Adversarial Neural Architecture Search

Amirmohammad Naeini

Agenda

Introduction

Gan Networks

Architecture Search

Method

Experiments

Conclusion

References

Introduction

- Image generation
- Trial and Error Architecture
  - DCGAN-based and ResNet-based.
- The benefits of specifically designing architecture
  - ResNet, DenseNet, MobileNet, ShuffleNet, EfficientNet, HRNet.
- Neural Architecture Search (NAS)
  - Gan-based tasks
- Hardware Limit vs Search Space
- Transferability and scalability.

Generative Networks

Examples

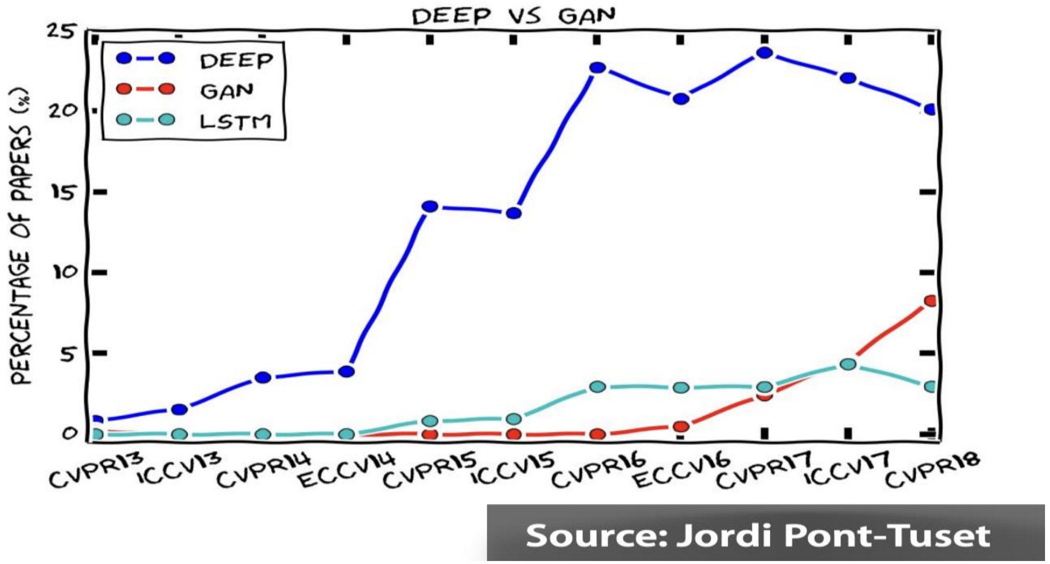
- Boltzmann Machines
- Variational Auto Encoders
- Diffusion Models
- **Generative Adversarial Networks**

“The most interesting idea in the last ten years in machine learning.”

Yann LeCun (Facebook AI Research)

GAN

- Goodfellow 2014
- Two-Player Game
- Loss function
  - Min-Max
- Hard to train



$$\min_G \max_D V(D,G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))].$$

Sample GAN Training

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

---

for number of training iterations **do**

  for k steps **do**

  • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.

  • Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.

  • Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for

  • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.

  • Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for

The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Neural Architecture Search

- AutoML
- Search for an Effective Architecture
- Expensive Costs
  - Search Space
  - Differentiable
    - Gradient Descent

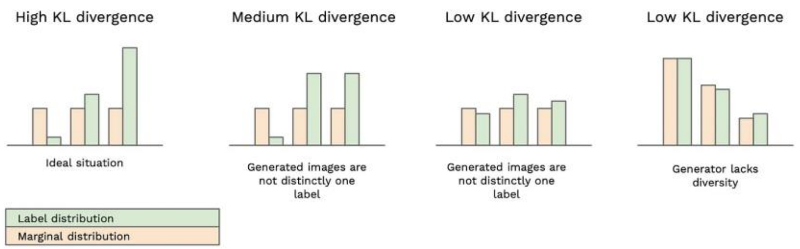
Computational Cost

- Search ImageNette
  - 15 hours on Colab Pro (40Gb A100)
- Search Cifar10
  - 15 hours on Colab Pro (40GB A100)
- Search Stl10
  - 15 hours on Colab Pro (40GB A100)
- Train Cifar10
  - 8 hours Colab Pro (40 GB A100)

Loss plot for Pandapower Implementation

Inception Score

- Automatically grade the quality of images
  - images have variety
  - each image distinctly looks like something
  - Kullback-Leibler (KL) divergence
  - Higher is Better



Inception Score Cases

$$\mathrm{IS} = \exp\left(\mathbb{E}_{\mathbf{x}\sim p_g}\left[\mathrm{KL}\left(p(\mathbf{y}\mid\mathbf{x}), |, p(\mathbf{y})\right)\right]\right)$$

Inception Score

FID Score

- Remove the Output Layer
- Pre-Trained Model (Inception V3)
- Mean and Covariance

$$FID(x,y) = ||\mu_x - \mu_y||^2 + Tr(C_x + C_y - 2(\sqrt{C_x * C_y}))$$

FID Score

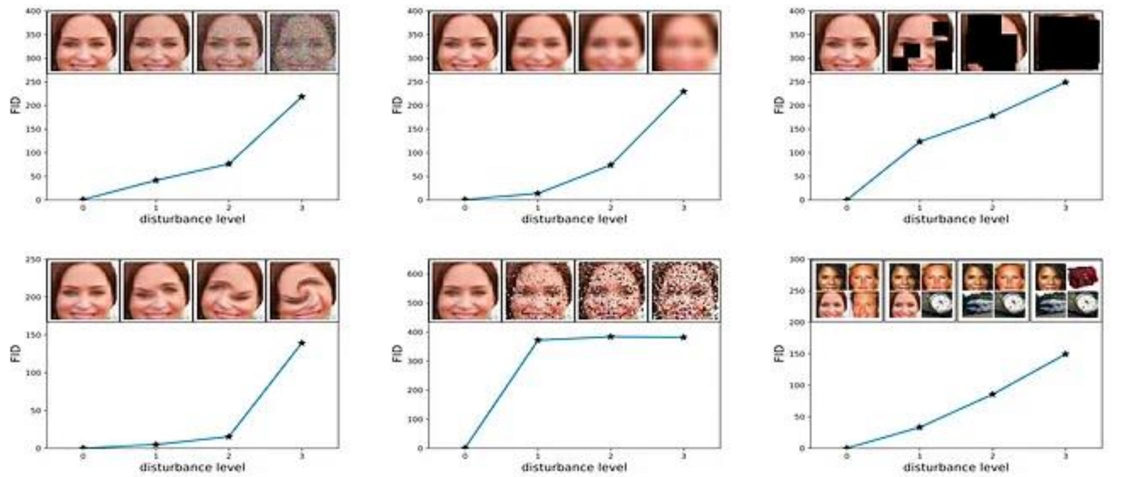


Figure 3: FID is evaluated for **upper left:** Gaussian noise, **upper middle:** Gaussian blur, **upper right:** implanted black rectangles, **lower left:** swirled images, **lower middle:** salt and pepper noise, and **lower right:** CelebA dataset contaminated by ImageNet images. The disturbance level rises from zero and increases to the highest level. The FID captures the disturbance level very well by monotonically increasing.

FID Score examples

Search Space

- Normal Operations
- Up-sample Operations
- Down-sample Operations

- None
- Convolution 1x1, Dilation=1
- Convolution 3x3, Dilation=1
- Convolution 5x5, Dilation=1
- Identity
- Convolution 3x3, Dilation=2
- Convolution 5x5, Dilation=2

- Transposed Convolution 3x3
- Nearest Neighbor Interpolation
- Bilinear Interpolation

- Average Pooling
- Convolution 3x3, Dilation=1
- Convolution 5x5, Dilation=1
- Max Pooling
- Convolution 3x3, Dilation=2
- Convolution 5x5, Dilation=2

Operations in Graph

Method

- 3 Up-Cell Generator
  - alpha
- 4 Down-cell Discriminator
  - Bheta
- Alpha and Bheta to genotypes

$$\min_{\alpha} \max_{\beta} V(\alpha, \beta) = \mathbb{E}_{x \sim p_{data}(x)}[\log D(x \mid \beta, W_D^*(\beta))]$$
$$+ \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G(z \mid \alpha, W_G^*(\alpha)) \mid \beta, W_D^*(\beta)))]$$

$s.t.$

$$W_D^*(\beta) = \underset{W_D(\beta)}{\arg\max}\, \mathbb{E}_{x \sim p_{data}(x)}[\log D(x \mid \beta, W_D(\beta))]$$
$$+ \mathbb{E}_{z \sim p_z(z)}[\log(1 - D(G_{D_\beta}^*(z) \mid \beta, W_D(\beta)))]$$
$$W_G^*(\alpha) = \underset{W_G(\alpha)}{\arg\min}\, \mathbb{E}_{z \sim p_z(z)}[\log(1 - D_{G_\alpha}^*(G(\alpha \mid W_G(\alpha)))],$$

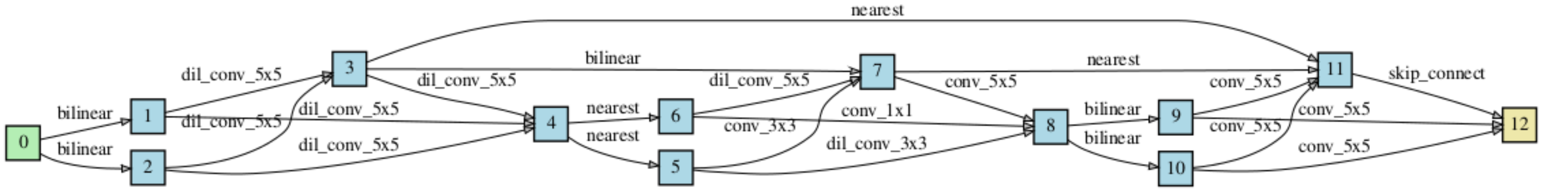
Objective Function

NAS GAN Training

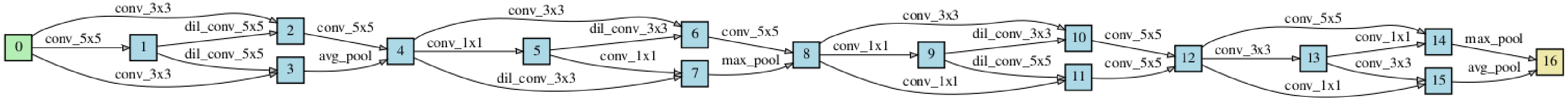
Algorithm 1 Minibatch stochastic gradient descent training of Adversarial Neural Architecture Search.

for number of training iterations **do**
  for k step **do**
    Sample minibatch of $2m$ noise samples $\left\{z^{(1)}, ..., z^{(2m)}\right\}$ from noise prior.
    Sample minibatch of $2m$ examples $\left\{x^{(1)}, ..., x^{(2m)}\right\}$ from real data distribution.
    Update the architecture of discriminator by ascending its stochastic gradient:
    $\nabla_{\beta} \frac{1}{m} \sum_{i=1}^{m} \left[\log(x^{i}) + \log(1 - D(G(z^{i})))\right]$
    Update the weights of discriminator by ascending its stochastic gradient:
    $\nabla_{W_D} \frac{1}{m} \sum_{i=m+1}^{2m} \left[\log(x^{i}) + \log(1 - D(G(z^{i})))\right]$
  end for
  Sample minibatch of $2m$ noise samples $\left\{z^{(1)}, ..., z^{(2m)}\right\}$ from noise prior.
  Update the architecture of generator by descending its stochastic gradient:
  $\nabla_{\alpha} \frac{1}{m} \sum_{i=1}^{m} \left[\log(1 - D(G(z^{i})))\right]$
  Update the weights of generator by descending its stochastic gradient:
  $\nabla_{W_G} \frac{1}{m} \sum_{i=m+1}^{2m} \left[\log(1 - D(G(z^{i})))\right]$
end for

Searching Results

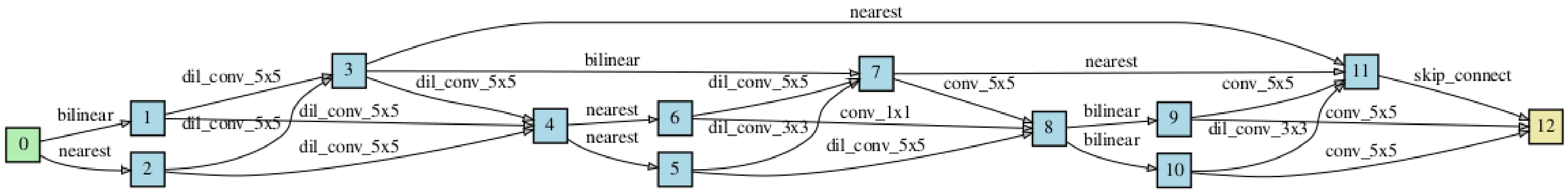


Generator CIFAR10

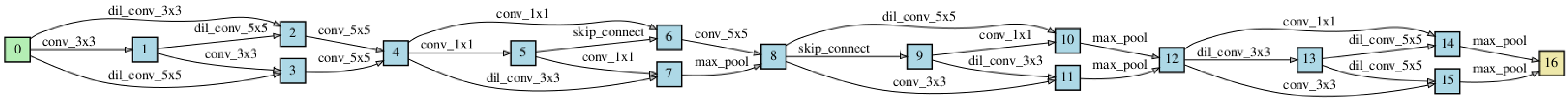


Discriminator Cifar10

Searching Results

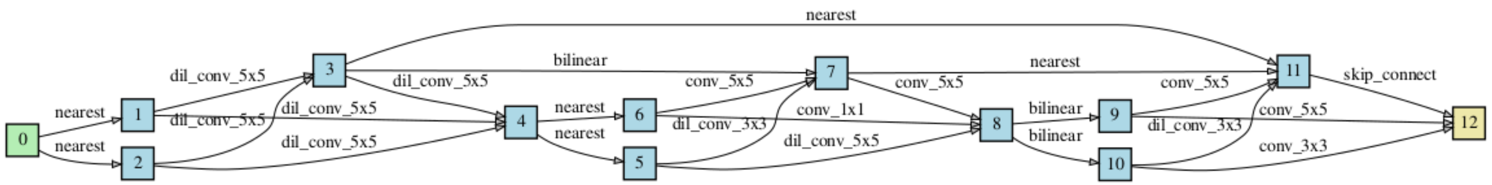


Generator STL10

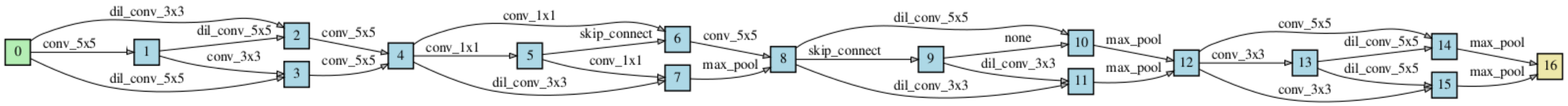


Discriminator STL10

Searching Results

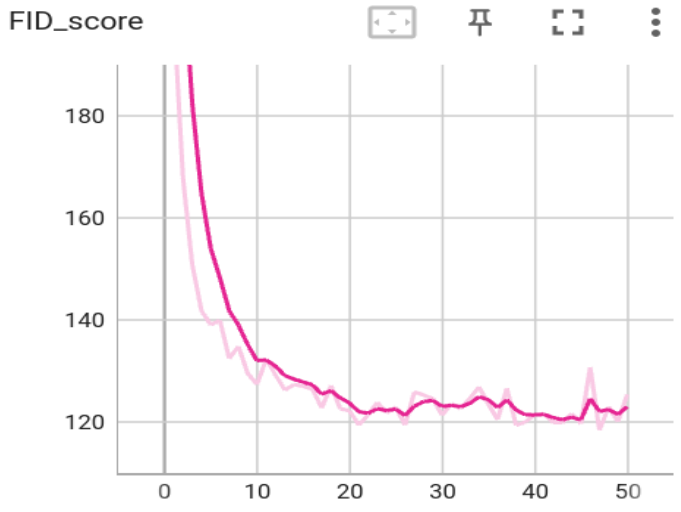


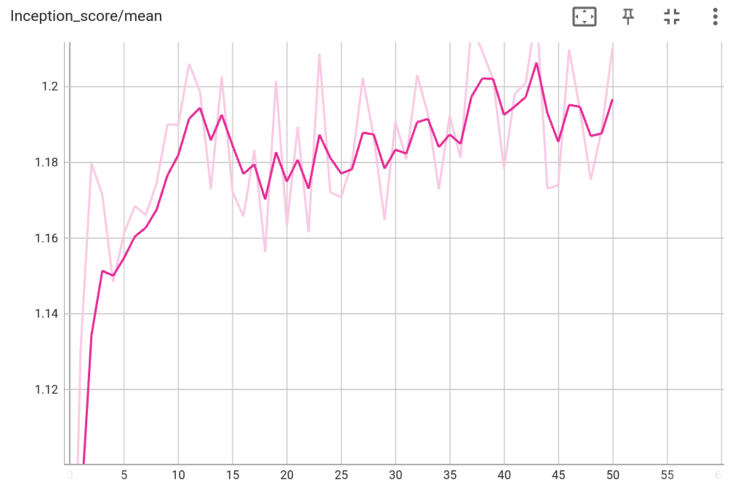
Generator Imagenette



Discriminator Imagenette

Training Results





Training Results

Conclusion

- Different Architecture
- Good Results
    - Not state of the art IS:1.25 , FID 105.32
    - Different Eval Batches
- 500 Compute Units Google
- Training

References

- Gao, C., Chen, Y., Liu, S., Tan, Z., & Yan, S. (2020). Adversarialnas: Adversarial neural architecture search for gans. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 5680-5689).
- Xinyu Gong, Shiyu Chang, Yifan Jiang, and Zhangyang Wang. Autogan: Neural architecture search for generative adversarial networks. In ICCV, 2019.
- Salimans, T., Goodfellow, I., Zaremba, W., Cheung, V., Radford, A., & Chen, X. (2016). Improved techniques for training gans. Advances in neural information processing systems, 29.
- Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., & Hochreiter, S. (2017). Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems, 30.

Thank you