A comparative evaluation on the generative ability of DCGANs

Jonna Marie Matthiesen
gusmatjoak@student.gu.se

Kinga Jenei
gusjeneki@student.gu.se

***Abstract*—This paper studies the application of deep convolutional generative adversarial networks in the field of unsupervised representation learning. A focus will be on the visual evaluation of the generative performance given three different image datasets. We will show that the success of our generative model highly depends on the diversity and size of the dataset.**

***Index Terms*—DCGAN, GAN, Deep Learning, generative model, representation learning**

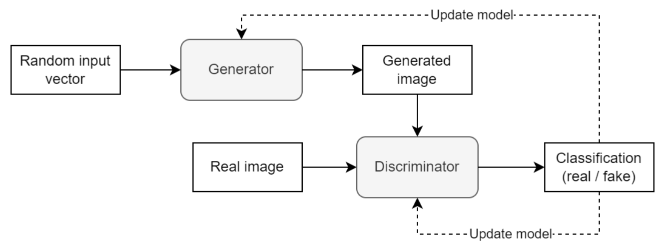


Fig. 1. GAN architecture. Own elaboration.

I. Introduction

In this project we investigate the structure and behaviour of Generative Adversarial Networks (GANs) and build our own network to create new images for different datasets. GANs are widely used for a range of different purposes, for example, image and music generation [1], image to image translation [2], text to image translation [3], and photography blending [4]. Generating synthetic images is highly useful since the amount of data is often not adequate for training a deep learning model. Enhancing the dataset by creating new images with GANs can yield better model performance and is used for example in the medical field [1, 5, 6].

In this project we will study and implement a specific type of GAN, called Deep Convolutional Generative Adversarial Network (DCGAN). DCGANs have been created specifically for the task of image generation and allow for a more stable training.

II. Background theory

A. Generative Adversarial Networks

GANs have been first introduced by Goodfellow et al. (2014). These models perform an unsupervised learning task by learning patterns in the training data and generating new images based on their findings. They consist of two separate networks, one generator G and one discriminator D, which are trained together. The generator is trained to create new examples that look like the training images, whereas the discriminator is trying to classify them as either real or fake. Based on the classification results both the discriminator and the generator are updated, see figure 1.

Let p_{data} be the distribution of the training dataset and let z be a random input vector sampled from the multivariate normal distribution, $z \sim \mathcal{N}(0, \mathbb{1})$. $G(z) = x$ describes the generated "fake" image created by the generator capturing the learned distribution of the data. It is $x \sim p_{gen}$, where p_{gen} describes the distribution of the images generated by the generator G. $D(x) = D(G(z))$ is the probability that x is a real image.

As described in [7] the counter-play of the generator and discriminator can be described as a two-player mini-max game: The discriminator D tries to minimize the probability $D(G(z))$ of the fake image $D(z)$ being classified as real image sampled from p_{data} while maximizing the probability of classifying real images as being real. The generator G tries to create an image x maximizing $D(x)$. In other words, G tries to minimize $\log(1-D(G(z))$ while D tries to maximize $\log D(s)+\log(1-D(G(z))$, where s is a real image sampled from p_{data}. We have the following mini-max game:

$$\min_G \max_D \mathbb{E}_{s\sim p_{data}}[\log D(s)] + \mathbb{E}_{z\sim\mathcal{N}(0,\mathbb{1})}[\log(1 - D(G(z))]. \tag{1}$$

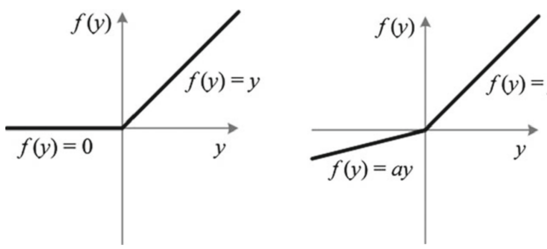
The optimal solution to this mini-max game is when $p_{data} = p_{gen}$, where the discriminator is unable to distinguish real images from fake ones,

$$\mathbb{E}_{x\sim p_{data}} \log D(x) = \mathbb{E}_{z\sim\mathcal{N}(0,\mathbb{1})} \log D(G(z)).$$

B. Deep Convolutional Generative Adversarial Networks

A standardized approach to GANs called Deep Convolutional Generative Adversarial Networks (DCGANs) was introduced by Radford et al. (2015) which led to more stable models. Different to GANs, DCGANs explicitly use convolutional and convolutional-transpose layers in the discriminator and generator, respectively. Other changes include elimination of the fully connected layers, usage of batch normalization and the Adam optimizer, ReLU and Tanh activation functions in the generator, and LeakyReLU and Sigmoid activations in the discriminator.

The activation function LeakyReLU is based on the ReLU activation function with the difference of having a small slope for negative values instead of a flat slope [9], see figure 2. Using LeakyReLU instead of the classical ReLU as an activation function solves the vanishing gradient problem for negative values [10].



(a) ReLU (b) LeakyReLU

Fig. 2. Comparison of the ReLU and LeakyReLU activation functions. The parameter α of the LeakyReLU function is an additional parameter, chosen to be 0.2 in [8] and our own generative network. [9]

III. Method / proposed solution

A. Data

The data used for this project consists of 3 different datasets. The first one contains images of 450 different bird species resulting in 95.376 images [11]. Each image is in jpg format with a 224x224x3 size and shows one bird taking up at least 50% of the pixels in the image. The second dataset consists of 16.130 colored images of animal faces in three different classes (cat, dog, wildlife) [12]. The images have a resolution of 512×512 and are in jpg format. The third dataset contains 87.900 jpg images of healthy and diseased crop leaves categorized into 38 different classes [13].

B. Model architecture and training

Given the architecture guidelines described in [8], stable deep convolutional GANs are made of a combinations of convolutional layers, batchnorm layers, and activation functions. For the generator we are using ReLU activation functions with an exception in the last layer using a Tanh activation function. The discriminator is using LeakyReLU activation functions and a Sigmoid activation in the last layer. As shown in figure 3 both networks, generator and discriminator, consist of five convolutional layers.

Our implementation of a DCGAN to generate new images given a dataset is based on [15]. In addition to the guideline in [8] and the implementation in [15] we have added Dropout layers as regularization to the discriminator's network to prevent overfitting in the predictions and reduce variance.

We preprocess the training data of each dataset by cropping each image to the size 64x64x3. Since the generator's output

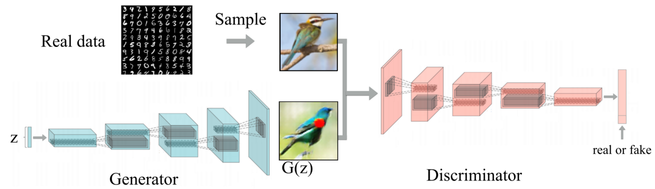


Fig. 3. DCGAN generator and discriminator architecture. [11, 14]

pixel values fall in the range [-1, 1] due to the tanh activation function in the generator's last layer, we normalize each image so that its pixel values fall within that range.

C. Model evaluation

Training a generative adversarial network consists of training two distinct neural networks to maintain the above mentioned equilibrium, see equation 1. As such, there is no objective loss function the GAN model seeks to minimize. Due to the lack of an objective way to evaluate the performance of the model, we resort to qualitative and quantitative measurements when assessing the performance of our GAN.

Quantitative metrics are most often used to evaluate GANs during training. They evaluate fidelity, the quality of images, and diversity, the variety of images [16]. While different metrics for evaluating GANs give a good impression on the overall performance of the model, they lack the ability to measure the authenticity of the model. A truly generative model should not duplicate training images and instead create new images as being sampled from the unknown distribution of the training data p_{data}. The question of when a GAN overfits and if it is truly generative is a open question in the field of unsupervised representative learning using generative models [17, 18].

To evaluate the performance of our model and prevent visual overfitting, we manually inspect and judge generated images during training at different iteration steps. By doing inspections of the fake images as the model learns to imitate the training distribution p_{data} we can stop training where the images start to look similar to the samples in the training dataset.

IV. Results / discussion

As mentioned previously we have chosen to evaluate the performance of our model by manually inspecting generated (fake) images given the real images from one of the three datasets. As shown in figure 4 generated leaf images show a great similarity to the given dataset of real leaf images. The generated fake birds, see figure 5, are harder to interpret as being birds but some good generated fakes in the top right and middle of the grid resemble birds quite accurately. Lastly, the dataset consisting of a mixture of dog, cat, and wildlife animal faces appears to be the most difficult for the proposed generative network given the results in figure 6.

Given the obtained result, the difficulty of the task of generating new images from an unknown distribution present in a dataset seems to be highly dependent on the dataset. While the bird and leaf image datasets contain more than 80.000 images each, the fake images generated from the leaf dataset seem to be closer to the original training data and can be easily mistaken for being real. Comparing the fake and real bird images as shown in figure 5 the performance of the generative model is not as good.

One imaginable reason is the difference in the complexity of the training dataset, meaning the diversity of training samples. The training images of the leaf dataset are similar in the sense that each sample shows a single leaf on a gray background. For the bird dataset, we also have only one bird present taking up at least 50% of the image's space but with different backgrounds, angles and motions, e.g. flying and standing.

For the third dataset we obtained the worst results in terms of quality and diversity of generated images. This dataset is not only very diverse, meaning faces from different animal species and different angles, but also much smaller with roughly 16.000 training images.

(a) Real leaves (b) Fake leaves

Fig. 4. Comparison of real and generated (fake) leaf images. Dataset: [11]; Own elaboration.

(a) Real birds (b) Fake birds

Fig. 5. Comparison of real and generated (fake) bird images. Dataset: [11]; Own elaboration.

(a) Real faces (b) Fake faces

Fig. 6. Comparison of real and generated (fake) animal faces. Dataset: [12]; Own elaboration.

V. Conclusions

The aim of this project was to study the structure and behaviour of GANs and build a DCGAN network to create new images for different datasets. In this work, three different datasets with images of plant leaves, birds, and animal faces are used as experimental data to evaluate the quality of the generated images. The results show that the success of the generative model is highly dependent on the size and complexity of the dataset with the leaf dataset being the easiest and the animal face dataset the hardest to generate new images from. We conclude that by successfully generating new images from training images using a deep convolutional generative adversarial network variable results can be obtained depending on the used training dataset.

In order to obtain better results, the architecture of the network could be further improved by studying more recent research. One improvement, originally introduced by Mirza and Osindero (2014), is the use of class labels in the model, allowing the output of the generator to be explicitly controlled. This improvement also increases the ability of using generative models to increase the size of a labeled dataset. In the future, backpropagation could also be investigated for the purpose of finding the inputs which produce a particular output image.

References

[1] C. Han, H. Hayashi, L. Rundo, R. Araki, W. Shimoda, S. Muramatsu, Y. Furukawa, G. Mauri, and H. Nakayama, "Gan-based synthetic brain mr image generation," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE, 2018, pp. 734–738.

[2] Z. Yi, H. Zhang, P. Tan, and M. Gong, "Dualgan: Unsupervised dual learning for image-to-image translation," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2849–2857.

[3] W. Liao, K. Hu, M. Y. Yang, and B. Rosenhahn, "Text to image generation with semantic-spatial aware gan," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 18 187–18 196.

[4] H. Wu, S. Zheng, J. Zhang, and K. Huang, "Gp-gan: Towards realistic high-resolution image blending," in *Proceedings of the 27th ACM international conference on multimedia*, 2019, pp. 2487–2495.

[5] T. Iqbal and H. Ali, "Generative adversarial network for medical images (mi-gan)," *Journal of medical systems*, vol. 42, no. 11, pp. 1–11, 2018.

[6] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Synthetic data augmentation using gan for improved liver lesion classification," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE, 2018, pp. 289–293.

[7] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial networks," 2014. [Online]. Available: https://arxiv.org/abs/1406.2661

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

[9] A. K. Dubey and V. Jain, "Comparative study of convolution neural network's relu and leaky-relu activation functions," in *Applications of Computing, Automation and Wireless Systems in Electrical Engineering*. Springer, 2019, pp. 873–880.

[10] A. Maniatopoulos and N. Mitianoudis, "Learnable leaky relu (lelelu): An alternative accuracy-optimized activation function," *Information*, vol. 12, no. 12, p. 513, 2021.

[11] "Birds 450 species," https://www.kaggle.com/datasets/gpiosenka/100-bird-species, Accessed: October 24, 2022.

[12] "Animal faces," https://www.kaggle.com/datasets/andrewmvd/animal-faces, Accessed: October 24, 2022.

[13] "New plant diseases dataset," https://www.kaggle.com/datasets/vipoooool/new-plant-diseases-dataset, Accessed: October 24, 2022.

[14] U. Haputhanthri, "Introduction to deep convolutional generative adversarial networks using pytorch," May 2020. [Online]. Available: https://medium.com/the-ai-team/introduction-to-deep-convolutional-generative-adversarial-networks-using-pytorch-92d88a19a574

[15] N. Inkawhich, "Dcgan tutorial," https://pytorch.org/tutorials/beginner/dcgan_faces_tutorial, Accessed: October 24, 2022.

[16] A. Borji, "Pros and cons of gan evaluation measures," *Computer Vision and Image Understanding*, vol. 179, pp. 41–65, 2019.

[17] C. Meehan, K. Chaudhuri, and S. Dasgupta, "A non-parametric test to detect data-copying in generative models," in *International Conference on Artificial Intelligence and Statistics*, 2020.

[18] A. Alaa, B. Van Breugel, E. S. Saveliev, and M. van der Schaar, "How faithful is your synthetic data? sample-level metrics for evaluating and auditing generative models," in *International Conference on Machine Learning*. PMLR, 2022, pp. 290–306.

[19] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.