Summary of Paper: Energy-Based Generative Adversarial Network

Presenter: Joseph Tobin

Department of Computer Science, University of Virginia

https://qdata.github.io/deep2Read/

Energy-Based Generative Adversarial Network (2016)

- Authors: Junbo Zhao, Michael Mathieu, Yann LeCun

We introduce the "Energy-based Generative Adversarial Network" model (EBGAN) which views the discriminator as an energy function that associates low energies with the regions near the data manifold and higher energies with other regions. Similar to the probabilistic GANs, a generator is trained to produce contrastive samples with minimal energies, while the discriminator is trained to assign high energies to these generated samples. Viewing the discriminator as an energy function allows to use a wide variety of architectures and loss functionals in addition to the usual binary classifier with logistic output. Among them, an instantiation of EBGAN is to use an auto-encoder architecture, with the energy being the reconstruction error. We show that this form of EBGAN exhibits more stable behavior than regular GANs during training. We also show that a single-scale architecture can be trained to generate high-resolution images.

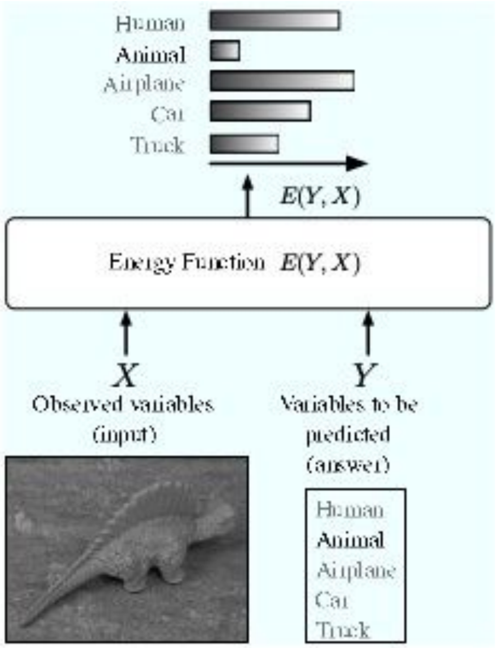
Generative Adversarial Networks

- Simultaneously train a discriminator and a generator
- Discriminator is trained to distinguish between generated samples and real samples
- Generator is trained to produce samples that cannot be distinguished from real samples
    - Generator uses gradient of the output of the discriminator with respect to its input in training

Energy Based Models (EBMs)

- EBMs capture dependencies between variables by associating a scalar energy to each configuration of variables
- Learning: Find energy function that associates low energies with correct values, high with incorrect
    - Loss functional (represents quality of energy func.) is minimized
- Inference: consider every matching between X and each label in Y and take the minimum energy (according to learned energy function) $Y^* = \mathrm{argmin}_{Y \in \mathcal{Y}} E(Y, X).$
    - In more complicated cases, have to determine "inference procedure"

Energy Based Models



- Used for:
    - Prediction, classification, decision making
    - Ranking
    - Detection
    - Conditional Density estimation

Auto-encoders

- Neural network that is trained to approximately copy input to output
- Encode data in low-dimensional space
- Decode encoded data back to feature space
- Originally deterministic, more recently probabilistic
- Traditionally used for feature learning and dimensionality reductions

EBGAN Auto-Encoder Model

- z random vector, G generator, E is energy estimate

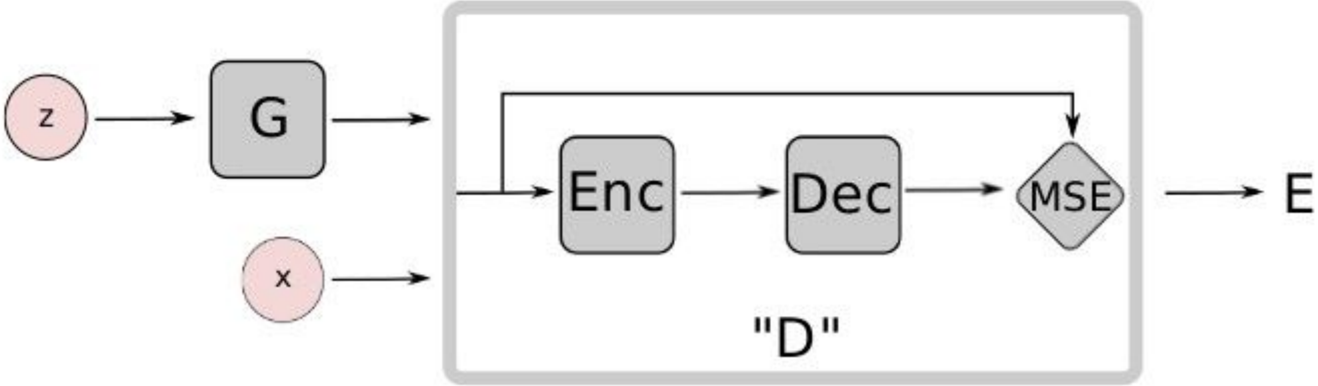


Figure 1: EBGAN architecture.

EBGAN Auto-Encoder Model Continued

- Discriminator loss function:
  - m = positive margin loss
  - Gives low energy to data samples, high energy to generated samples

$$
\begin{aligned}
f_D(x, z) &= D(x) + [m - D(G(z))]^+ \\
&= \|Dec(Enc(x)) - x\| + [m - \|Dec\Big(Enc(G(z))\Big) - G(z)\|]^+,
\end{aligned}
$$

- Generator loss function
  - Standard for adversarial, minimize second term of discriminator loss

$$
\begin{aligned}
f_G(z) &= \|D(G(z))\| \\
&= \|Dec\Big(Enc(G(z))\Big) - G(z)\|
\end{aligned}
$$

EBGAN Auto-Encoder Model Continued

- We cannot used gradient based optimization because generator loss is no longer negative because of discriminator loss
- But every Generator minima is a Discriminator maxima, and therefore we can find a fixed point in optimization
- Use adam optimization method (Kingma & Ba, 2014)
  - First order gradient descent method

Repelling Regularizer (for Generator)

- Want Gen. to generate diverse data samples
- Minibatch Distribution inputs multiple batches of samples for more effective training
  - But MBD is difficult to use with auto-encoder frameworks
- Propose repelling regularizer imposed on encoder representation (bs = batch size)
  - Makes samples as orthogonal as possible to make gradient avoid producing similar samples during back propagation

$$f_{PT}(S) = \frac{1}{bs(bs-1)} \sum_{i} \sum_{j \neq i} \left(\frac{S_i^{\top} S_j}{\|S_i\| \|S_j\|} \right)^2,$$

Training Stability

- Want learning model to be stable: prediction does not change when the training data is modified slightly
- Bridge from regular GAN to EBGAN by applying Gibbs dist.

$$P(Y|X) = \frac{e^{-\beta E(Y,X)}}{\sum_{Y \in \mathcal{Y}} e^{-\beta E(Y,X)}},$$

- E(Y,X) = corresponding energy value of (Y,X), Beta a hyperparameter
- E(Y*,x) -> - infinity iff P(Y*|X) = 1 in negative log-likelihood loss function
  - Where Y* is the correct categorization of X

Concerns for Training Auto-Encoders and L2 Loss

- Auto-encoders learn little more than an identity function
  - Does not have any effect because it must learn to reconstruct real samples while also separating reconstructions from generated samples
  - Auto-encoder exclusively serves as specific form of energy function
- L2 loss is suboptimal because certain tasks have regression-to-mean problem
  - Obtaining sharp reconstructions is not the goal of EBGANs
- Many other energy formulations are applicable in place of D and other auto-encoders can be considered among other potentials

GAN vs EBGANS (MNIST)

- Trained on MNIST, reduced grid search workhouse

- number of layers in G, nLayerG
- number of layers in D, nLayerD
- number of neurons in G, sizeG
- number of neurons in D, sizeD

Table 1: Grid search specs

Settings	EBGANs	GANs
nLayerG	[2, 3, 4, 5]	[2, 3, 4, 5]
nLayerD	[2, 3, 4, 5]	[2, 3, 4, 5]
sizeG	[400, 800, 1600, 3200]	[400, 800, 1600, 3200]
sizeD	[128, 256, 512, 1024]	[128, 256, 512, 1024]
dropoutD	[true, false]	[true, false]
optimD	adam	[adam, sgd]
optimG	adam	[adam, sgd]
lr	0.001	[0.01, 0.001, 0.0001]
Total number of experiments:	**512**	**6144**

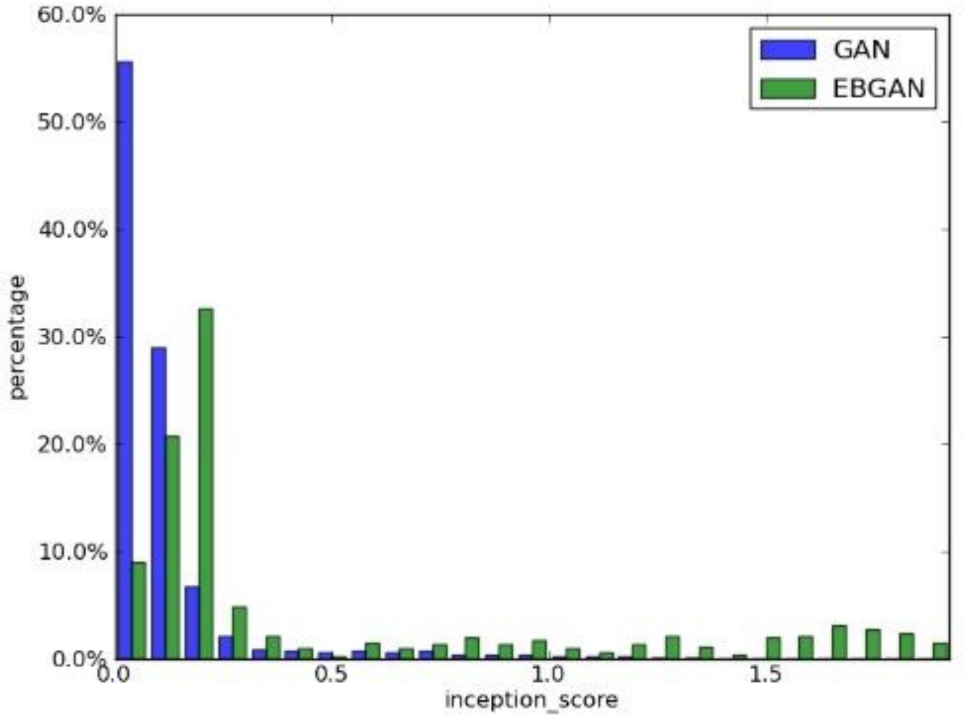
GAN vs EBGANS (MNIST)

- Batch normalization, ReLU default
- Weights initialized from N(0, 0.002), N(0,0.02) in D, G respectively
- Use inception score proposed by (Salimans et al., 2016)
  - Numerical assessment approach

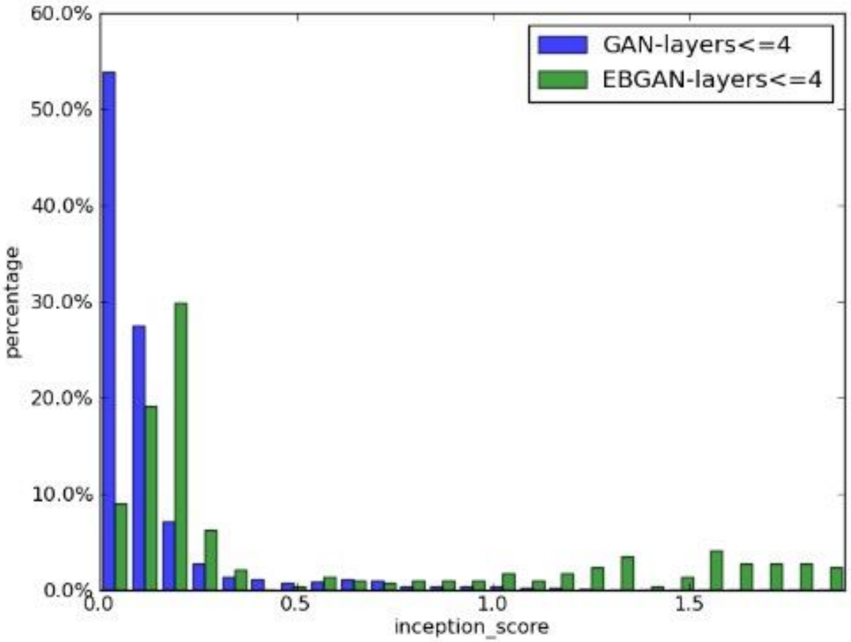
$$I = E_x KL(p(y|\mathbf{x}) \| p(y)),$$

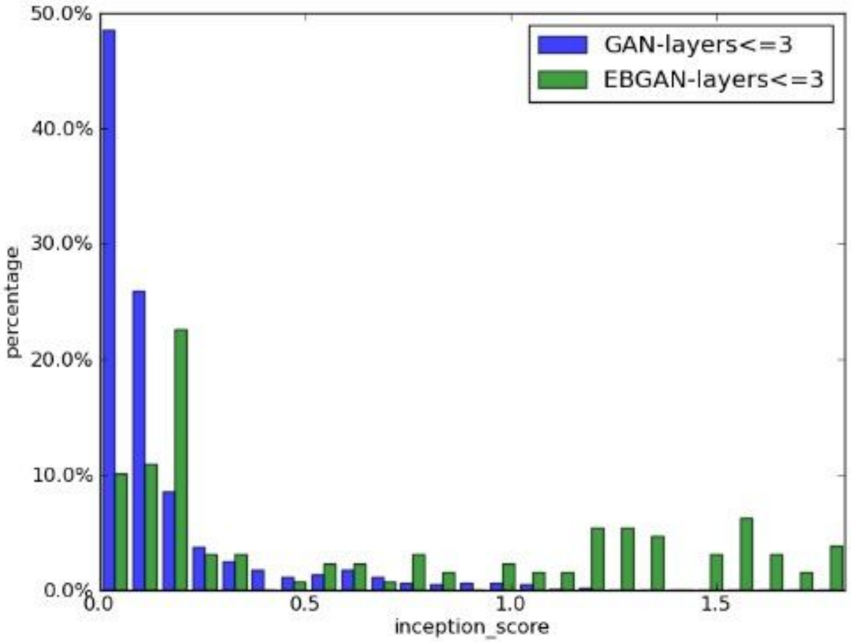
- Histograms depict performance according to number of layers
- Paper also includes performance according to different optimizations according to previous params

GAN vs EBGANS (MNIST)



GAN vs EBGANS (MNIST)



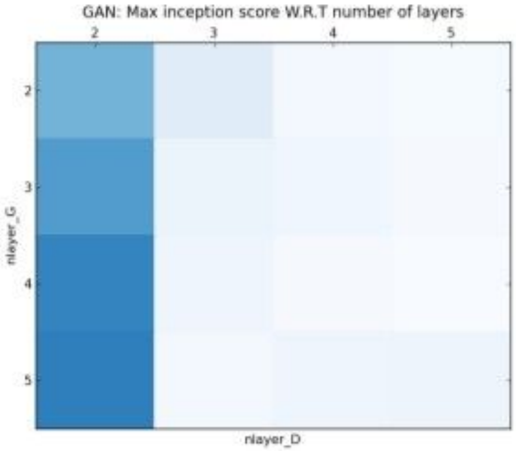


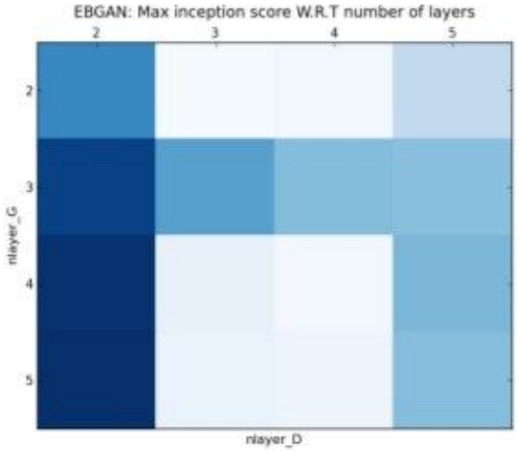
GAN vs EBGANS (MNIST)

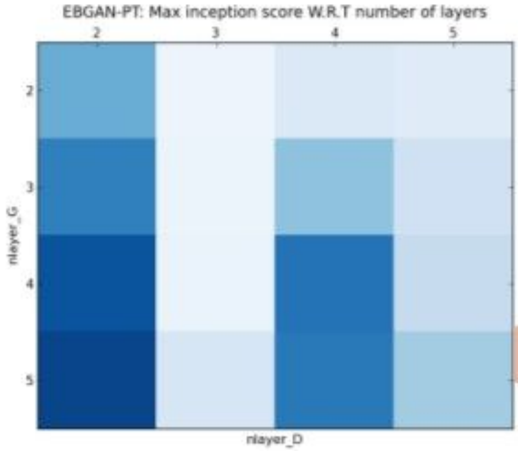
- EBGANs consistently perform more reliably and satisfactorily than GANs
  - GANs may be comparable to EBGANs if we use more intensely tuned hyper-parameters and meta-parameters
- Used heatmaps to show best and median performance among the models and also used EBGAN-PT (pullaway term)
- Based on grid search and qualitative results, EBGANs and EBGAN-PTs produce more visually appealing digits and performed more reliably

GAN vs EBGANS (MNIST)

Max

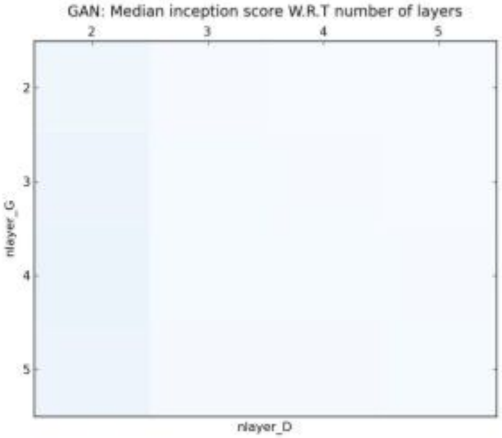


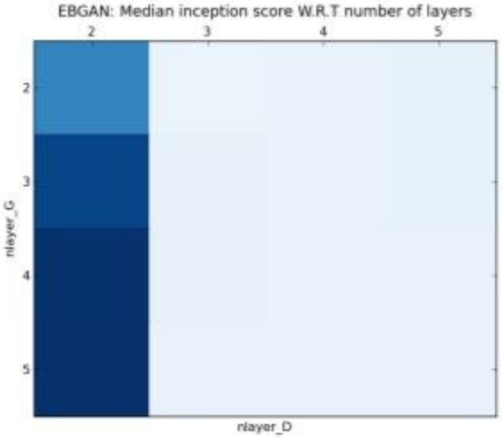


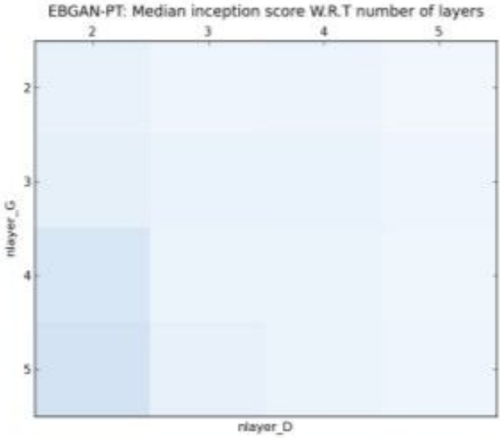


GAN vs EBGANS (MNIST)

Median







GAN vs EBGANS (LSUN & CELEBA), (IMAGE NET)

- Tested on LSUN and CelebA image dataset
- Test on both full images and cropped patches
- Use generator based on DCGAN from (Radford et al., 2015)
- While both DCGANs and EBGANs produce high quality images, EGBAN-PTs generates higher quality and more diverse images
- ImageNet
  - 128 x 128 and 256 x 256 (unprecedented resolution) images

Conclusion

- Combine GANs and auto-encoders and then give this combination an energy-based perspective to provide new approach to image generation
- EBGAN generally have better convergence patterns and scalability
  - Seamlessly works for given families of energy-based loss functions
- Future work would be in continuing to approach GANs from the energy based perspective in addition to developing more theoretical understanding of GANs in general

Additional References

- http://yann.lecun.com/exdb/publis/pdf/lecun-06.pdf
- https://arxiv.org/abs/1406.2661
- http://www.deeplearningbook.org/
- https://en.wikipedia.org/wiki/Stability_(learning_theory)