EECE 5690 - Project Report
Machine Learning for the Generation of Artificial Dog Images

David Helminiak[1], Najibakram Maheenaboobacker[1], and Scott Stewart[1]
Marquette University, EECE 5690, Dong Hye Ye[1]

I. Abstract

Advancements in deep learning have led to the widespread adoption and implementation of Generative Adversarial Networks (GANs). These models, based on an inversion of the basic autoencoder structure, are used in order to perform improved image up-sampling, dataset augmentation, physics' simulations, and advanced image manipulation. The highly realistic results have become a noteworthy problem in the form of deep fakes, where traditional metrics can no longer distinguish reality from falsified data. Thereby, a new race has developed between the generation of even more realistic fakes and techniques to prevent malicious use of machine learning. Herein, a Convolutional Variational AutoEncoder (CVAE) and Wasserstein GAN with gradient penalty (WGAN-GP) with novel contributions, generate fake dog images, and are evaluated with an Inception-v3 network.

II. Introduction

There exists a growing issue with the generation of perceptually indistinguishable images, necessitating the development of methods for their detection [1]. In order to encourage development, Kaggle hosted a competition in 2019 to design models, which can create fake dog images [2]. Most submissions were based on the Generative Adversarial Network (GAN) [3], with the best results produced using the recently developed BigGAN architecture. Runner-ups included StyleGAN [4] and ProGAN [5] models; the former being noteworthy for not creating images from scratch, but instead transforms images between latent spaces. While the competition results were fairly realistic, the competition limitations, such as only allowing 9 hours for training, deprecated the potential of slower architectures, such as Wasserstein GAN (WGAN). As use of modified datasets was disallowed, there was a general reliance on random cropping, meaning the results were often a matter of luck.

This work aimed to improve on the competition results, without a constraint on training time and with dataset modifications. Based on the GAN [3], Deep Convolutional GAN (DCGAN), and autoencoder [6], this work builds a Wasserstein GAN with gradient penalty (WGAN-GP) and a Convolutional Variational AutoEncoder (CVAE). An Inception-v3 classification model was used to evaluate them.

[1]Electrical, Electronics and Computer Engineering at Marquette University. Milwaukee, WI.

A. Literature Review

1) CVAE: CVAEs are a foundational architecture in the consideration of design for generative image networks. As demonstrated by Bao et al. in 2017 [7], CVAEs can avoid the typical issues associated with GANs and deep neural networks, namely mode collapse and vanishing gradients. However, it is difficult to encode them with small details.

2) GAN: The GAN was proposed in 2014 by Goodfellow et al. [8] and was improved upon with the DCGAN in 2015 by Radford et al. [9], replacing pooling with strided convolutions for higher dimensions. A notable variation, the SRGAN by Ledig et al. [10], follows each upscaling convolutional transpose block in the generator with a secondary block to improve image fidelity. The DCGAN was also advanced by Arjovsky et al. to form WGAN in early 2015 [11], employing the earth mover's distance in the loss function to improve stability.

$$\frac{|D(x_1) - D(x_2)|}{|x_1 - x_2|} \leq 1 \tag{1}$$

$$\mathop{\mathbb{E}}_{\tilde{\mathbf{x}}\sim\mathbb{P}_g}[D(\tilde{\mathbf{x}})] - \mathop{\mathbb{E}}_{\tilde{\mathbf{x}}\sim\mathbb{P}_r}[D(\mathbf{x})]\} + \lambda \mathop{\mathbb{E}}_{\hat{\mathbf{x}}\sim\mathbb{P}_{\hat{\mathbf{x}}}}[(\|\hat{x}D(\hat{\mathbf{x}})\|_2 - 1)^2] \tag{2}$$

To better conform with the 1-Lipschitz continuity inequality, (Equation 1), Gulrajani et al., later in 2015 [12], proposed the WGAN-GP architecture, replacing clipping with a gradient penalty. This resulted in the discriminator loss described by Equation 2, where λ adjusts the penalty's strength. Some of the best practices have ultimately been combined by the 2019 work for BigGAN by Brock et al. [13], which uses truncated normal noise for the generator input, to control trade-off between resulting image fidelity and variation.

Finally, while GANs typically employ either Adam [14], or RMS propagation [15], a development in 2016: Nadam optimization [16] by Dozat et al., adds Nesterov momentum to Adam, which can lead to faster and smoother convergence.

3) Inception-v3: The inception deep convolution architecture (Inception-v1 or GoogLeNet) was introduced in 2015 [17]. This was refined as Inception-v2, employing batch normalization [18], and as Inception-v3 adding factorizing convolution to improve the relative network performance/efficiency [19]. With 42 convolutional layers, the computation cost is only $\sim$ 2.5 higher than that of the original, with a reduction of $\sim 25\%$ in classification error.

B. Novelty

Novel to this study, a re-cyclical CVAE is created, feeding outputs back through the network multiple iterations to

improve quality. The CVAE also uses latent spaces, extracted with K-Means, for image transformation. The DCGAN architecture employs secondary convolutional blocks for improved quality, as does the determined WGAN-GP architecture. This WGAN-GP also replaces convolutional transposition layers with upsampling to avoid checkerboard artifacts, incorporates layer normalization as a drop-in replacement for batch normalization in the generator, uses Nadam optimization, and truncated normal noise for generator input.

III. Dataset Description

The dataset being used is the Stanford Dog Dataset, as extracted from ImageNet [20]. The set contains 120 breeds with $20,579$ images, with an average 173 ± 23 images/breed and sizes of $386\text{x}443 \pm 134$ pixels. All images were manually cropped to a 1:1 aspect ratio, focusing only on a single dog subject wherever possible; matching the chosen dog against the included annotations. All watermarks and as much text as possible were removed. If the dog could not fit within the chosen window size, focus was placed on incorporating its head. All crops under 400x400 were enlarged as needed. Several resulting images are visualized in Figure 5.

IV. Methods

A. GAN

The GAN models were constructed with Python v3.7.7 and TensorFlow 2.1.0 with 1-4 1080TI NVIDIA GPUs. The initial DCGAN architecture used batch normalization (except for the discriminator input and generator output), fully connected input/output layers, tanh activation for the generator's final layer, and sigmoid for the discriminator's, with leaky ReLU for all other activations to account for vanishing gradients. All convolutions were performed with 4x4 kernels with binary cross entropy used for loss, and a batch size of 128. Nadam optimization was used with initial learning rates of 0.0002 and 0.0005, for the generator and discriminator respectively, as well as Beta1/2 values of 0.5 and 0.999. The discriminator used dropout at a rate of 0.3, with additional one-sided label smoothing (at a 20% rate), as well as adding random normal noise to the real images during each iteration (standard deviation of 0.1). These choices were made to better avoid mode collapse, improve training time, and align the progression of the networks' convergence. Weights were initialized with a truncated normal distribution (0 mean and standard deviation of 0.02). Following SRGAN, secondary convolutional blocks (stride of 1) in the generator were confirmed to provide notably improved fidelity with minimal computational overhead. The best perceptual images were found at $\sim$ 200 epochs, where initial evaluation using the Inception-v3 model (Section V) showed them as equivalent to from-scratch CVAE images.

The DCGAN was transformed into a WGAN-GP, using the earth mover's metric for loss along with a gradient penalty, with λ=10. Layer normalization was added to the generator, with no normalization used in the discriminator. Nearest-neighbor upscaling was used as a drop-in replacement for convolutional transposition to avoid checkerboard artifacts. The weight initialization was modified to correspond to the number of input units, the with He-et-al method [21]. Since the discriminator network is known to train much faster than the generator's, asynchronous learning rates were initially employed, however the practice of training the generator for 5 steps for each of the discriminator's (and lowering Beta2 to 0.9) was found to better reduce convergence noise and improve stability.

Initial optimization included the following studies: adding 1) layers at the end of the generator, decreasing in power of 2; 2) dropout (rates of 0.1 and 0.3) to each convolutional layer of the discriminator; 3) layer normalization at each convolutional layer of the discriminator; and 4) truncated random normal noise input to the generator (after training). The networks were run for 200 epochs, with visual comparison between results made with 100 images from each. Thereby observed, adding smaller convolutional layers to the generator's end significantly harmed the resultant quality. Dropout and layer normalization in the discriminator had no visible impact on the final results, but doubling the convolutional layers in the generator allowed for faster convergence. The injection of truncated random normal noise into the generator also improved the Inception-v3 evaluation.

Two final runs of the WGAN-GP architecture (Figure 1) were run for $10,000$ and $5,000$ epochs with respective batch sizes of 32 and 8. The final losses are visualized for each in Figure 2 with example images shown in Figure 5. These runs performed training across 4 GPUs, which improved speed from $\sim$ 1:50 minutes/epoch to $\sim$ 20 seconds/epoch.

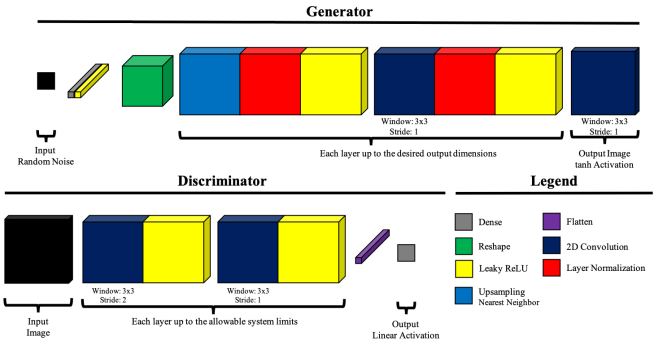

Fig. 1. Finalized architecture of the WGAN-GP network

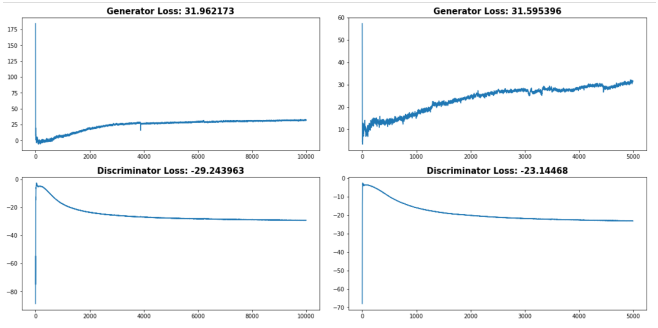

Fig. 2. WGAN-GP training losses after (Left) $10,000$ epochs with batch size of 32 and (Right) losses after $5,000$ epochs with batch size of 8

B. CVAE

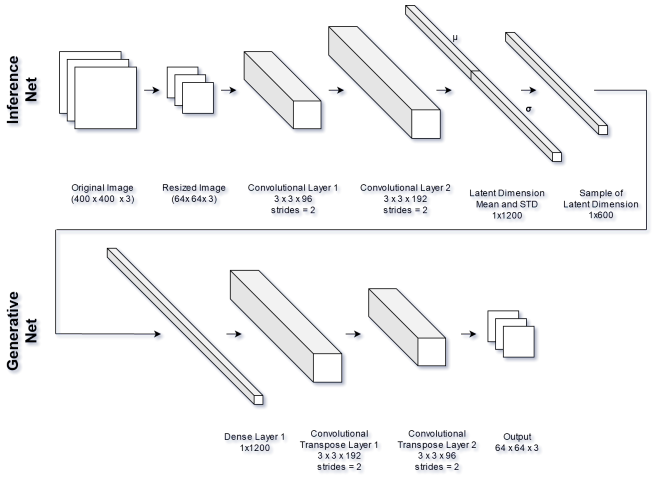

Fig. 3. The CVAE network architecture

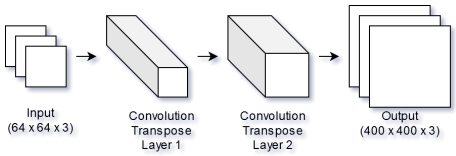

Fig. 4. The upscaling network architecture

The CVAE architecture (Figure 3) was designed to encode the latent dimensions of the training dataset images. This was combined with an upscaling network (Figure 4) to encode even finer details. Diminishing returns with increased training data was observed over development, such that only half of the preprocessed data was employed.

After training both the CVAE and upscaling networks, latent dimensions were extracted from the real images and run through K-Means, where k=1000. The mean and standard deviation for every k, where there was more than one image, were fed into the CVAE and subsequently, the upscaling network. The resulting image was "recycled" through these models until convergence (3 times). An alternative approach, transforming existing images from their existent latent spaces was also employed. After feeding a random dataset image through the CVAE's inference network, random noise was added to the latent space, then passed through the generative net, producing higher quality images.

V. Evaluation

An Inception-v3 classification network evaluated each models' generation capabilities. Transfer learning (weights from ImageNet) reduced the time requirement. Training consisted of 1) CNN (Convolutional Neural Network) feature extraction and 2) classification with fully connected and softmax layers. A subset of $8,743$ images was used to train, with a train/test/validation split of 80%/10%/10%, reaching accuracies of 96.2%, 85.3%, and 81.5% after 50 epochs.

Additional architectures, preceding Inception-v3, were explored with and without transfer learning (summarized in Table I). Inception-v3 versions A and B differed in using ReLU and Leaky ReLU activations respectively.

TABLE I
Summary of Classification Models

	Train Acc. (%)	Val Acc. (%)	Test Acc.(%)	Train Loss	Val Loss
VGG16 w/o TL	9.5	7.5	9.3	3.9	4.2
VGG16 w TL	71.2	53.6	56.6	4.6	5.9
Incep.v3A w TL	71.5	32.5	67.4	2.5	1.6
Incep.v3B w TL	96.2	85.3	81.5	0.25	0.8

VI. Results

The networks' (CVAE and WGAN-GP) effectiveness was tested by passing 200 outputs from both through the Inception-v3 network. 5 random images from each models' output and from the top Kaggle result are shown in Figure 5. Confidence levels for each were plotted in Figure 6, and show that the current WGAN-GP network leads the CVAE by 16% in producing convincing fake images. There still remains significant room for improvement for both models, with a relatively poor maximum average confidence of 39%. However, by examining the top 5 frequently occurring classes seen in Table II (classified by the inception model), the varied distribution in breed evidences the lack of mode collapse for both CVAE and WGAN-GP results. It may also be noted that the WGAN-GP training was still stable and converging at the time its final results were produced.

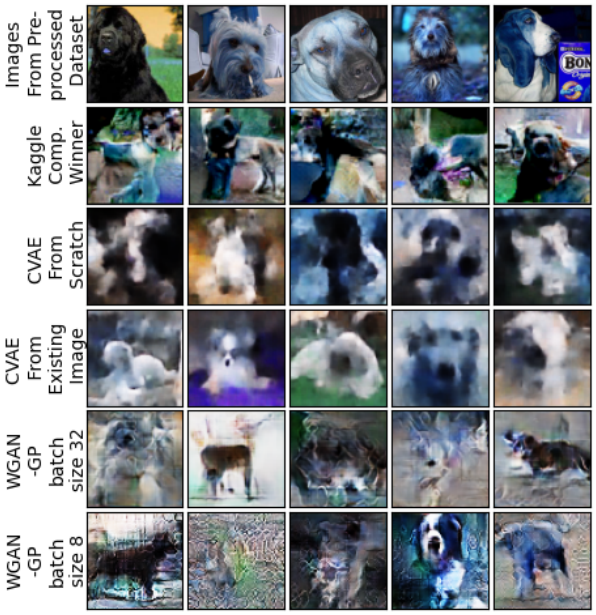

Fig. 5. Randomly selected images from the training dataset, final model outputs, and the best Kaggle model [2]

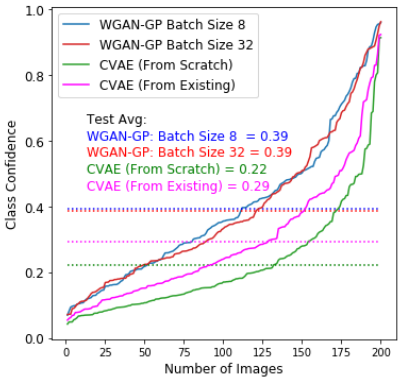

Fig. 6. Confidence that WGAN-GP and CVAE outputs contain a dog

TABLE II

PERCENT DISTRIBUTION OF THE 5 MOST FREQUENT DOG BREEDS

Model	Icelandic Sheepdog	Brussels Griffon	German Wire -haired Pointer	Dach-shund	Australian Shepherd	Other
WGAN -GP Batch Size 32	15.0	9.5	2.0	11.5	0.45	57.5
WGAN -GP Batch Size 8	14.5	14.5	0.6	11.0	0.4	50.0
CVAE (From Scratch)	18.5	10.5	9.5	1.0	6.0	54.5
CVAE (From Existing)	18.0	5.5	14.5	4	7.5	50.5

VII. CONCLUSION

Although, progress in generating fake images has been rapidly accelerating, its still a difficulty to distinguish the right architectural direction for the machine learning community to commit to for future development. For example, the computationally simpler CVAE model was seen to match with early GAN results and is comparable with the WGAN-GP outputs. Further, while the WGAN-GP architecture has a bottleneck in its slow convergence time, it remains attractive for its inherent robustness; a feature somewhat less present in more recent architectures. In order to improve upon the WGAN-GP's limitation, future work with the current architecture should be performed to accelerate the generator after initial discriminator convergence and examine the effects of adjusting the gradient weighting multiplier. It may be preferable to employ spectral normalization [22] which mitigates the computational cost of the gradient penalty, or implement the strategy of a Conditional GAN (CGAN) [23] to use image labels; customizing weight normalization on a class-by-class basis. For an improved evaluation of the models presented, the Inception-v3 network would benefit from a larger and more class-balanced training set. Additional metrics, such as the Inception Score (IS), or Fréchet Inception Distance (FID) would also help in performing a comprehensive and effective evaluation.

SUPPORTING INFORMATION

Code for this project has been made available at: https://github.com/Yatagarasu50469/Marquette-Course-Projects/tree/master/dogImageGeneration-EECE5690

ACKNOWLEDGEMENT

The authors thank Stanford University for making their dog dataset available and Dr. Ye for the research opportunity.

REFERENCES

[1] F. Marra, D. Gragnaniello, D. Cozzolino, and L. Verdoliva. Detection of gan-generated fake images over social networks. In *2018 IEEE Conference on Multimedia Information Processing and Retrieval*, 2018.

[2] Kaggle. Generative dog images, Aug 2019.

[3] V. Dumoulin, K. Arulkumaran, B. Sengupta, and A.A. Bharath. Generative adversarial networks: An overview. In *IEEE-SPM*, April 2017.

[4] T. Karras, S. Laine, and T. Aila. A style-based generator architecture for generative adversarial networks, 2018.

[5] T. Karras, T. Aila, S. Laine, and J. Lehtinen. Progressive growing of gans for improved quality, stability, and variation, 2017.

[6] M. Stewart. Comprehensive introduction to autoencoders, April 2019.

[7] J. Bao, D. Chen, F. Wen, H. Li, and G. Hua. CVAE-GAN: fine-grained image generation through asymmetric training. *CoRR*, abs/1703.10155, 2017.

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

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

[10] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi. Photo-realistic single image super-resolution using a generative adversarial network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 105–114, 2017.

[11] M. Arjovsky, S. Chintala, and L. Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223, International Convention Centre, Sydney, Australia, 06–11 Aug 2017. PMLR.

[12] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville. Improved training of wasserstein gans, 2017.

[13] A. Brock, J. Donahue, and K. Simonyan. Large scale gan training for high fidelity natural image synthesis, 2018.

[14] D.P. Kingma and J. Ba. Adam: A method for stochastic optimization, 2014.

[15] T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude, 2012.

[16] T. Dozat. Incorporating nesterov momentum into. In *ICLRWorkshop*, 2015.

[17] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

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

[19] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision, 2015.

[20] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Novel dataset for fine-grained image categorization. In *First Workshop on Fine-Grained Visual Categorization, IEEE Conference on Computer Vision and Pattern Recognition*, Colorado Springs, CO, June 2011.

[21] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification, 2015.

[22] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida. Spectral normalization for generative adversarial networks, 2018.

[23] M. Mirza and S. Osindero. Conditional generative adversarial nets, 2014.