Lecture Notes for

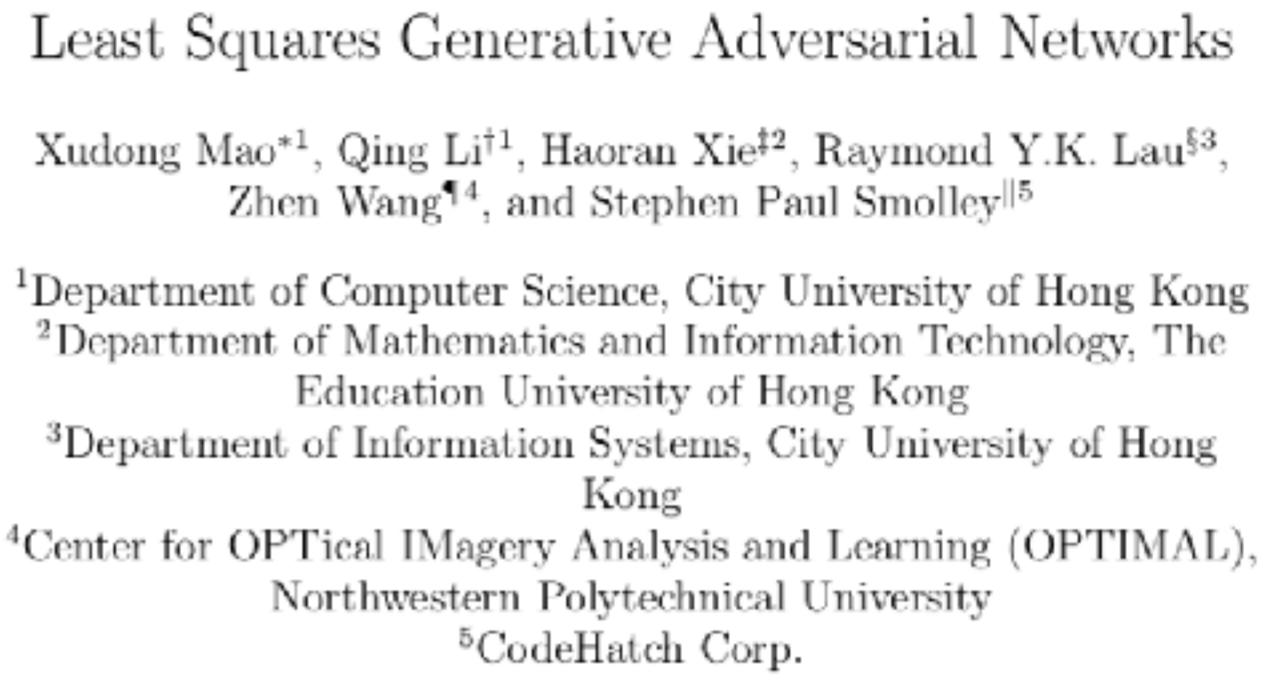
Neural Networks and Machine Learning

Practical GANs and LSGAN

Logistics and Agenda

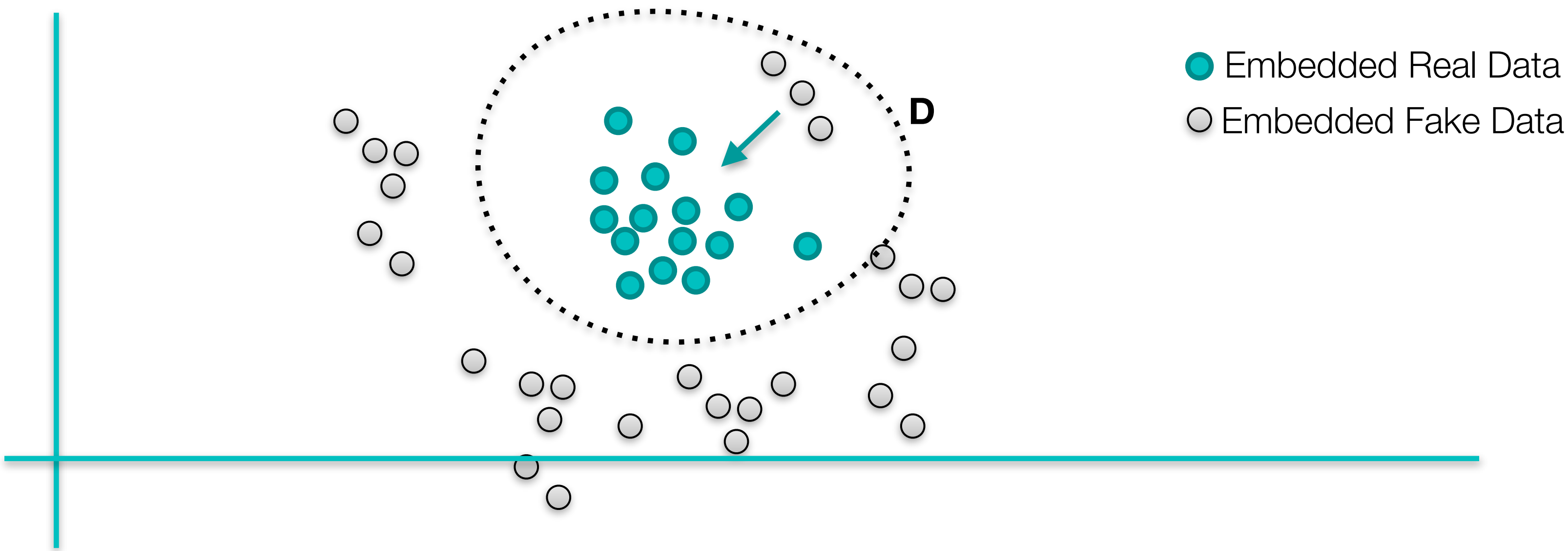
- Logistics
  - Student Paper: None
- Agenda
  - LSGAN
  - Practical GANs
  - Wasserstein GAN (next time)
  - WGAN-GP (next time)
  - BigGAN (next next time)
  - More GAN Examples (Holy GAN-zooks)

LSGAN


Least Squares Generative Adversarial Networks
Xudong Mao*1, Qing Li†1, Haoran Xie‡2, Raymond Y.K. Lau§3,
Zhen Wang¶4, and Stephen Paul Smolley‖5
1Department of Computer Science, City University of Hong Kong
2Department of Mathematics and Information Technology, The Education University of Hong Kong
3Department of Information Systems, City University of Hong Kong
4Center for OPTical IMagery Analysis and Learning (OPTIMAL), Northwestern Polytechnical University
5CodeHatch Corp.

The Least Squares GAN

- **Observation**: Generated points may (by chance) be classified as real by Discriminator—but they are still not representative of the real data
- **Solution**: Incentivize even correctly classified labels to move toward real data distribution



Mao, Xudong, Qing Li, Haoran Xie, Raymond YK Lau, Zhen Wang, and Stephen Paul Smolley. "Least squares generative adversarial networks." In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2794-2802. 2017.

Incentivizing with Least Squares

- Assume a=fake label, b=real label, c=misleading label
- The new loss function is:

$$\min_D V_{\text{LSGAN}}(D) = \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x}) - b)^2\right] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) - a)^2\right]$$

$$\min_G V_{\text{LSGAN}}(G) = \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) - c)^2\right],$$

- Here we can take advantage of the labels, even when they are classified correct/incorrect
  - …because we have a distance to margin
  - …that's it!

But that results is not publishable!

- We need to find a way to address issues that the research community is interested in (even is we don't address them)
  - Yay for academia!
- **Discussion**: is this wrong for the authors to do?

In the original GAN paper [7], the authors has shown that minimizing Equation 1 yields minimizing the Jensen-Shannon divergence:

$$C(G) = KL\left(p_{\text{data}} \middle\| \frac{p_{\text{data}} + p_g}{2}\right) + KL\left(p_g \middle\| \frac{p_{\text{data}} + p_g}{2}\right) - \log(4). \quad (3)$$

Here we also explore the relation between LSGANs and f-divergence. Consider the following extension of Equation 2:

$$\begin{aligned} \min_D V_{\text{LSGAN}}(D) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x}) - b)^2\right] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) - a)^2\right] \\ \min_G V_{\text{LSGAN}}(G) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x}) - c)^2\right] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) - c)^2\right]. \end{aligned} \quad (4)$$

Note that adding the term $\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[(D(\boldsymbol{x})-c)^2]$ to $V_{\text{LSGAN}}(G)$ does not change the optimal values since this term does not contain parameters of G.

We first derive the optimal discriminator D for a fixed G as below :

$$D^*(\boldsymbol{x}) = \frac{b p_{\text{data}}(\boldsymbol{x}) + a p_g(\boldsymbol{x})}{p_{\text{data}}(\boldsymbol{x}) + p_g(\boldsymbol{x})}. \quad (5)$$

3.2.3 Parameters Selection

One method to determine the values of a, b, and c in Equation 2 is to satisfy the conditions of $b - c = 1$ and $b - a = 2$, such that minimizing Equation 2 yields minimizing the Pearson χ^2 divergence between $p_d + p_g$ and $2p_g$. For example, by setting $a = -1$, $b = 1$, and $c = 0$, we get the following objective functions:

$$\begin{aligned} \min_G V_{\text{LSGAN}}(D) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x}) - 1)^2\right] - \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) + 1)^2\right] \\ \min_G V_{\text{LSGAN}}(G) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})))^2\right]. \end{aligned} \quad (8)$$

Another method is to make G generate samples as real as possible by setting $c = b$. For example, by using the 0-1 binary coding scheme, we get the following objective functions:

$$\begin{aligned} \min_D V_{\text{LSGAN}}(D) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x}) - 1)^2\right] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})))^2\right] \\ \min_G V_{\text{LSGAN}}(G) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) - 1)^2\right]. \end{aligned} \quad (9)$$

In practice, we observe that Equation 8 and Equation 9 show similar performance. Thus either one can be selected. In the following sections, we use Equation 9 to train the models.

LS-GAN Parameter Selection

3.2.3 Parameters Selection

One method to determine the values of a, b, and c in Equation 2 is to satisfy the conditions of $b - c = 1$ and $b - a = 2$, such that minimizing Equation 2 yields minimizing the Pearson χ^2 divergence between $p_d + p_g$ and $2p_g$. For example, by setting $a = -1$, $b = 1$, and $c = 0$, we get the following objective functions:

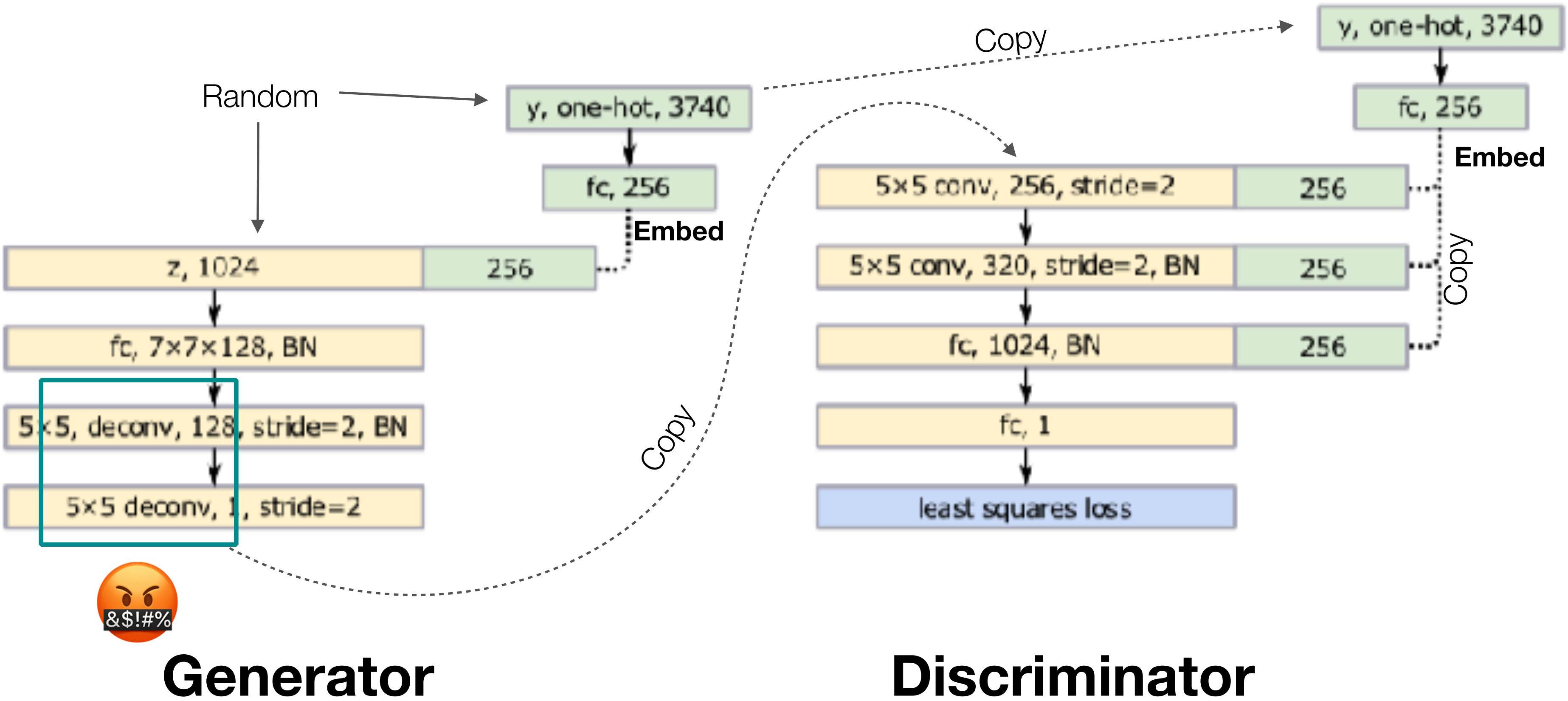
$$\begin{aligned}
\min_D V_{\text{LSGAN}}(D) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x}) - 1)^2\right] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) + 1)^2\right] \\
\min_G V_{\text{LSGAN}}(G) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})))^2\right].
\end{aligned} \tag{8}$$

Another method is to make G generate samples as real as possible by setting $c = b$. For example, by using the 0-1 binary coding scheme, we get the following objective functions:

$$\begin{aligned}
\min_D V_{\text{LSGAN}}(D) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}\left[(D(\boldsymbol{x}) - 1)^2\right] + \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})))^2\right] \\
\min_G V_{\text{LSGAN}}(G) &= \frac{1}{2}\mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}\left[(D(G(\boldsymbol{z})) - 1)^2\right].
\end{aligned} \tag{9}$$

In practice, we observe that Equation 8 and Equation 9 show similar performance. Thus either one can be selected. In the following sections, we use Equation 9 to train the models.

Architecture Employed (less mode collapse)



Main idea: by exposing the generator here, we can help avoid mode collapse.

LS-GAN Results

- Some takeaways:
  - RMSProp seems to be better than Adam
  - Reasonable values for a,b,c have similar performance
  - Mode collapse is still a problem, but not nearly as bad as regular GANs

Experiment:
Generate 2D samples from known Mix of Gaussian Distributions, then train 3 layer GANs to generate the same 2D data.

Does one learn the distribution?

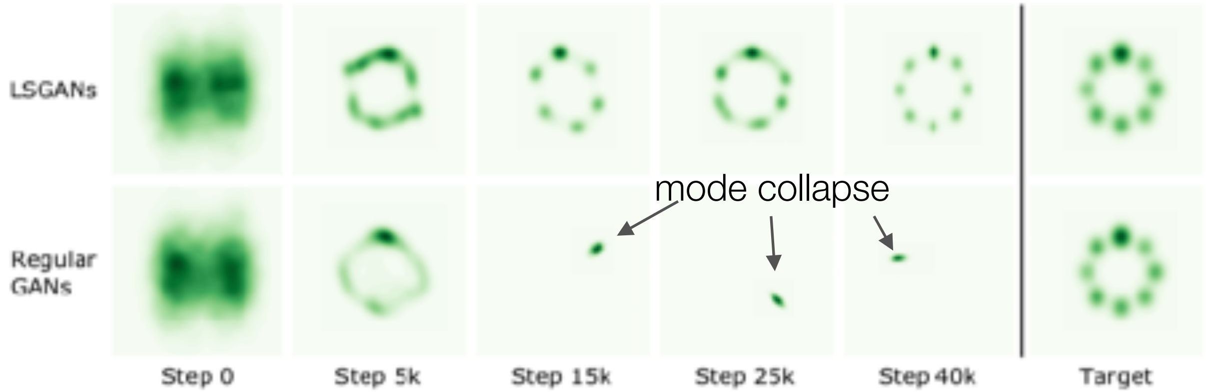
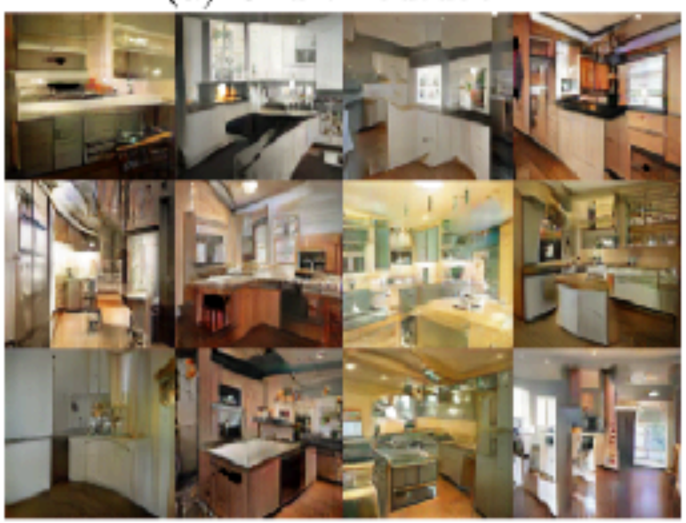


Figure 8: Dynamic results of Gaussian kernel estimation for LSGANs and regular GANs. The final column shows the real data distribution.

Qualitative Results

(a) Church outdoor.

(b) Dining room.



(c) Kitchen.

(d) Conference room.

More Qualitative Results

(a) Generated by LSGANs.

(b) Generated by DCGANs (Reported in [11]).

How can we trust the results?

- Do we trust that the authors are not cherry picking the results?
- If I perform random seed optimization and run my algorithms for longer, can I always claim its better?
  - …but maybe not for the reasons I publish…
- Without strong quantitative evaluation criteria for image generation, can there be a solution to this?
  - Require human subjects evaluation?
  - But can't they still tune the results for their algorithm before the human subjects testing?
  - What about open sourcing the weights of a tuned algorithm?

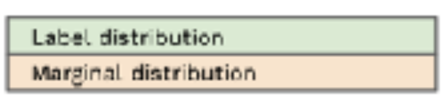
An Accepted Measure: Inception Score

$$\hat{p}(y) = \frac{1}{N}\sum_i \overbrace{p(y \mid \mathbf{x}^{(i)}_{fake})}^{\text{marginal}}$$

Expected class distribution through a trained CNN, like VGG should be **nearly uniform** in ideal case

$$IS(G) \approx \exp\left(\frac{1}{N}\sum_i D_{KL}\left(p(y \mid \mathbf{x}^{(i)}_{fake}) \,\|\, \hat{p}(y)\right)\right)$$

average KL Divergence of marginal of generated images with $\hat{p}$, should **differ dramatically**, ideally



https://medium.com/octavian-ai/a-simple-explanation-of-the-inception-score-372dff6a8c7a

LSGAN Inception Score

Discriminator Feature-based Inference by Recycling the Discriminator of GANs

Generative adversarial networks (GANs)successfully generate high quality data by learning amapping from a latent vector to the data. Various studies assert that the latent space of a GAN is semanticallymeaningful and can be utilized for advanced

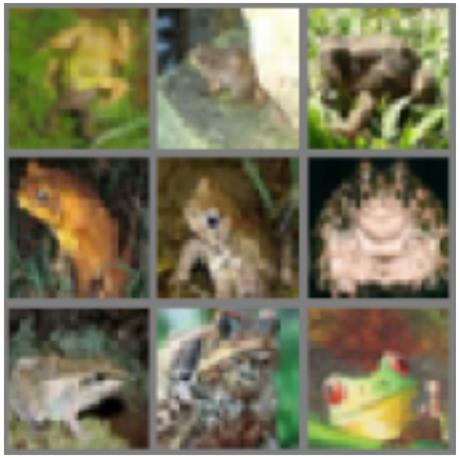
Metric	DCGAN	LSGAN
Inception score	6.50	5.98

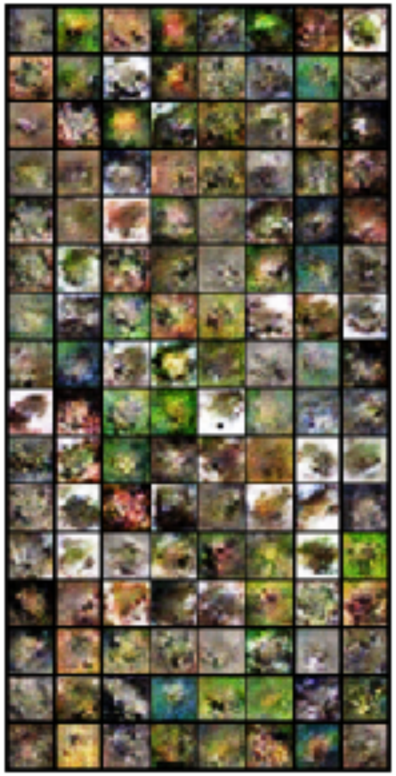
Higher is better

https://paperswithcode.com/paper/high-quality-bidirectional-generative/review/

GANs in PyTorch

Master Repository:
`07c GANsWithTorch.ipynb`





Revisiting Demo with the LS-GAN architecture
(but not the one hot encoding embeddings…)

GANs Loss

Meta-meta-learning for Neural Architecture Search through arXiv Descent

Antreas Antoniou
MetaMind
aa@mm.ai

Nick Pawlowski
Googel x^2
nick@x.x

Jack Turner
slow.ai
jack@slow.ai

James Owers
Facebrook AI Research Team
jim@fart.org

Joseph Mellor
Institute of Yellow Jumpers
joe@anditwasall.yellow

Elliot J. Crowley
ClosedAI
elliot@closed.ai

Abstract

Recent work in meta-learning has set the deep learning community alight. From minute gains on few-shot learning tasks, to discovering architectures that are slightly better than chance, to solving intelligence itself[1], meta-learning is proving a popular solution to every conceivable problem ever conceivably conceived ever.

In this paper we venture deeper into the computational insanity that is meta-learning, and potentially risk exiting the simulation of reality itself, by attempting to meta-learn at a third learning level. We showcase the resulting approach—which we call *meta-meta-learning*—for neural architecture search. Crucially, instead of *meta-learning* a neural architecture differentiably as in DARTS (Liu et al., 2018) we *meta-meta-learn* an architecture by searching through arXiv. This *arXiv descent* is GPU-free and only requires a handful of graduate students. Further, we introduce a regulariser, called *college-dropout*, which works by randomly removing a single graduate student from our system. As a consequence, procrastination levels decrease significantly, due to the increased workload and sense of responsibility each student attains.

The code for our experiments is publicly available at [redacted].
Edit: we have decided not to release our code as we are concerned that it may be used for malicious purposes.

Why are GANs so difficult to train?

- Why did we need to add noise to labels ?
- Why were sparse gradients needed?
- Does using least squares really solve the problem?
- Formalization:
  - GANs will not converge
  - GAN outputs all might be similar (**mode collapse**)
  - Slow training: gradient vanishes, sometimes not recoverable
  - Theory behind this is still developing, but a good start:

Generative Adversarial Networks (GANs): What it can generate and What it cannot?

P Manisha
manisha.padala@research.iiit.ac.in

Sujit Gujar
sujit.gujar@iiit.ac.in

TOWARDS PRINCIPLED METHODS FOR TRAINING GENERATIVE ADVERSARIAL NETWORKS

Martin Arjovsky
Courant Institute of Mathematical Sciences
martinarjovsky@gmail.com

Léon Bottou
Facebook AI Research
leonb@fb.com

The Optimal Discriminator

- Discriminator is maximizing:

$$\max_{d} \mathbf{E}_{x \leftarrow p_{data}} \left[\log d(x)\right] + \mathbf{E}_{x \leftarrow g(z)} \left[\log(1 - d(x))\right]$$

$$\max_{d} \int_{x \in [P_{data}, P_g]} \left(\boxed{p_{data}(x) \cdot \log d(x) + p_g(x) \cdot \log(1 - d(x))} \right) dx$$

$$\nabla_f = 0 = \frac{1}{\partial d(x)} \left(p_{data}(x) \cdot \log d(x) + p_g(x) \cdot \log(1 - d(x)) \right)$$

… math …

$$d(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)}$$

-**Conclusion**: Discriminator is optimal when this condition occurs

Optimality of Generator

- We can similarly optimize the generator to get:

$$C(G) = -\log(4) + KL\left(p_{data} \parallel \frac{p_{data} + p_g}{2}\right) + KL\left(p_g \parallel \frac{p_{data} + p_g}{2}\right)$$

$$C(G) = -\log(4) + 2.JSD(p_{data} \parallel p_g)$$ Jenson-Shannon Divergence

- **Vanishing gradients**: if $p_{data}||p_g \approx 0$ then JSD saturates and the gradient is basically zero. Optimization has no idea what direction to move in…
- What if we use some tricks to keep the optimization moving even when we do not know which direction to go?