Energy-Based Generative Adversarial Networks

Wenhao Yang

School of Mathematical Sciences
Peking University

March 9, 2017

Outline

Energy-Based Model

- Try to learn a function f, which maps each point of an input space to a single scalar.

$$f : X \to R \tag{1}$$

  where the scalar is called "energy".

- Ideal function f: the desired configuration gets assigned low energies while the incorrect ones are given high energies.

- E.g:
  Supervised learning: low energy corresponds to correct label.
  Unsupervised learning: low energy corresponds to the data manifold.

Genrative Adversarial Networks

GAN are combined with the following parts:

- Discriminator
- Generator
- etc

Energy-Based Generative Adversarial Networks

- Discriminator is viewed as an energy function, without explicit probabilistic interpretation.
- Discriminator is trained to assign low energy values to regions of high data density.
- Generator is viewed as a trainable parameterized function that produces samples in regions of the space to which the generator assigns low energy.

EBGAN Model

- Notation:
  p_{data}: the underlying probability density of the distribution that produces the dataset.
  G: the generator which is trained to produce a sample $G(z)$
  p_z: A known distribution to provide a random vector z
  D: the discriminator which takes either real or generated samples and estimates the energy value $E \in R$

- Objectie functional:
  Given a positive margin m, a data sample x and a generated sample $G(z)$, defining the discriminator loss $\mathcal{L}_D$ and the generator loss $\mathcal{L}_G$:

$$\mathcal{L}_D(x, z) = D(x) + [m - D(G(z))]^+ \tag{2}$$

$$\mathcal{L}_G(z) = D(G(z)) \tag{3}$$

EBGAN Model

- Define:

$$V(G,D) = \int_{x,z} \mathcal{L}_D(x,z) p_{data}(x) p_z(z) dx dz \tag{4}$$

$$U(G,D) = \int_{z} \mathcal{L}_G(z) p_z(z) dz \tag{5}$$

- Aim: train the D to minimize V and train the G to minimize U
- Nash equilibrium of the system is a pair (G^*, D^*) that satisfies:

$$V(G^*,D^*) \leq V(G^*,D) \ \ \forall D \tag{6}$$

$$U(G^*,D^*) \leq U(G,D^*) \ \ \forall G \tag{7}$$

EBGAN Model

Theorem

If (D^*, G^*) *is a Nash equilibrium of the system,then* $p_G \overset{a.s}{=} p_{data}$ *and* $V(D^*, G^*) = m$

Theorem

Nash equilibrium of this system exists and is characterized by:
(a) $p_G \overset{a.s}{=} p_{data}$
(b) there exists a constant $\gamma \in [0, m]$ *such that* $D^*(x) \overset{a.s}{=} \gamma$

Auto-Encoders

The discriminator D is structured as an auto-encoder:

$$D(x) = ||Dec(Enc(x)) - x|| \tag{8}$$

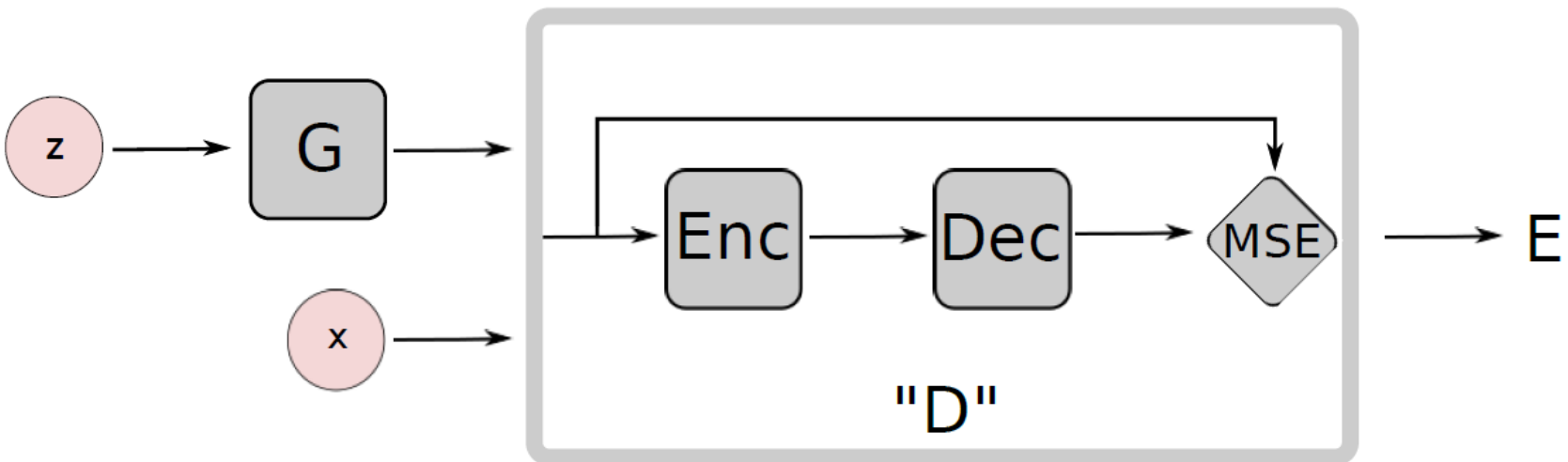

Figure 1: EBGAN architecture with an auto-encoder discriminator.

Auto-Encoders

Why Auto-Encoders?

- Offer a diverse targets for the discriminator
- Reconstruction loss will likely produce very different gradient directions within the minibatch, allowing for larger minibatch size without loss of efficiency.
- Given regularization, auto-encoders have the ability to learn an energy manifold without supervision or negative examples.

Problems:

- Auto-encoders may learn little more than an identity funcion, which means attributing low energy to the whole space.

Solution:

- Regularization: to give higher energy to points outside of the data manifold.

Auto-Encoders

Repelling Regularizer:

- Aim: to keep the model from producing samples that are clustered in one or a few modes of p_{data}
- Pulling-away(PT) effect:

$$f_{PT}(S) = \frac{1}{N(N-1)} \sum_i \sum_{j \neq i} \left(\frac{S_i^T S_j}{||S_i|| ||S_j||} \right)^2 \tag{9}$$

where $S \in R^{s \times N}$ denotes a batch of sample representations taken from the encoder output layer.

- PT term is used in the generator loss but not discriminator loss

Exhaustive Grid Search On MNITST

Table 1: Grid search specs

Settings	Description	EBGANs	GANs
`nLayerG`	number of layers in G	[2, 3, 4, 5]	[2, 3, 4, 5]
`nLayerD`	number of layers in D	[2, 3, 4, 5]	[2, 3, 4, 5]
`sizeG`	number of neurons in G	[400, 800, 1600, 3200]	[400, 800, 1600, 3200]
`sizeD`	number of neurons in D	[128, 256, 512, 1024]	[128, 256, 512, 1024]
`dropoutD`	if to use dropout in D	[true, false]	[true, false]
`optimD`	to use Adam or SGD for D	adam	[adam, sgd]
`optimG`	to use Adam or SGD for G	adam	[adam, sgd]
`lr`	learning rate	0.001	[0.01, 0.001, 0.0001]
#experiments:	-	**512**	**6144**

Exhaustive Grid Search On MNITST

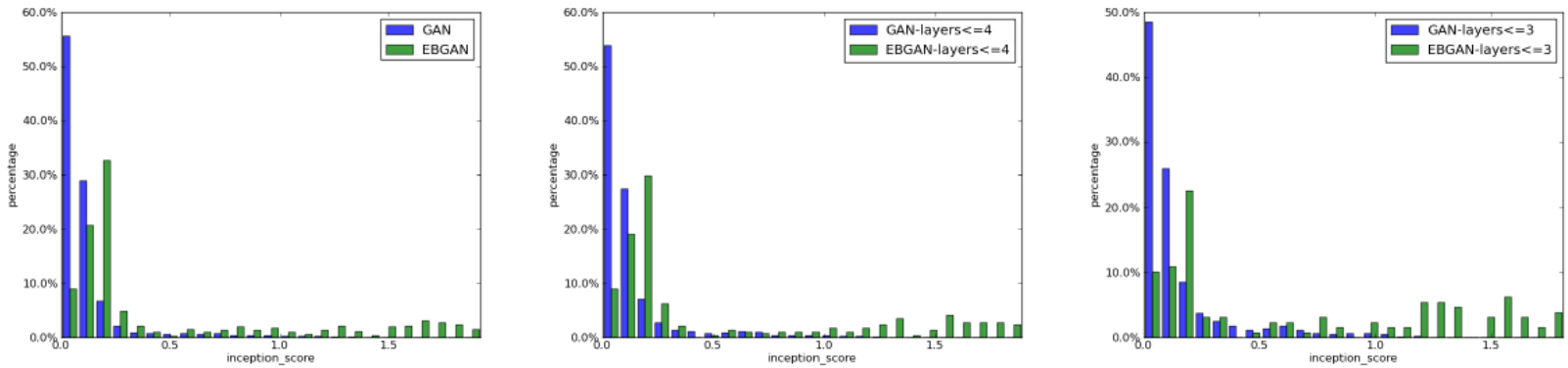

Figure 2: **(Zooming in on pdf file is recommended.)** Histogram of the inception scores from the grid search. The x-axis carries the inception score I and y-axis informs the portion of the models (in percentage) falling into certain bins. Left (a): general comparison of EBGANs against GANs; Middle (b): EBGANs and GANs both constrained by `nLayer[GD]<=4`; Right (c): EBGANs and GANs both constrained by `nLayer[GD]<=3`.

Exhaustive Grid Search On MNITST

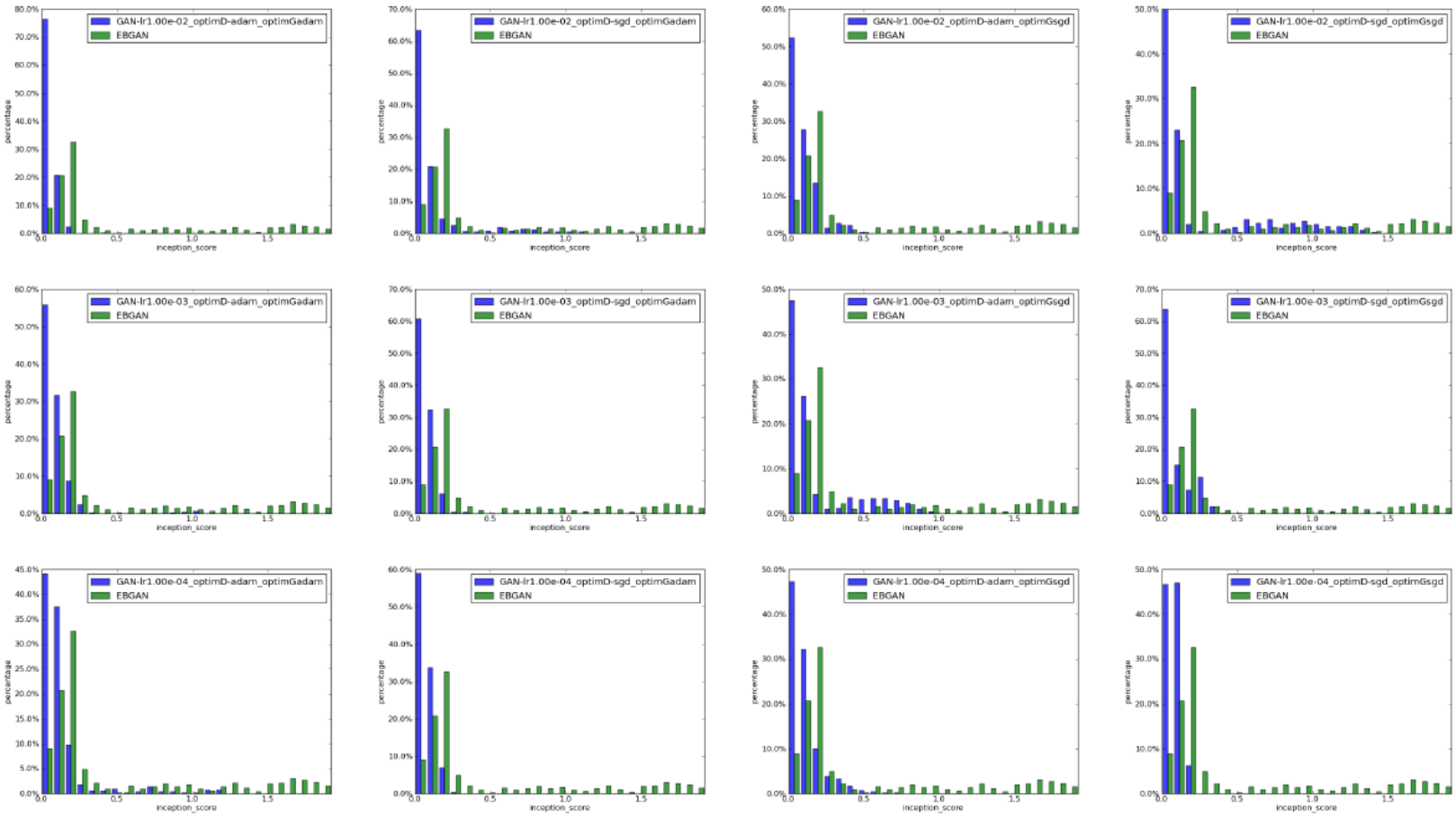

Figure 3: **(Zooming in on pdf file is recommended.)** Histogram of the inception scores grouped by different optimization combinations, drawn from `optimD`, `optimG` and `lr` (See text).

Exhaustive Grid Search On MNITST

Figure 1: Best GAN model

Exhaustive Grid Search On MNITST

Figure 2: Best EBGAN model

Exhaustive Grid Search On MNITST

Figure 3: Best EBGAN-PT model

Semi-Supervised Learning On MNIST

Positioning a bottom-layer-cost LN into an EBGAN framework.

Table 2: The comparison of LN bottom-layer-cost model and its EBGAN extension on PI-MNIST semi-supervised task. Note the results are error rate (in %) and they were averaged over 15 different random seeds.

model	100	200	1000
LN bottom-layer-cost, reported in Pezeshki et al. (2015)	1.69±0.18	-	1.05±0.02
LN bottom-layer-cost, reported in Rasmus et al. (2015)	1.09±0.32	-	0.90±0.05
LN bottom-layer-cost, reproduced in this work (see appendix D)	1.36±0.21	1.24±0.09	1.04±0.06
LN bottom-layer-cost within EBGAN framework	**1.04±0.12**	**0.99±0.12**	**0.89±0.04**
Relative percentage improvement	23.5%	20.2%	14.4%

LSUN & CELEBA

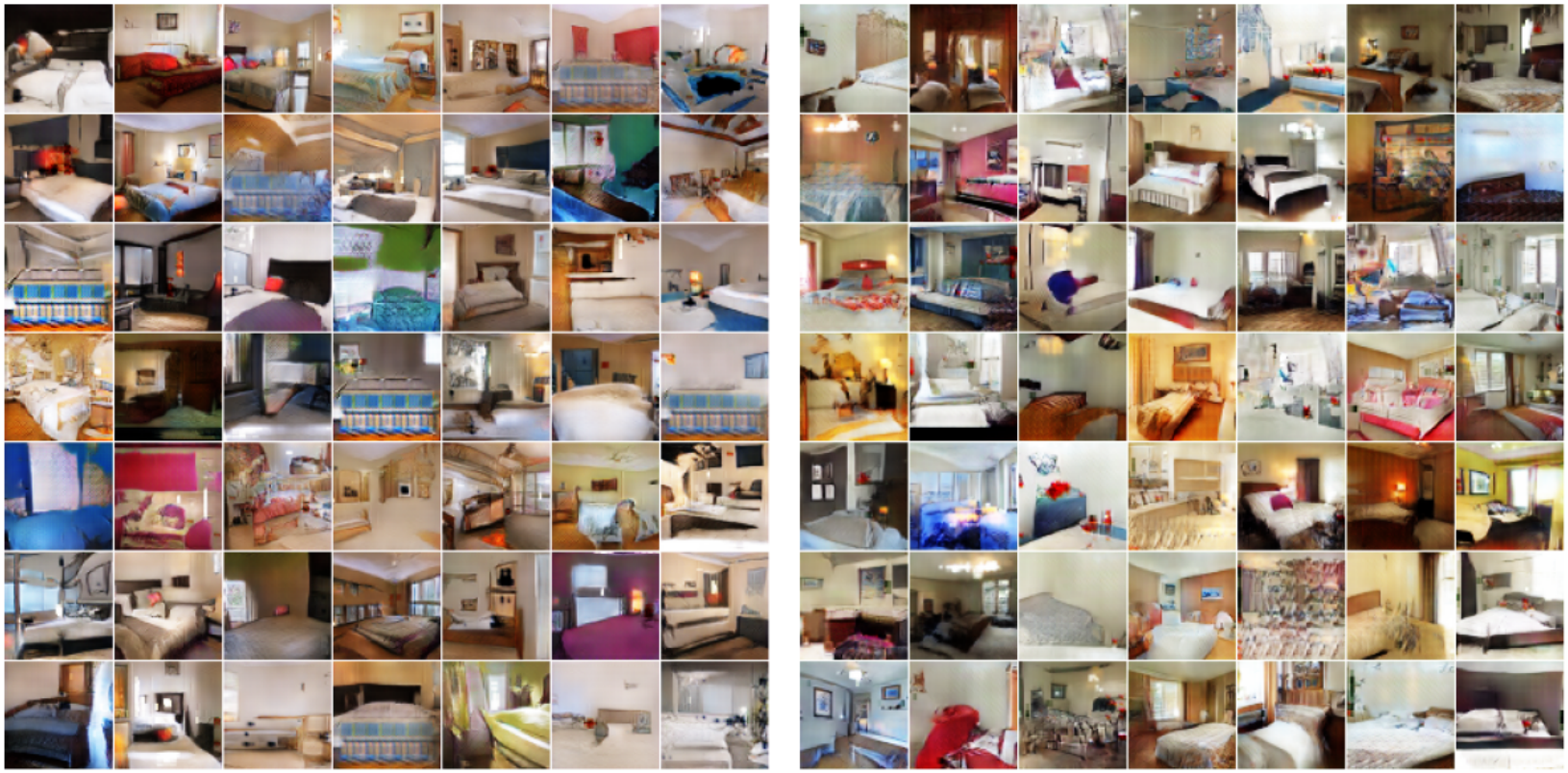

Figure 5: Generation from LSUN bedroom full-images. Left(a): DCGAN generation. Right(b): EBGAN-PT generation.

LSUN & CELEBA

Figure 6: Generation from CelebA face dataset. Left(a): DCGAN generation. Right(b): EBGAN-PT generation.

Imagenet

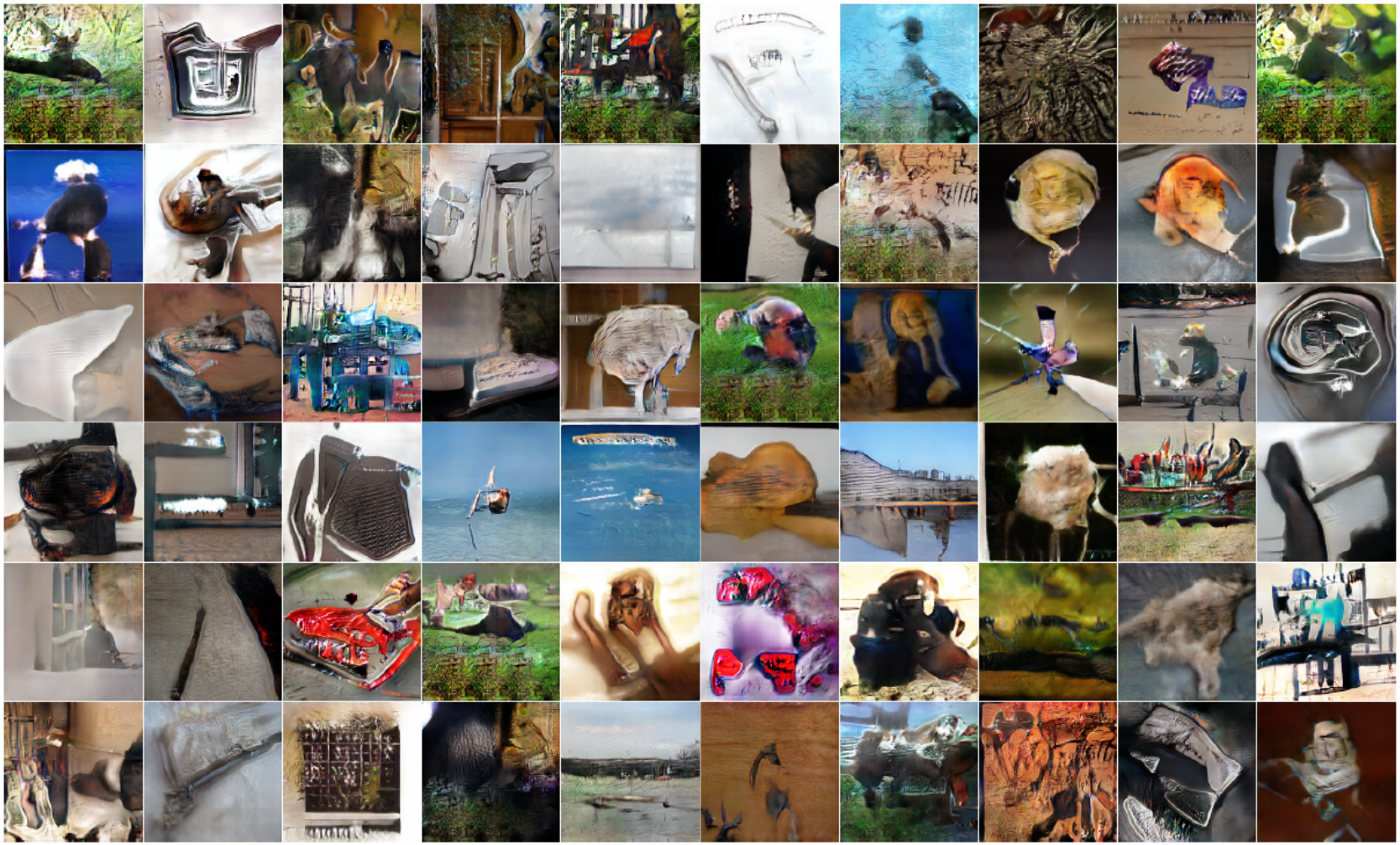

Figure 7: ImageNet 128×128 generations using an EBGAN-PT.

Imagenet

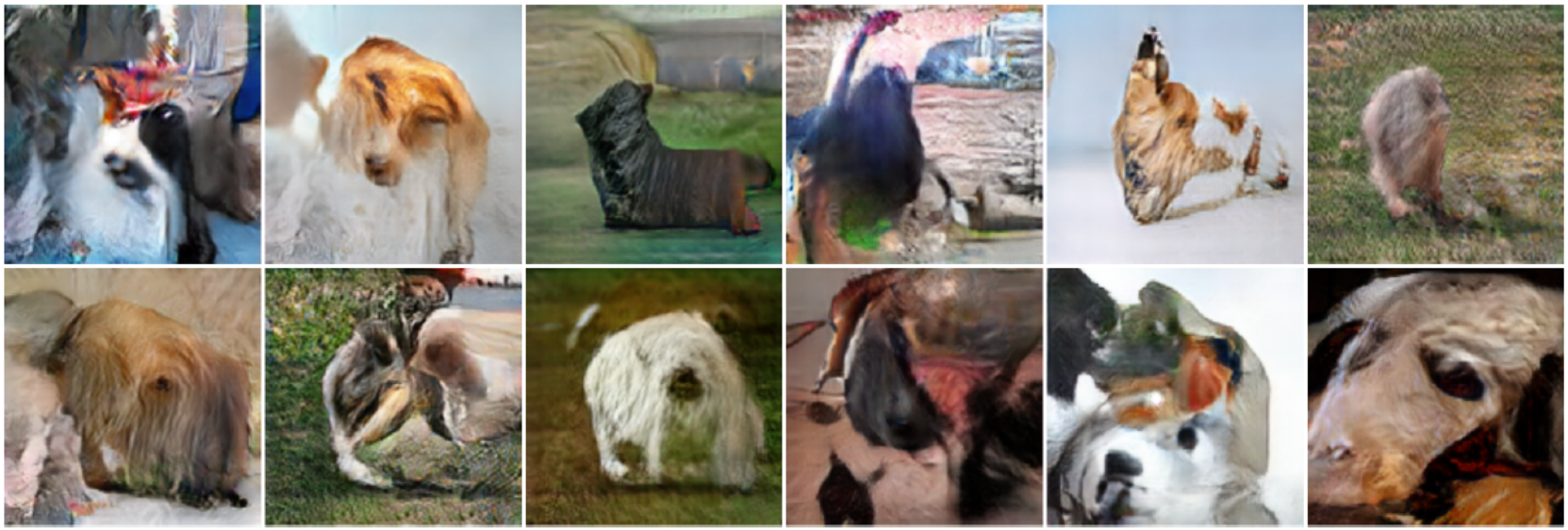

Figure 8: ImageNet 256 × 256 generations using an EBGAN-PT.

Reference

Reference:
[1] J ZhaoM MathieuY Lecun; Energy-based Generative Adversarial Network