STABLE AND IMPROVED GENERATIVE ADVERSARIAL NETS (GANS): A CONSTRUCTIVE SURVEY

Guanghao Zhang[⋆] *Enmei Tu*[†] *Dongshun Cui*[≀]

[⋆] School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore
[†] Rolls-Royce@NTU Corporate Lab, Nanyang Technological University, Singapore
[≀] Interdisciplinary Graduate School, Energy Research Institute@NTU, Singapore

ABSTRACT

In this paper, we present a general and applicable adversarial training framework based on a comprehensive survey, not limited to straightforward GANs related works, also including shallow neural networks and reinforcement learning. Concentrating on challenging face synthesis task, we summarize a stable training pipeline: 1) booting training procedure with noise injection; 2) fixing weights of fully connected layer in generator to improve performance further; 3) involving Markov decision module to dynamically choose learning rates of discriminator and generator respectively. Finally in experiments, we highlight a mutual evaluation criterion over entropy score based on a pre-trained classifier and manual voting.

Index Terms— image synthesis, generative adversarial networks, GAN, stable GAN

1. INTRODUCTION

Recent developments have demonstrated notable achievements in object classification or tracking that arouse an interesting task – modeling real-world objects and generating perceptually plausible data. Focusing on image generation in this paper, variational auto-encoder [1] (VAE) outperformed concurrent reconstruction error based learning auto-encoders by adding a prior latent variable distribution restriction, commonly a Gaussian distribution with diagonal covariance matrix for simplification. Although VAEs could generate perceptually indistinguishable new images from training dataset, the strong prior regularization which avoids over-fitting problem may not follow actual latent distribution.

Among effective deep generative methods for this ambition, emergent generative adversarial networks (GANs) [2] outfits a more flexible and effective metric learning method for voluminous applications [3, 4, 5, 6] compared with commonly adapted Euclidean distance in original data space or non-linearly projected subspace. The adversarial training procedure consists of two simultaneous optimization objectives: 1) updating a discriminator network to tell apart synthesized data from real data; 2) updating a generator network to fool the discriminator as real.

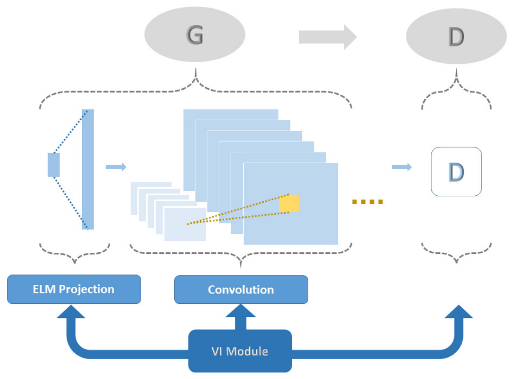

Fig. 1. Original generator G is divided into ELM projection and Convolution. The VI module is to control three learning rates for ELM projection, convolution and discriminator respectively. Input of VI module are the norm of gradient for each layer in G and D.

Ideally, this game process can reach a Nash equilibrium $p_g = p_{data}$ with gingerly designed network hyperparameters. Although following the theoretical analysis [2], there exists this global optimum. In practice, the networks converge to local optimum due to the adversarial target. Another well-known critical limitation is that GAN is very unstable to train, resulting in nonsensical output commonly.

A deep convolutional network architecture [3] was proposed for a detailed and applicable implementation that summarized several technical criteria: removing any pooling layers in generator and discriminator, using batch normalization [7] ahead of convolutional or fractional-convolutional layers, choosing ReLu activation for generator and LeakyReLu activation for discriminator. In this paper, our implementations and experiments follow these design principles.

For simple tasks, such as generating MNIST digits, re-implementation of DCGAN [3] is adequate for stable training. While for challenging tasks, the face generation, straightforward duplication results in common failure.

Laplacian pyramid networks [8] is to generate low-resolution image first and refine result step by stfep. The motivation is similar to residual networks which demonstrates benchmark of training very deep networks. A simplified meanwhile effective scheme is to boot GAN training by adding noise to discriminator, such as one-sided label smoothing [9, 10] or equivatlently instance noise [11] and continuous noise [12].

The shallow neural network involving in this paper, actually called extreme learning machines (ELMs), attracts our attention due to its capability of competing with DNN in some typical tasks just using one-single hidden layer. In addition, input projection parameters are randomly selected and keep unchanged. This phenomenon motivates us to bridge ELMs with GANs for a simplified networks structure and then obtain a improved performance.

Including Markov decision related methods in this paper derives from the demands to avoid struggling in tuning learning rates for a convergent training progress. Usually for classification problem, we adapt a linearly or exponentially decayed scheme to learning rate. Nevertheless, simple scheme can not be applied to GANs' domain directly. So we introduce value iteration related reinforcement learning algorithm to change learning rates dynamically.

2. PIPELINE METHODS

2.1. Generative Adversarial Networks (GANs) and Noise Injection

Define a normal prior distribution $p_z(\mathbf{z})$ as the directly sampled latent space, then the generator's mapping to data space is $G(\mathbf{z}, \boldsymbol{\theta_g})$, parametrized by $\boldsymbol{\theta_g}$. The scalar output of discriminator $D(\mathbf{x}, \boldsymbol{\theta_d})$ is scaled to $[0, 1]$, where $\boldsymbol{\theta_d}$ is the network parameter, representing the probability of $\boldsymbol{x}$ from p_{data} than p_g. Objectives of generator and discriminator are to minimize $Loss_g = log(1 - D(G(\mathbf{z}))$ and maximize $Loss_d = log(D(\boldsymbol{x})) + log(1 - D(G(\boldsymbol{z})))$ respectively. So the minimax game with value function $V(G, D)$ is as below:

$$\begin{aligned} min_G max_D V(G, D) = E_{\boldsymbol{x} \sim p_{data}(\boldsymbol{x})}[log(D(\boldsymbol{x})] \\ + E_{\boldsymbol{z} \sim p_z(\boldsymbol{z})}[log(1 - D(G\boldsymbol{z}))] \end{aligned} \quad (1)$$

In practice, analytic solution for Equation 1 is computationally prohibitive that we actually approximate the optimization problem by updating two objectives simultaneously. Typically for MNIST dataset, one-by-one updating would result in claimed results [2]. While in no experiments did we find applicable updating-step selection for CelebA [13] dataset by original GAN [2].

Actually, training criterion of generative networks could be re-formulated as minimizing Jensen-Shannon distance between p_{data} and p_g:

$$C(G) = 2JSD(p_{data}||p_g) - log4 \quad (2)$$

At early learning stage, G only generates nonsensical results (white noise) which could be perfectly identified from groundtruth. If p_{data} and p_g have disjoint supports Ψ, Ω respectively, then there is a optimal discriminator $D^{\star}(\boldsymbol{x})$ with accuracy 1 and $\bigtriangledown_x D^{*}(\boldsymbol{x}) = 0$ [12], gradient of Eq. 2 would not be sufficient. Observed in our experiments, log value of generator gradient norm can be lower than -6. The straightforward solution is to mix or filter p_{data}, p_g with noise to spread Ψ, Ω to increase the intersection $\Psi \cap \Omega$ manifold. Highly recommended noise types are listed below:

- ◇ Label smoothing or label noise [9, 10]: replacing original 0 and 1 outputs by smooth value such as .1 or .5.
- ◇ Instance noise [11]: applying a filter to images and feeding $x * \varepsilon$.
- ◇ Continuous noise [12]: mixing input x with noise ε as the input of discriminator $x + \varepsilon$.

Instance noise and continuous noise are conducted for the input of discriminator while label smoothing is for output. When the loss of discriminator converges to a tiny value, generator can still learn gradient by annealing noise level. We call this progress as 'booting GANs' since this trick only contributes to early training stage significantly and nothing to final optimum.

2.2. Extreme Learning Machines (ELMs)

Extreme learning machine [14] was proposed as a powerful generalized single-layer feedforward neural network, of which the weights connected input with hidden neurons are randomly generated and tune-free. Recent study [15] shows that ELMs can compete with deep neural networks using simpler nets architecture and less parameters. Single-layer ELM is formularized as $\boldsymbol{Y} = \boldsymbol{\beta}(g(\boldsymbol{WX} + \boldsymbol{B})))$. $\boldsymbol{W}, \boldsymbol{B}$ are randomly generated hidden weights. $g(\cdot)$ is the activation function, which usually is a Sigmoid function. $\boldsymbol{X}$, $\boldsymbol{Y}$ are input and output respectively. The only unknown parameter is β, which represents the fully connected weights to output. Analytic solution for ELM's objective function exists as below:

$$\boldsymbol{\beta} = \boldsymbol{Y}(g(\boldsymbol{WX} + \boldsymbol{B})))^{\dagger} \quad (3)$$

Where $(\cdot)^{\dagger}$ is a Moore-Penrose pseudoinverse function. The **key logic** of linking ELMs to robust GANs training is that: the linear projection from sampled z to high-dimensional vector h which is then reshaped to convolutional layer, is analogous to ELM random projection $g(\cdot)$: $ReLu(f(z)) \approx g(\boldsymbol{WX} + \boldsymbol{B})$. ELMs' hidden neuron size is the most elevated parameter that $\sim 10k$ can achieve higher rank than deep belief network and deep Boltzmann machine for MNIST classification task. In practice, dimension of h is about $\sim 35k$ in our generator fulfillment. Heuristic motivation is that, at the convergent stage of GANs, we can treat the movement of h as a 'escape' to neighboring 'safe' modes rather than 'explore' to unknown manifold. To bridge mutual connection, we decompose the generator into two parts: ELM projection and Convolution as presented in Figure 1.

Restore training by fixing the parameters of ELM projection based on a pre-trained GAN model, then we observe the improvement. This scheme is called GAN-E-layer for short.

2.3. Markov Decision Process (MDP)

Dynamical learning rate is not equivalent to updating generator or discriminator to alternately over-powered. A fundamental sequential model is Markov decision process (MDP) [16], which can be formularized as 4 tuples $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$. The progress $\mathcal{M}$ consists of actions set $\mathcal{A}$, states set $\mathcal{S}$, state transition function $\mathcal{P} = P(s'|s,a)$ and reward function $\mathcal{R} = R(s,a)$. $P(s'|s,a)$ prescribes the probability function given current state s and action a. $R(s,a)$ calculates the reward value based on observation of environment after making an action a.

Define the $value$ of arriving state s as $V(s)$, which is iteratively updated function as following:

$$V_{i+1} = max_a[R(s,a) + \gamma \sum_{s'} P(s'|s,a)V_i(s')] \quad (4)$$

Where $\gamma \in (0,1)$ is a discount factor. When $i \to \infty, V_i \to V^*$, actually we take a trade-off value k. Most significant contribution of value iteration networks (VIN) [17] is linking $P(s'|s,a)$ to convolutional layers that every kernel corresponds to transition value of one action a. Then the looping of Equation 4 is shifted to k recurrence of convolution.

The reason of involving VIN rather than other reinforcement learning methods [16, 18] is that VIN can learn faster with less training data and perform better long-term planning. This advantage is crucial to GANs' refinement as training and testing stages are unified in GANs' scenario. VI module must converge faster to act as director of tuning learning rates well.

As we expect a higher reward after long run, the MDP $\mathcal{M}$ would not feed reward back for every action. In implementation, VI module is updated for every 5 step planning following the reward scheme: $+1$ for reducing two losses together, -1 for increasing two losses meanwhile and 0.5 for alternative overwhelming to encourage faster optimization.

2.4. Unified Framework

This subsection is to group the former three elements into a overall framework to learn a DNN structure [19]. Firstly, explain essences of each above part shortly:

- ◇ Noise injection: booting GANs' training.
- ◇ ELMs: showing that learning rate of ELM projection should be low and different learning rates are essential.

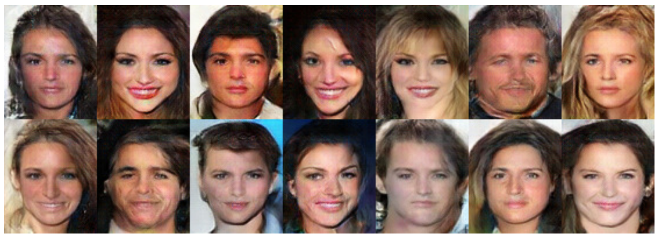
Fig. 2. Best scored results by pipeline in 5 batches.

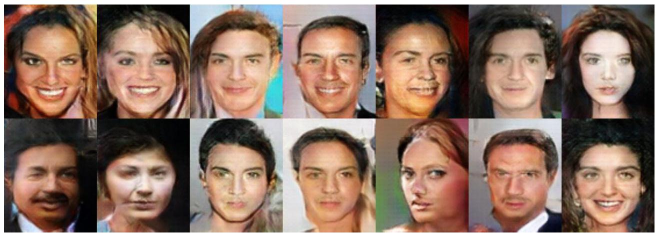
Fig. 3. Best scored results by GAN in 5 batches.

- ◇ VIN: observing learning loss and gradients to determine learning rate for each component and updating itself.

Among various noise injection methods, we compare label noise and simplified instance noise for rapid implementation. The former marks small proportion of one batch as fake data. The subsequent applies Gaussian filter to the input of discriminator to reduce the complexity of images and spread manifold supports. Our implementation shows that label noise may occasionally fail even for same label noise selection. Our simplified noise filter can boost training steadily for same annealing selection and doesn't require another auxiliary denosing auto-encoder to train generator.

To clarify details in VIN: states are quantified $loss$ value pair of generator and discriminator; observations are the index of gradient's norm of each layer in G and D following a predefined distribution division.

The unified framework abides the structure in Figure 1, of which norms of gradients to VI module are not presented.

3. EXPERIMENTS

The experimental scenario focuses on face synthesis (face area in CelebA [13] is cropped and re-sized to 80*98) rather than highly compared MNIST digits or Cifar dataset. MNIST is less challenging. And image quality evaluation for cifar is unconvincing that manual check is not applicable for most researchers and calculating class entropy $\boldsymbol{H(Y|X)}$ [9] with a pre-trained deep classifier can not be transferred to one-class generation tasks.

We highlight the peer-evaluation [20] between GAN and our unified training framework to show the effectiveness

Table 1. Peer-evaluation results

Player / Referee	GAN	GAN-E-layer	Pipeline
GAN	0.359	0.3865	**0.451**
GAN-E-layer	0.396	**0.4692**	0.362
Pipeline	0.363	0.3483	**0.4777**
Ave.	0.3727	0.4013	**0.4302**

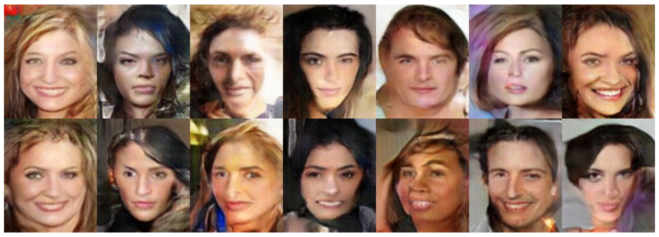

Fig. 4. Random selected results by pipeline in 5 batches.

across pipeline. Given two individually trained GAN models A and B, when discriminator of A acts as referee, the peer-evaluation score of generator of B is calculated as below:

$$C_B = \int_{z\sim p(\boldsymbol{z})} D_A(G_B(z))dz \tag{5}$$

Actually, in no experiments did we implement orginal GAN [2] successfully. Therefore baseline is achieved by applying noise injection as stated in section 2.1. Recall that we claim noise injection only work at early training stage and has no influence to final result. In comparison, only label noise and convolutional noise are simultaneously and randomly adapted. Peer-evaluation scores are listed in table 1 for GAN, GAN-E-layer and Pipeline. Training iterations of GAN is $10k$. GAN-E-layer and Pipeline both extend $5k$ iterations based on a $5k$-run pre-trained GAN. Batch sizes share the same 128.

3.1. GAN's Details

The generator network architecture follows DCGAN [3]'s design principles. Convolutional structure is B-K5-C1024-S2, B-K5-C512-S2, B-K5-C256-S2, B-K5-C128-S2 and B-K5-C3-S2. For clarification, for B-K5-C1024 represents batch normalization, kernel size, channel size and stride. The convolutional structure of discriminator network architecture is B-K4-C64-S2, B-K3-C128-S2, B-K3-C256-S2 and B-K3-C512-S2. Our code is modified based on InfoGAN [21].

3.2. VIN's Details

We implement the value iteration network following analogous design for grid-world search game [17]. The observation in that game is maps of target, holes and others. While for

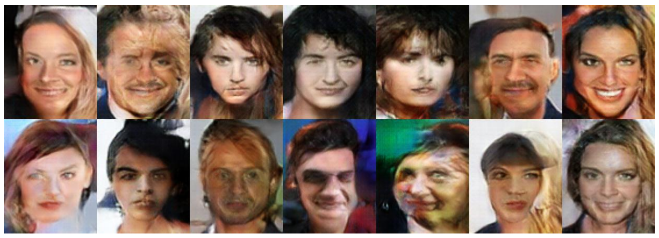

Fig. 5. Random selected results by GAN in 5 batches.

our problem, the observation is gradients' norm for each layer with the assumption that can help to make better update decision. The states are quantified loss value pair $(loss_G, loss_D)$.

In practice there are 15 bins for each loss from 0.7 to 1.5. The lower bound is set to 0.7 due to the optimal derivation $loss^*_{GorD} \approx 0.69$ and the upper bound is 1.5 for the reason that $loss$ value is log value.

We concatenate the one-hot quantification vector of log value of each gradient norm. The final vector is non-linearly projected into a longer vector, which can be reshaped to fit the following convolution. The $value\ \mathcal{V}$ is parameterized by convolutional layers K3-C40-S1, K1-C1-S1 and K3-C9-S1. The former two operations correspond to $reward\ \mathcal{R}$.

The actions $\mathcal{A}$ is set to $[D-0.001, D-0.0005, D-0.0001, G-C-0.001-E-0.0001, G-C-0.0005-E-0.0001, G-C-0.0001-E-0.0001, GC-0.001-E-0, G-C-0.0005-E-0, G-C-0.0001-E-0]$, among which $G-C-0.001-E-0.0001$ represents updating convolutional layers with learning rate 0.001 and ELM projection with 0.0001 in G.

4. CONCLUSION

In this paper, we study cross-domain and comprehensive related works for stable and improved GANs' training. A pipeline is summarized to fit extensive adaptions. Quantitative evaluations show its effectiveness and significance. Selected outputs of pipeline training with high self-evaluation score are presented in figure 3. Although there are no perceptually repeated observations [22], we find more high-quality female faces than male for each training strategy. This missing modes problem would be covered during the following research.

5. ACKNOWLEDGEMENT

This work was conducted within the Rolls-Royce@NTU Corporate Lab with support from the National Research Foundation (NRF) Singapore under the Corp Lab@University Scheme.

6. REFERENCES

[1] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[2] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, pp. 2672–2680, 2014.

[3] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[4] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," *arXiv preprint arXiv:1609.04802*, 2016.

[5] X. Yan, J. Yang, K. Sohn, and H. Lee, "Attribute2image: Conditional image generation from visual attributes," in *European Conference on Computer Vision*, pp. 776–791, Springer, 2016.

[6] S. Reed, Z. Akata, X. Yan, L. Logeswaran, B. Schiele, and H. Lee, "Generative adversarial text to image synthesis," in *Proceedings of The 33rd International Conference on Machine Learning*, vol. 3, 2016.

[7] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[8] E. L. Denton, S. Chintala, R. Fergus, *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Advances in neural information processing systems*, pp. 1486–1494, 2015.

[9] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, pp. 2226–2234, 2016.

[10] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2818–2826, 2016.

[11] C. K. Sønderby, J. Caballero, L. Theis, W. Shi, and F. Huszár, "Amortised map inference for image super-resolution," *arXiv preprint arXiv:1610.04490*, 2016.

[12] M. Arjovsky and L. Bottou, "Towards principled methods for training generative adversarial networks," in *NIPS 2016 Workshop on Adversarial Training. In review for ICLR*, vol. 2016, 2017.

[13] Z. Liu, P. Luo, X. Wang, and X. Tang, "Deep learning face attributes in the wild," in *Proceedings of International Conference on Computer Vision (ICCV)*, 2015.

[14] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, "Extreme learning machine: theory and applications," *Neurocomputing*, vol. 70, no. 1, pp. 489–501, 2006.

[15] J. Tang, C. Deng, and G.-B. Huang, "Extreme learning machine for multilayer perceptron," *IEEE transactions on neural networks and learning systems*, vol. 27, no. 4, pp. 809–821, 2016.

[16] D. P. Bertsekas, D. P. Bertsekas, D. P. Bertsekas, and D. P. Bertsekas, *Dynamic programming and optimal control*, vol. 1. Athena Scientific Belmont, MA, 1995.

[17] A. Tamar, S. Levine, P. Abbeel, Y. WU, and G. Thomas, "Value iteration networks," in *Advances in Neural Information Processing Systems*, pp. 2146–2154, 2016.

[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[19] B. Zoph and Q. V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016.

[20] D. J. Im, C. D. Kim, H. Jiang, and R. Memisevic, "Generating images with recurrent adversarial networks," *arXiv preprint arXiv:1602.05110*, 2016.

[21] X. Chen, Y. Duan, R. Houthooft, J. Schulman, I. Sutskever, and P. Abbeel, "Infogan: Interpretable representation learning by information maximizing generative adversarial nets," in *Advances in Neural Information Processing Systems*, pp. 2172–2180, 2016.

[22] T. Che, Y. Li, A. P. Jacob, Y. Bengio, and W. Li, "Mode regularized generative adversarial networks," *arXiv preprint arXiv:1612.02136*, 2016.