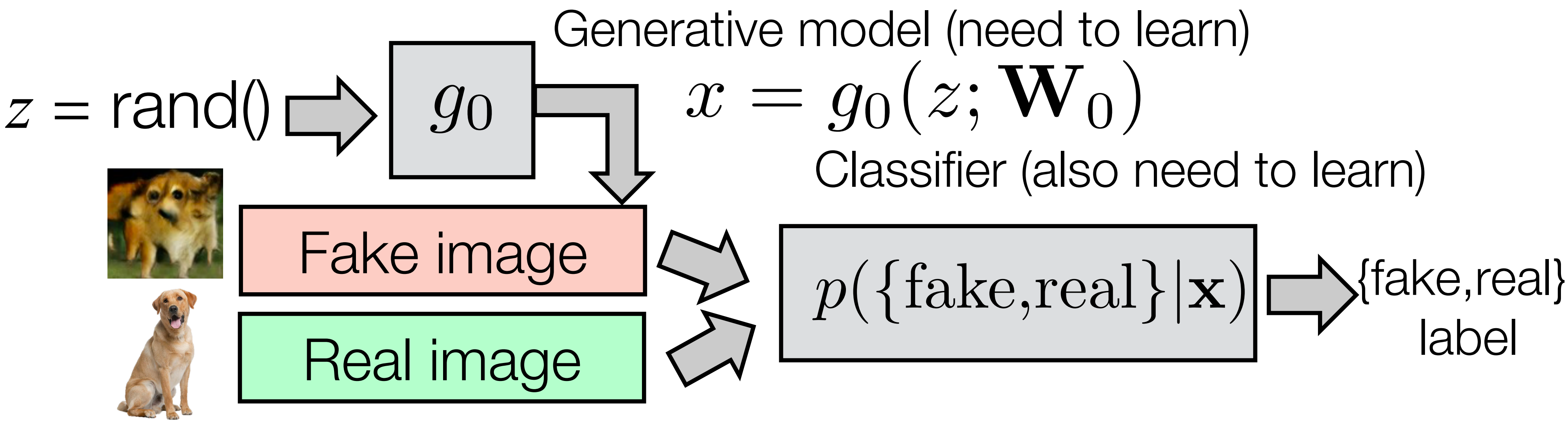
a very good $g_0(x)$ is the generator already!

Generative Adversarial Networks (GANs)

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. *Generative adversarial nets*. In NIPS 2014

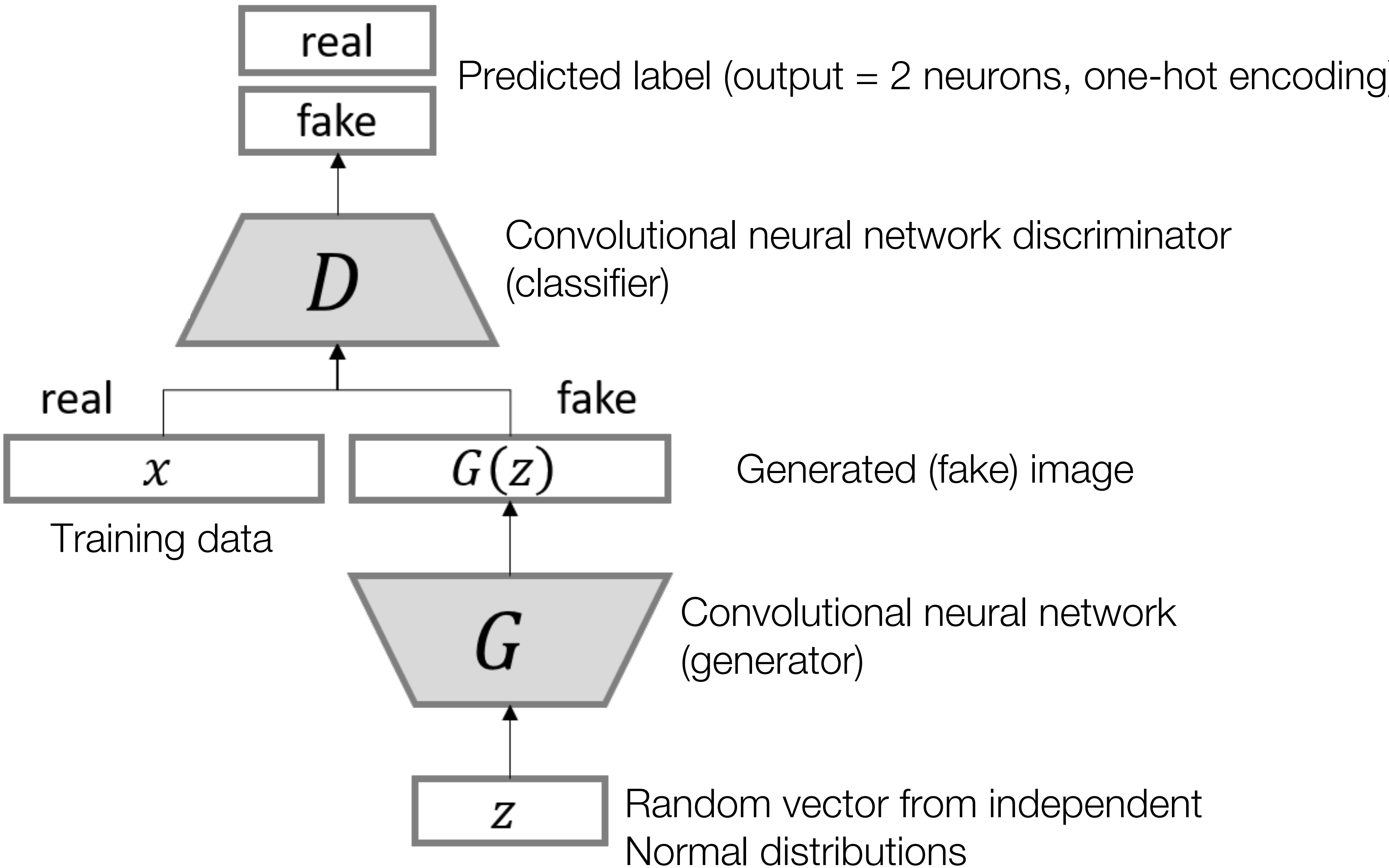
Examples of Generative Neural Network Models

- Generative Adversarial Network architecture



- The main idea is to learn the generator and classifier together
  - Goal of the generator is to fool the classifier
  - Goal of the classifier is to identify the fakes of the generator

Architecture of a GAN



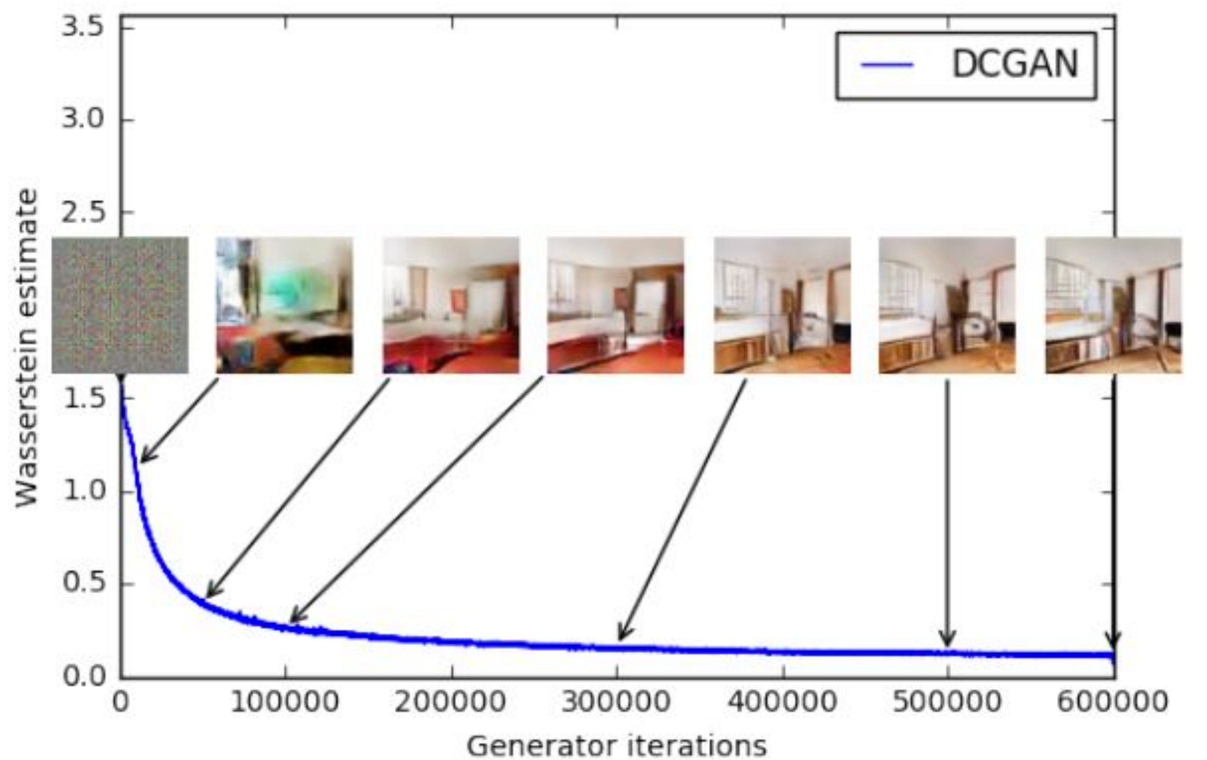
Search for a good GAN

- What is the model space?
  - Generators with parameter $\mathbf{W_0}$ and classifiers with parameters $\mathbf{W}$
- What is the search space?
  - Given a classifier $f(x ;\mathbf{W})$, we can search for $\mathbf{W_0}$ of a generator $g_0(z; \mathbf{W}_0)$, that maximizes the misclassification error of the classifier using gradient ascent
  - Given a generator $g_0(z; \mathbf{W}_0)$, the classifier will search for a $\mathbf{W}$ that minimizes the misclassification error using gradient descent
  - Alternate between maximizing and minimizing the classification error

Example of a GAN in action (DCGAN)

Bedroom generation example

- Wasserstein distance is a score function where lower is better
- The search algorithm uses an estimate of the Wasserstein distance as score function.
- Wasserstein distance avoids some of the issues of the likelihood (as we will see later)



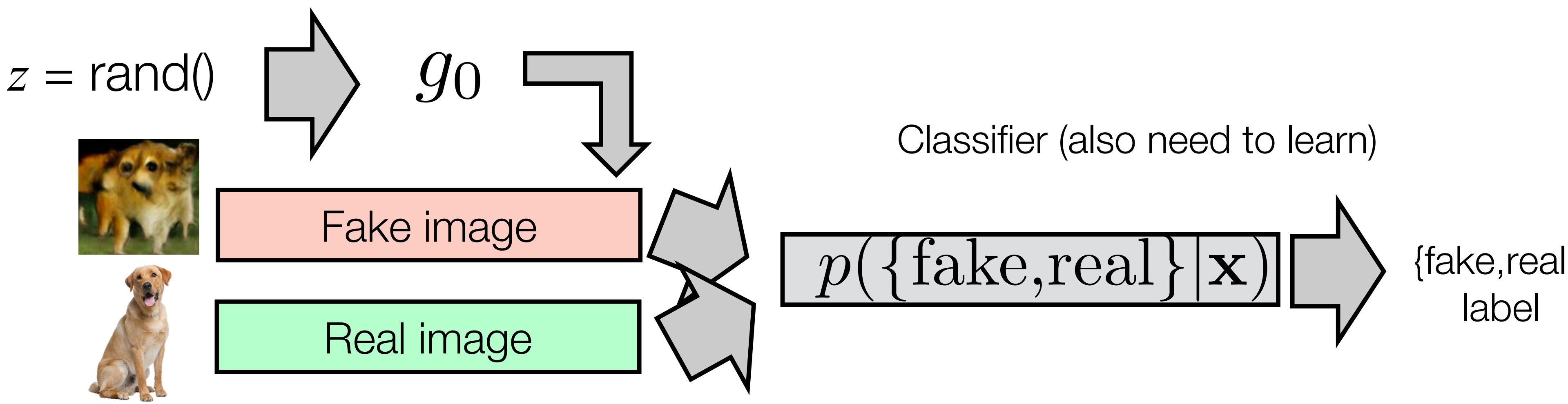
"Number of gradient steps"

- Radford, Alec, Luke Metz, and Soumith Chintala. "Unsupervised representation learning with deep convolutional generative adversarial networks." *ICLR 2016.*
  the above example uses the Wasserstein score function
  Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein GAN, ICML 2017

Issues with GANs

- **Mode collapse**: using just classification error, the generator may specialize on one type of example

e.g., learn to counterfeit just one type of dog



- The Wasserstein distance is a score function that tends to avoid mode collapsing (not covered in this class)