A Study on Several Face Generator Models (DCGAN, RealNVP, VAE)

Sajjad Pakdamansavoji [1 2 3]

Abstract

This document is prepared to report the results of a study conducted as the final project of EECS 6327(probabilistic models in machine learning). It consists of seven major parts each of which being responsible for disseminating a unique aspect of this study. At the beginning a brief statement is made in the Abstract depicting a big picture of this study. With the general scope determined in the abstract, the study is formally ,and in detail presented in the Introduction. Once the problem is fully introduced, a complete literature review is given in the Related Works and Background section. In that section not only the previous works are examined but also some empirical results are also given for them. While these sections, so far, focused on previous works and theoretical aspects of out study, the following sections will focus on the empirical ones. To begin with the empirical results, the dataset, software, and hardware used in this study is given. Then for each model the results of training the networks from random weights are given. With that information in hand, a concise comparison of the models are given in the next section. In this section different aspects of the generative models are compared with each other including methodology, convergence speed and training stability. To wrap up the study a statement is given regarding the future works and the way this study can be extended. At the end related references are cited accordingly.

[*]Equal contribution [1]Department of Electrical Engineering and Computer Science, York University, Toronto, ON, CA [2]Vector Institute, Toronto, On, CA [3]Center of Vision Research, York University, Toronto, ON, CA. Correspondence to: Sajjad Pakdamansavoji <savoji@yorku.ca>.

Project Report for EECS6327 (Fall 2020), Department of Electrical Engineering and Computer Science, York University.

1. Introduction

Generating real-like face images has been an area of extensive research for many years. The result of such extensive research has been the invention of several methodologies for generative tasks. In this paper we study three specific models for such a task. The first model we study is a member of Generative Adversarial Networks(GAN) called Deep Convolutional GAN(DCGAN). While GAN models follow the generator-discriminator approach, there are other models which try to estimate the underlying distribution of faces and generate instances based on that estimated distribution. From the latter category we study Variational Autoencoder(VAE) and Real Values Non-volume Preserving Normalizing Flow(RealNVP).

Face generation has a number of applications in the game industry, media industry, computer vision and virtual reality to name a few. While its applications are quite diverse in nature, state of the art models still suffer the ability to generate consistent and diverse images. For example, in the case of a computer game, the task of generating a unique face from several points of view remains challenging if not impossible. Therefore, comparing these models in terms of the quality of generation can greatly benefit the research community as well as the industry.

Our main contributions in this study are as followed:

- Reviewing the theoretical setup of the DCGAN model
- Reviewing the theoretical setup of the RealNVP model
- Reviewing the theoretical setup of the VAE model
- Compering the aforementioned models in terms of methodology, deformation, blurriness, training stability, convergence speed, and interpretability

The first three contributions are elaborated in section 2, 3, and 4. While the last contribution is given in section5. In section 6, possible research paths based on this study are presented shortly.

2. Related Works and Background

In this section a brief review of related works is given. I start by explaining the GAN methodology and continue to describe a more specific case of GANs called DCGAN. Then, with the same approach, a general view of autoencoders is given and that general view is narrowed down to a specific case of variational autoencoders. Last but not least, a schema for probability estimation is given which is called Normalizing Flow(NF). This schema is further studied for the case of real valued non-volume preserving flows called (RealNVP).

2.1. Generative Adversarial Networks(GAN)

The methodology of generative adversarial networks was first introduced in 2014 by Ian J. Goodfellow (Goodfellow et al., 2014). In that work they proposed a new estimation methodology for generative models in which two networks are trained at the same time. These networks are a Generative model G capturing the data distribution and a discriminative model D estimating the probability of distinguishing between fake and real data. This setup is similar to that of a mini-max two player game. In the space of all arbitrary functions G and D there exist at least one solution in which G successfully captures the data distribution while D's probability of making a mistake equals 0.5 everywhere. In the case that these networks(G and D) are neural models, one can train those models using gradient based optimization approaches such as gradient descent.

Deep learning is promised to find rich, hierarchical models representing the underlying probability distribution over a variety of data commonly used in the literature including natural images (Bengio, 2009). So far the most impressive models in the literature have only involved discrimination methodologies in which they estimate a probability distribution of the data being associated with a class label in a finite set.(Hinton et al., 2012a)(Krizhevsky et al., 2012) It is assumed that such a successful performance is primarily due to backpropagation, dropout algorithms, and using piecewise linear units with well-behaved gradients.(Glorot et al., 2011)(Goodfellow et al., 2013)(Jarrett et al., 2009)

While the family of deep discriminative models have been well-studied, conversely, deep generative models have had less of an impact. This is due to the fact that the problem being addressed, which is approximating many intractable probabilistic computations, is way too difficult when it is fused with maximum likelihood estimation and related strategies. Also it is worth mentioning that piecewise linear units has not yet proved to be useful as a leverage in the context of generative models. Therefore they proposed a new generative model estimating procedure that sidesteps these difficulties.

In the proposed framework, the generative model G is facing a discriminative adversary D learning to distinguish whether a sample is from the model distribution or the data distribution. This setup is analogous to the adversary of forgers and critics in the domain of art for instance. The competitive nature in this setup drives both sides to improve their methods until the counterparts are indistinguishable from the original distribution. This framework can be adapted to different training algorithms for many kinds of models and optimization algorithms. In this study, we will review the case in which both G and D are neural models. This setup is commonly referred to as adversarial nets. This being the case, one can train both models using only backpropagation while being able to sample from the generative part only using a forward pass.(Hinton et al., 2012b)

The adversarial modeling framework is most straightforward to apply when the models are both neural networks. To learn the generators distribution p_g over data x, we define a prior on input noise variable $p_z(z)$ then represent a mapping to data space as $G(z;\theta_g)$, where G is a differentiable function represented by a neural network with its weights being wrapped into a single variable θ_g. We also define a second neural model $D(x;\theta_d)$ that outputs a single scalar representing the probability that x came from the data rather than p_g. We train D to maximize the probability of assigning the correct label to both training examples and samples from G. One should simultaneously train G to minimize $log(1 - D(G(z)))$. In other words G and D play the following two player minimax game with the value function $V(G, D)$:(Goodfellow et al., 2014)

$$\begin{aligned} min_G\, max_D V(G,D) = {} & E_{p_{data}(x)}\left\{log(D(x))\right\} \\ & + E_{p_z(z)}\left\{1 - log(D(G(z)))\right\} \end{aligned} \tag{1}$$

The framework of GAN comes with its own perks and pitfalls relative to previous modeling frameworks. The disadvantages are mainly that there is no explicit representation of $p_g(x)$, and that D must be synchronized well with G during training (in particular, G must not be trained too much without updating D, in order to avoid "the Helvetica scenario" in which G collapses too many values of z to the same value of x to have enough diversity to model p_{data}). The aforementioned advantages are mainly computational. GANS may also gain some statistical advantage from the generator network not being updated directly with data examples, but only with gradients flowing through the discriminator. This means that components of the input are not copied directly into the generator's parameters. Another advantage of adversarial networks is that they can represent very sharp, even degenerate distributions, while methods based on Markov chains require that the distribution be somewhat

blurry in order for the chains to be able to mix between modes.(Goodfellow et al., 2014)

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets with 1 switch per iteration.

Input: data X, generator G, discriminator D, rep K
for number of training iterations **do**
  for k steps **do**
    - sample a mini batch of m noise samples $\{z^{(1)}, z^{(2)}, ..., z^{(m)}\}$ from noise prior $p_g(z)$
    - sample a mini batch of m data samples $\{x^{(1)}, x^{(2)}, ..., x^{(m)}\}$ from distribution $p_{data}(x)$
    - Update the discriminator by ascending its stochastic gradient:

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

  end for
  - sample a mini batch of m noise samples $\{z^{(1)}, z^{(2)}, ..., z^{(m)}\}$ from noise prior $p_g(z)$
  - Update the generator by descending its stochastic gradient:

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

end for

2.2. Deep convolutional GAN (DCGAN)

In recent years, supervised learning with convolutional networks(CNNs) has been extensively adopted in the field of computer vision. While unsupervised learning with CNNs has not been adopted to that extent. In the DCGAN paper (Radford et al., 2015a) authors tried to close the gap between the success of CNNs for supervised learning and unsupervised learning. They introduce the class of networks called DCGAN which have certain architectural constraints and establish that they might be strong options for unsupervised learning. They trained such a model on several datasets to derived enough evidence that their proposed networks learn a hierarchy of representations from object parts to scenes in both the generator and discriminator. Additionally, they use the learned features for novel tasks to demonstrate their applicability in different visual tasks.

Learning well-structured and reusable feature representation for a large dataset of unlabeled images has been an active research area. In the context of computer vision, one can leverage the practically unlimited amount of unlabeled images and videos to learn good intermediate representations, which can then be used on a variety of supervised learning tasks such as image classification. The authors propose that one way to build good image representations is by training generative adversarial networks (Goodfellow et al., 2014), and later reusing parts of the generator and discriminator as feature extractors for supervised tasks. GANs provide an attractive alternative to maximum likelihood techniques. One can additionally argue that their learning process and the lack of a heuristic cost function (such as pixel-wise independent mean-square error) are attractive to representation learning. GANs have been known to be unstable to train, often resulting in generators that produce nonsensical outputs. There has been very limited published research in trying to understand and visualize what GANs learn, and the intermediate representations of multi-layer GANs. In the DCGAN paper authors make the following contributions. In the DCGAN paper (Radford et al., 2015a) authors made the following contributions:

- they propose and evaluate a set of constraints on the architectural topology of Convolutional GANs that make them stable to train in most settings. They name this class of architectures Deep Convolutional GANs (DCGAN)
- They use the trained discriminators for image classification tasks, showing competitive per- formance with other unsupervised algorithms.
- They visualize the filters learnt by GANs and empirically show that specific filters have learned to draw specific objects.
- They show that the generators have interesting vector arithmetic properties allowing for easy manipulation of many semantic qualities of generated samples.

Previously, many attempts to scale up GANs using CNNs to model images have not been as successful as one might desire. For instance, the authors of the paper LAPGAN (Denton et al., 2015) were motivated by this fact to construct an alternative way to iteratively upscale low resolution fake images which can be modeled more reliably to images with bigger size. The authors of the paper DCGAN (Radford et al., 2015a) also encountered similar difficulties trying to scale GANs using CNN architectures commonly used in supervised learning. They, however, were able to find a family of architectures that resulted in stable training across a range of datasets while allowing for training higher resolution and deeper generative models. They approach was primarily adopting four important updates to the architecture of CNN demonstrated below:

1. Use all convolutional net (Springenberg et al., 2014) which replaces deterministic spatial pooling functions (such as maxpooling) with strided convolutions, allowing the network to learn its own spatial downsampling

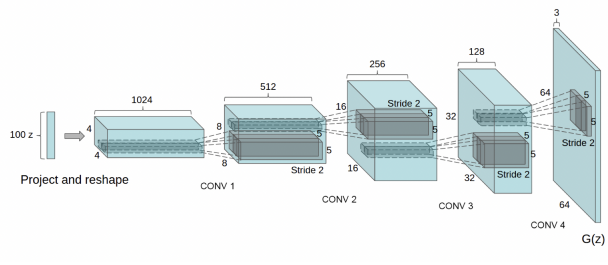


Figure 1. DCGAN generator used for LSUN scene modeling. A 100 dimensional uniform distribu- tion Z is projected to a small spatial extent convolutional representation with many feature maps. A series of four fractionally-strided convolutions (in some recent papers, these are wrongly called deconvolutions) then convert this high level representation into a 64 × 64 pixel image. Notably, no fully connected or pooling layers are used.

2. Eliminating fully connected layers on top of convolutional features. The strongest example of this is global average pooling which has been utilized in state of the art image classification models. We found global average pooling increased model stability but hurt convergence speed. A middle ground of directly connecting the highest convolutional features to the input and output respectively of the generator and discriminator worked well. See Fig 1 for an abstract view of the architecture.

3. Batch Normalization (Ioffe & Szegedy, 2015b)stabilizes learning by normalizing the input to each unit to have zero mean and unit variance. This helps deal with training problems that arise due to poor initialization and helps gradient flow in deeper models. This proved critical to get deep generators to begin learning, preventing the generator from collapsing all samples to a single point which is a common failure mode observed in GANs.

4. Use ReLU activation (Nair & Hinton, 2010) is used in the generator with the exception of the output layer which uses the Tanh function. We observed that using a bounded activation allowed the model to learn more quickly to saturate and cover the color space of the training distribution. Within the discriminator we found the leaky rectified activation (Xu et al., 2015) to work well, especially for higher resolution modeling. This is in contrast to the original GAN paper, which used the maxout activation (Goodfellow et al., 2014).

In the DCGAN paper (Radford et al., 2015a) authors proposed a more stable group of neural models for training generative adversarial networks and they show results that adversarial networks learn good representations of images for supervised learning and generative modeling. There are still some forms of model instability remaining - they noticed as models are trained longer they sometimes collapse a subset of filters to a single oscillating mode.

2.3. Variational Autoencoder(VAE)

The VAE paper (Kingma & Welling, 2013) discuss the question of how one can perform efficient inference and learning in directed probabilistic models, in the presence of continuous latent variables with intractable posterior distributions, and large datasets. To answer it, the authors introduce a stochastic variational inference and learning algorithm that scales to large datasets and, under some mild differentiability conditions, even works in the intractable case. Their contribution is two-fold. First, they show that a reparameterization of the variational lower bound yields a lower bound estimator that can be straightforwardly optimized using standard stochastic gradient methods. Second, they show that for i.i.d. datasets with continuous latent variables per datapoint, posterior inference can be made especially efficient by fitting an approximate inference model (also called a recognition model) to the intractable posterior using the proposed lower bound estimator. Theoretical advantages are reflected in experimental results.(Kingma & Welling, 2013)

The variational Bayesian (VB) approach involves the optimization of an approximation to the intractable posterior. Unfortunately, the common mean-field approach requires analytical solutions of expectations w.r.t. the approximate posterior, which are also intractable in the general case. We show how a reparameterization of the variational lower bound yields a simple differentiable unbiased estimator of the lower bound; this SGVB (Stochastic Gradient Variational Bayes) estimator can be used for efficient approximate posterior inference in almost any model with continuous latent variables and/or parameters, and is straightforward to optimize using standard stochastic gradient ascent techniques.(Kingma & Welling, 2013)

For the case of an i.i.d. dataset and continuous latent variables per datapoint, they propose the AutoEncoding VB (AEVB) algorithm. In the AEVB algorithm we make inference and learning especially efficient by using the SGVB estimator to optimize a recognition model that allows us to perform very efficient approximate posterior inference using simple ancestral sampling, which in turn allows us to efficiently learn the model parameters, without the need of expensive iterative inference schemes (such as MCMC) per datapoint. The learned approximate posterior inference model can also be used for a host of tasks such as recognition, denoising, representation and visualization purposes. When a neural network is used for the recognition model, we arrive at the variational auto-encoder.(Kingma & Welling,

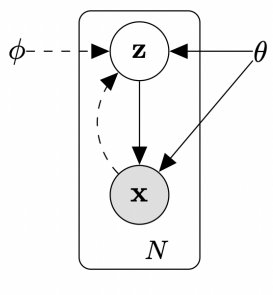


Figure 2. The type of directed graphical model under consideration. Solid lines denote the generative model $p_\theta(z)p_\theta(x|z)$, dashed lines denote the variational approximation $q_\phi(z|x)$ to the intractable posterior $p_\theta(z|x)$. The variational parameters ϕ are learned jointly with the generative model parameters θ.

2013) A general view of this model is depicted in Fig 2.

The marginal likelihood is composed of a sum over the marginal likelihoods of individual data points $log(p_\theta(x^1, ..., x^N)) = \sum_{i=1}^{N} log(p_\theta(x^i))$ which can each be written as

$$logp_\theta(x^i) = D_{KL}\left(q_\phi(z|x^i), p_\theta(z|x^i)\right) + L(\theta, \phi; x^i) \quad (2)$$

The first RHS term is the KL divergence of the approximate from the true posterior. Since this KL-divergence is non-negative, the second RHS term $L(\theta, \phi; x^i)$ is called the (variational) lower bound on the marginal likelihood of datapoint i, and can be written as:

$$L(\theta, \phi; x^i) = E_{q_\phi}\{-logq_\phi(z|x) + logp_\theta(z|x)\} \quad (3)$$

The authors have introduced a novel estimator of the variational lower bound, Stochastic Gradient VB (SGVB), for efficient approximate inference with continuous latent variables. The proposed estimator can be straightforwardly differentiated and optimized using standard stochastic gradient meth- ods. For the case of i.i.d. datasets and continuous latent variables per datapoint we introduce an efficient algorithm for efficient inference and learning, Auto-Encoding VB (AEVB), that learns an approximate inference model using the SGVB estimator. The theoretical advantages are reflected in experimental results.(Kingma & Welling, 2013)

Algorithm 2 Minibatch version of the Auto-Encoding VB (AEVB) algorithm for Either of the two SGVB estimators

$\theta, \phi \leftarrow$ initialize parameters
repeat
  Initialize $noChange = true$.
  $X^M \leftarrow$ random minibatch of M points
  $\epsilon \leftarrow$ Random samples from noise distribution $p(\epsilon)$
  $g \leftarrow \nabla_{\theta,\phi} L_M(X^M, \epsilon, \theta, \phi)$
  $\theta, \phi \leftarrow$ update params using g
until convergence of θ and ϕ
Return θ and ϕ

2.4. Real-Valued Non-volume Preserving NF(RealNVP)

Unsupervised learning of probabilistic models is a central yet challenging problem in machine learning. Specifically, designing models with tractable learning, sampling, inference and evaluation is crucial in solving this task. We extend the space of such models using real-valued non-volume preserving (real NVP) transforma- tions, a set of powerful, stably invertible, and learnable transformations, resulting in an unsupervised learning algorithm with exact log-likelihood computation, exact and efficient sampling, exact and efficient inference of latent variables, and an interpretable latent space. In the RealNVP paper (Dinh et al., 2016) the authors demonstrate its ability to model natural images on four datasets through sampling, log-likelihood evaluation, and latent variable manipulations.

The domain of representation learning has undergone tremendous advances due to improved super- vised learning techniques. However, unsupervised learning has the potential to leverage large pools of unlabeled data, and extend these advances to modalities that are otherwise impractical or impossible. One principled approach to unsupervised learning is generative probabilistic modeling. Not only do generative probabilistic models have the ability to create novel content, they also have a wide range of reconstruction related applications including inpainting (Theis & Bethge, 2015) (van den Oord et al., 2016) (Sohl-Dickstein et al., 2015), denoising (Ballé et al., 2015), colorization (Zhang et al., 2016), and super-resolution (Bruna et al., 2015). As data of interest are generally high-dimensional and highly structured, the challenge in this domain is building models that are powerful enough to capture its complexity yet still trainable. We address this challenge by introducing real-valued non-volume preserving (real NVP) transformations, a tractable yet expressive approach to modeling high-dimensional data. This model can perform efficient and exact inference, sampling and log-density estimation of data points. Moreover, the architecture presented in this paper enables exact and efficient reconstruction of input images from the hierarchical features extracted by this model.

In this paper, the authors tackle the problem of learning highly nonlinear models in high-dimensional continuous spaces through maximum likelihood. In order to optimize the log-likelihood, they introduce a more flexible class of architectures that enables the computation of log-likelihood on continuous data using the change of variable formula. Building on our previous work in (Dinh et al., 2014), we define a powerful class of bijective functions which enable exact and tractable density evaluation and exact and tractable

inference. Moreover, the resulting cost function does not rely on a fixed form reconstruction cost such as square error (Larsen et al., 2015) (Radford et al., 2015b), and generates sharper samples as a result. Also, this flexibility helps us leverage recent advances in batch normalization (Ioffe & Szegedy, 2015a) and residual networks (He et al., 2015) (He et al., 2016) to define a very deep multi-scale architecture with multiple levels of abstraction.

Given an observed data variable $x \in X$, a simple prior probability distribution p_z on a latent variable $z \in Z$, and a bijection $f : X \to Z$ (with $g = f^1$), the change of variable formula defines a model distribution on X by

$$p_X(x) = p_Z(f(x)) \left| det \left(\frac{\partial f(x)}{\partial x^T} \right) \right| \tag{4}$$

$$log(p_X(x)) = log(p_Z(f(x))) + log \left(\left| det \left(\frac{\partial f(x)}{\partial x^T} \right) \right| \right) \tag{5}$$

Computing the Jacobian of functions with high-dimensional domain and codomain and computing the determinants of large matrices are in general computationally very expensive. This combined with the restriction to bijective functions makes Equation 4 appear impractical for modeling arbitrary distributions.

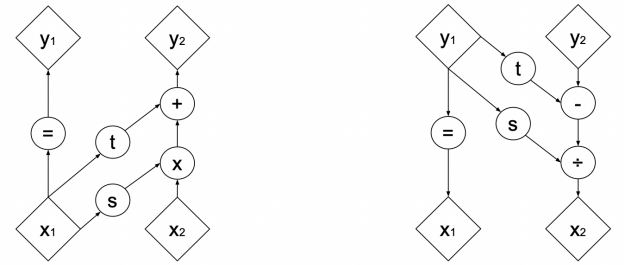
As shown however in (Dinh et al., 2014), by careful design of the function f , a bijective model can be learned which is both tractable and extremely flexible. As computing the Jacobian determinant of the transformation is crucial to effectively train using this principle, this work exploits the simple observation that the determinant of a triangular matrix can be efficiently computed as the product of its diagonal terms.

The authors of RealNVP (Dinh et al., 2016) built a flexible and tractable bijective function by stacking a sequence of simple bijections. In each simple bijection, part of the input vector is updated using a function which is simple to invert, but which depends on the remainder of the input vector in a complex way. We refer to each of these simple bijections as an affine coupling layer. Given a D dimensional input x and $d < D$, the output y of an affine coupling layer follows the equations and a graph of the coupling layer is given in Fig 3

$$y_{1:d} = x_{1:d} \tag{6}$$

$$y_{d+1:D} = x_{d+1:D} \odot exp\left(s(x_{1:d}) + t(x_{1:d})\right) \tag{7}$$

This paper (Dinh et al., 2016) presented a technique bridging the gap between auto-regressive models, variational autoencoders, and generative adversarial networks. Like auto-regressive models, it allows tractable and exact log-likelihood evaluation for training. It allows however a much more flexible functional form, similar to that in the generative model of variational autoencoders. This allows for fast and exact sampling from the model distribution. Like GANs, and unlike variational autoencoders, our technique does not require the use of a fixed form reconstruction cost, and instead defines a cost in terms of higher level features, generating sharper images. Finally, unlike both variational autoencoders and GANs, our technique is able to learn a semantically meaningful latent space which is as high dimensional as the input space.



(a) Forward propagation (b) Inverse propagation

Figure 3. Computational graphs for forward and inverse propagation. A coupling layer applies a simple invertible transformation consisting of scaling followed by addition of a constant offset to one part x2 of the input vector conditioned on the remaining part of the input vector x1. Because of its simple nature, this transformation is both easily invertible and possesses a tractable determinant. However, the conditional nature of this transformation, captured by the functions s and t, significantly increase the flexibility of this otherwise weak function. The forward and inverse propagation operations have identical computational cost.

3. Software, Hardware, and Data

3.1. Dataset

For this study the CelebA dataset was chosen as it provides a rich dataset of unlabeled face images which can be used for all the aforementioned neural models. CelebFaces Attributes Dataset (CelebA) is a large-scale face attributes dataset with more than 200K celebrity images, each with 40 attribute annotations. The images in this dataset cover large pose variations and background clutter. CelebA has large diversities, large quantities, and rich annotations, including 10,177 number of identities, 202,599 number of face images, and 5 landmark locations, 40 binary attributes annotations per image. The dataset can be employed as the training and test sets for the following computer vision tasks: face attribute recognition, face recognition, face detection, landmark (or facial part) localization, and face editing synthesis.

3.2. Hardware

As this projected involved training several neural models, all of the experiments were executed on a GeForce GTX 1080 graphic card. This graphic card has 2560 cuda cores, a processor clock of 1733 MHz, a memory clock of 10 Gbps, standard memory config 10 GB, and a memory interface of 256 bit.

3.3. Software

All of models were codded using a high level python based framework called pytorch. PyTorch is an open source machine learning library based on the Torch library, used for applications such as computer vision and natural language processing, primarily developed by Facebook's AI Research lab (FAIR). It is free and open-source software released under the Modified BSD license. Although the Python interface is more polished and the primary focus of development, PyTorch also has a C++ interface. PyTorch provides two high-level features: Tensor computing (like NumPy) with strong acceleration via graphics processing units (GPU) and Deep neural networks built on a type-based automatic differentiation system.

4. Experiments

In total three generative models were trained from scratch for this study which I will go over the details and results of each experiments in the following subsections.

4.1. DCGAN results

DCGAN was trained with the following specifications:

Batch size = 128 , Size of latent noize = 100 , Number of feature maps in generator = 64 , Number of feature maps in discriminator = 64 , Learning Rate = 0.0002 , Optimizer= Adam(beta1 = 0.5) , Loss = Binary Cross Entropy , Num Epochs = 100

The results of this experiments are two-fold. First the learning curve of DCGAN is depicted in Figure 4. Please note that the learning curve of GANs are not similar to that of regular classifiers as the gnerator and the discriminator are constantly in competition hence resulting fluctuated trends in both generator and discriminator's learning curves.

Also a sample of fake and real images are given in the following figure. Please note that this fake images are generated using limited resources and yet they are comparable with the real ones.

4.2. RealNVP Results

The realNVP model was train with the following specifications:

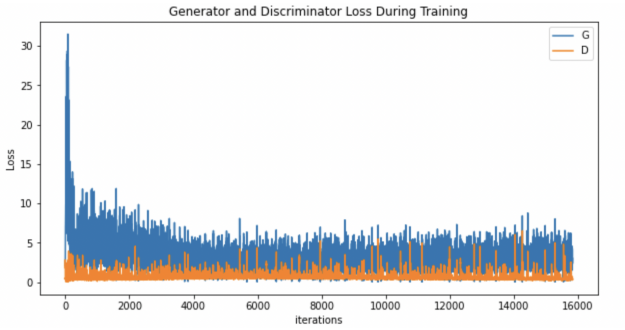


Figure 4. Generator Loss and Discriminator Loss in DCGAN.

Figure 5. A sample of real images from CelebA.

Figure 6. Fake faces generated using DCGAN.

Figure 7. Fake faces generated using realNVP.

Batch size = 64, Base dimension = 32, Learning Rate = 0.001, Optimizer= Adamax, Loss = Logit Log Probability ,Num Epochs = 50

As the result of this study a grid of fake faces samples from the estimated images distribution is given in the following figure. The quality of images are reasonable while considering the amount of resource and training time dedicated for this model.

4.3. VAE Results

The VAE model was trained with the following specifications:

Batch size = 144 , Latent dimension = 128 , Learning Rate = 0.005 , Scheduler Gamma = 0.95 , Optimizer= SGD , Loss = Log Probability + KLD , Num Epochs = 30

In Figure 8 a generated and reconstructed grid of faces from the VAE model are depicted.

Figure 8. Empirial results for VAE.

5. Comparison

In this section the three models studied in this work will be compares. The comparison will be based on several aspect which are given in the following.

- **methodology**: These models use vary different methodologies for data generation. GAN models follow an adversarial methodology while VAE and NF follow a direct distribution approximation methodology.
- **Deformation**: As depicted in the figures 6, 7, and 8 many of the faces are deformed; meaning that they do not follow the natural symmetric shape of a face. Based on that property, my results show that VAE generates more symetric faces while RealNVP and DCGAN suffer heavily from deformation. **Blur**: Another unsuitable property of generated images is being blurry. With regards to that issue, VAE performs the worst while DCGAN and RealNVP have sharper images.
- **Training Stability**: In contrast with VAE and NF, GAN models do not have a stable training process. This is because the generator and discriminator are not trained at the same time which might lead to mode collapse.
- **Convergence Speed**: The DCGAN model was trained for 100 epochs of 0.5 min resulting a 50 min training time. The RealNVP was trained for 50 epcochs of 2 minis resulting in a 100 min training time. The VAE was trained for 30 epchs of 6 min resulting in a 180 min training time. With that results their convergance speed is ranked as DCGAN ¿ RealNVP ¿ VAE.

6. Conclusion

In this study three neural models were compared with each other. This comparison covered a wide variety of computational aspects as well as visual criterion. In sum, the DCGAN model had the fastest convergence speed while its quality of generation was worst due to deformation and mode collapse. VAE had the slowest convergence speed while having the best image generation quality in terms of symmetry and deformation. However it heavily suffers from blurriness. RealNVP, based on our results was in the sweep spot meaning that it had a fairly fast convergence speed while the quality of its generation being comparable to the real data generating distribution.

References

Ballé, J., Laparra, V., and Simoncelli, E. P. Density modeling of images using a generalized normalization transformation. 2015.

Bengio, Y. *Learning deep architectures for AI*. Now Publishers Inc, 2009.

Bruna, J., Sprechmann, P., and LeCun, Y. Super-resolution with deep convolutional sufficient statistics, 2015.

Denton, E., Chintala, S., Szlam, A., and Fergus, R. Deep generative image models using a laplacian pyramid of adversarial networks, 2015.

Dinh, L., Krueger, D., and Bengio, Y. Nice: Non-linear independent components estimation, 2014.

Dinh, L., Sohl-Dickstein, J., and Bengio, S. Density estimation using real nvp, 2016.

Glorot, X., Bordes, A., and Bengio, Y. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pp. 315–323. JMLR Workshop and Conference Proceedings, 2011.

Goodfellow, I. J., Warde-Farley, D., Mirza, M., Courville, A., and Bengio, Y. Maxout networks. 2013.

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

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition, 2015.

He, K., Zhang, X., Ren, S., and Sun, J. Identity mappings in deep residual networks, 2016.

Hinton, G., Deng, L., Yu, D., Dahl, G., rahman Mohamed, A., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T., and Kingsbury, B. Deep neural networks for acoustic modeling in speech recognition. *Signal Processing Magazine*, 2012a.

Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R. Improving neural networks by preventing co-adaptation of feature detectors, 2012b.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015a.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015b.

Jarrett, K., Kavukcuoglu, K., Ranzato, M., and LeCun, Y. What is the best multi-stage architecture for object recognition? In *2009 IEEE 12th International Conference on Computer Vision, ICCV 2009*, Proceedings of the IEEE International Conference on Computer Vision, pp. 2146–2153, 2009. ISBN 9781424444205. doi: 10.1109/ICCV.2009.5459469. Copyright: Copyright 2010 Elsevier B.V., All rights reserved.; 12th International Conference on Computer Vision, ICCV 2009 ; Conference date: 29-09-2009 Through 02-10-2009.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes, 2013.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Larsen, A. B. L., Sønderby, S. K., Larochelle, H., and Winther, O. Autoencoding beyond pixels using a learned similarity metric, 2015.

Nair, V. and Hinton, G. E. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pp. 807–814, Madison, WI, USA, 2010. Omnipress. ISBN 9781605589077.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015a.

Radford, A., Metz, L., and Chintala, S. Unsupervised representation learning with deep convolutional generative adversarial networks, 2015b.

Sohl-Dickstein, J., Weiss, E. A., Maheswaranathan, N., and Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics, 2015.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. Striving for simplicity: The all convolutional net, 2014.

Theis, L. and Bethge, M. Generative image modeling using spatial lstms. In Cortes, C., Lawrence, N., Lee, D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/2b6d65b9a9445c4271ab9076ead5605a-Paper.pdf.

van den Oord, A., Kalchbrenner, N., and Kavukcuoglu, K. Pixel recurrent neural networks, 2016.

Xu, B., Wang, N., Chen, T., and Li, M. Empirical evaluation of rectified activations in convolutional network, 2015.

Zhang, R., Isola, P., and Efros, A. A. Colorful image colorization, 2016.