Data Augmentation of Grape Leaf Images using DCGAN

Pratyush Kumar
Electronics and Electrical Engineering
Indian Institute of Technology
Guwahati, Assam
pratyush.kumar@iitg.ac.in

Vasudev Krishna Kottra Chal
Electronics and Electrical Engineering
Indian Institute of Technology
Guwahati, Assam
c.vasudev@iitg.ac.in

Jigyansu Dash
Biosciences and Bioengineering Department
Indian Institute of Technology)
Guwahati, Assam
d.jigyansu@iitg.ac.in

***Abstract*—Deep learning models for leaf disease identification often suffer from limited training data, which leads to reduced performance. This paper explores the use of a Deep Convolutional Generative Adversarial Network (DCGAN) for data augmentation of grape leaf images to address this challenge. A DCGAN model, based on the original DCGAN paper [1], was trained on a dataset of 4,062 authentic grape leaf images [4][7] to generate 16 synthetic samples. The quality of these generated images was assessed using the Fréchet Inception Distance (FID) score [3]. The impact of this limited data augmentation on classification accuracy was evaluated using a pre-trained Inception V3 model from PyTorch. The paper aims to highlight the results of DCGAN-based data augmentation for grape leaf disease identification, even with a limited number of generated images. The findings contribute to understanding the feasibility and effectiveness of this approach for improving classification accuracy in scenarios with constrained resources.**

***Index Terms*—**

I. Introduction

The agricultural sector faces significant challenges in disease management, and grape cultivation is no exception. Accurate and timely identification of grape leaf diseases is crucial for preventing outbreaks and ensuring healthy yields. Deep learning has emerged as a powerful tool for automated disease diagnosis, offering the potential to analyze images of grape leaves and identify various diseases with high confidence scores. However, the performance of these deep learning models heavily relies on the availability of large, diverse, and labeled datasets. In many cases, acquiring sufficient training data, especially for specific grape varieties or uncommon diseases, can be difficult and expensive.

Data augmentation techniques are commonly employed to address this limitation by artificially expanding the training dataset. Traditional augmentation methods involve simple transformations such as rotation, cropping, flipping, mirroring, color adjustments, and noise injection. While these techniques can introduce some variability, they are limited in their ability to generate truly novel and diverse samples. These limitations can lead to overfitting, where the model performs well on the training data but struggles to generalize to unseen examples.

Generative Adversarial Networks (GANs) offer a more sophisticated approach to data augmentation. GANs consist of two neural networks, a generator and a discriminator that engage in a competitive process. The generator learns to create synthetic images that resemble real images from the training data, while the discriminator learns to distinguish between real and generated images. This adversarial training forces the generator to produce increasingly realistic images to fool the discriminator.

Deep Convolutional Generative Adversarial Networks (DCGANs) are a specific type of GAN that utilize convolutional layers in both the generator and discriminator. This architectural choice enables DCGANs to capture the variations and patterns within images, making them well-suited for generating high-quality synthetic images. Another issue that DCGAN tackles is class imbalance, DCGAN can be used to generate synthetic images of under-represented classes, helping to balance the dataset and improve the model's ability to identify rare diseases.

The performance of this data augmentation technique is evaluated using a pre-trained Inception V3 model obtained from PyTorch. Inception V3 is a popular image classification model, its deep architecture and ability to capture complex features make it a suitable choice for evaluating the quality of the generated images and the impact of data augmentation on classification accuracy.

Furthermore, the Fréchet Inception Distance (FID) score is employed to assess the quality and diversity of the generated images. FID is a metric that compares authentic images' distribution with generated images' distribution in the feature space of a pre-trained Inception V3 network. Lower FID scores indicate that the generated images are more similar to the authentic images regarding their visual characteristics and diversity.

By combining the classification accuracy of Inception V3 with the FID score, this research provides a comprehensive evaluation of the effectiveness of DCGAN-based data augmentation for grape leaf disease identification. The findings contribute to the growing body of knowledge on GANs for data augmentation in agriculture. They offer insights into their potential for improving the accuracy and robustness of deep learning models in real-world applications.

II. Background

One of the basic principles of GAN is to generate a novel dataset that mimics the probability distribution of the original dataset. Let us assume the probability distribution of the generator as P_G and that of the initial data as P_{data}. The generator G maps the random data to the target probability distribution. This is shown as:

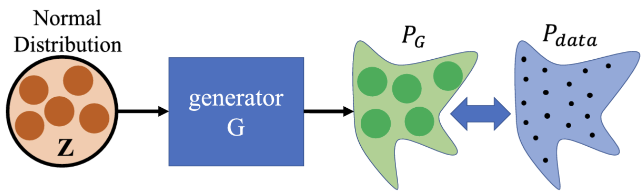

Fig. 1. The basic working mechanism of GANs

GANs try to develop a probability distribution that perfectly mimics the initial dataset under optimal conditions. As a result, the data generated by the generator does not increase the amount of information, and the model does not learn new information. To closely approximate the initial data distribution, the target generator G^* in Eq. (1) should aim to minimize the divergence between P_G and P_{data}:

$$G^* = \arg\min Div(P_G, P_{data}) \tag{1}$$

But, in reality, P_G doesn't perfectly match P_{data}, and as a result, the generator model cannot perfectly fit the probability distribution of the dataset. This property allows the generated images to be used as data augmentation so that further training can improve the accuracy of recognition.

III. Original Grape Leaf Dataset

All the grape leaf images both healthy and infected totalling 4062 images have been downloaded from the plant village dataset [4]. The distribution of images in each class is not perfectly balanced, the number of healthy images is almost 3 times lower than each of the infected images. The below figure shows one image from each class.

Class	Total
Black rot	1180
Esca measles	1383
Leaf spot	1076
Healthy	423
Total	4062

IV. Model Description

A. Generator

To accurately capture the features of images in the original dataset, the generator and discriminator models must go hand

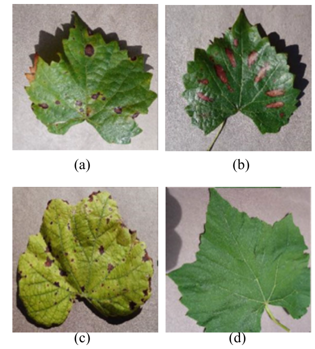

Fig. 2. The four types of grape leaf diseases. (a) Black rot. (b) Esca measles. (c) Leaf spot. (d) Healthy.

in hand. It is harder for the generator and discriminator models to work with contrasting features due to the dataset consisting of images of grape leaves affected by different kinds of diseases. Due to this reason, a generator model with a fully connected linear layer, followed by a transpose convolution layer and then a LeakyReLu layer, has been used[9]. The basic architecture of Generator is shown below:

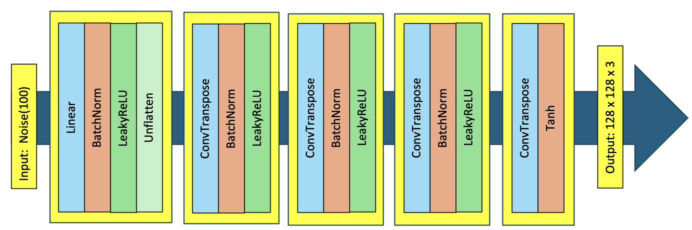

Fig. 3. The basic architecture of the generator

A fully connected linear layer maps the noise vector into a larger dimensional space, thus expanding it into a more significant vector. In this model, the fully connected linear layer expands the noise into a size of 8*8*256 elements or 16,384 dimensions, which acts as a suitable tensor for image generation. Once reshaped, it can pass to the other layers of the model.

Batch normalization[10] is performed on the generated data, which stabilizes and speeds up training and helps prevent mode collapse. Mode collapse occurs when the model produces a limited output set without considering the datasets' full diversity. This mitigates the chances of vanishing and exploding gradients during backpropagation. LeakyReLU layers are used instead of ReLU to introduce non-linearity into the model, allowing for smaller gradients when processing negative inputs.

Unflatten layer is used to reshape the vector from 8*8*256 to $(\text{batch\_size (i.e., 64)}, 256, 8, 8)$, preparing it for the later convolution layers.

Transposed convolutional (deconvolution) layers are used, thus upsampling the input tensor from spatial dimensions 8*8 to 16*16. Suitable kernel sizes, strides, and padding achieve the required dimensions.

Then normalization and LeakyReLU activation are used. We have repeated these steps three times to reach the target resolution 128*128. Tanh() activation gives the output pixel values in the range [-1,1]. This range is compatible with familiar image preprocessing methods.

Thus, the generator model progressively upsamples the noise vector through deconvolution, adding more spatial and semantic details at each step until it forms a realistic-looking image with three color channels and a resolution of 128*128. The loss of the generator is the same as the original GAN:

$$L(G) = L_{adv}(G) = \mathbb{E}_{x\sim P_{noise}}[\log(1 - D(G(z)))]$$

B. Discriminator

The discriminator model consists of 3 stacks of the same set of layers placed one after the other using the sequential method (PyTorch code).[8]. The discriminator's task is to downsample the image gradually and capture meaningful features that will help the model differentiate between real and fake images(i.e., generated images). The basic architecture of the discriminator is shown below:

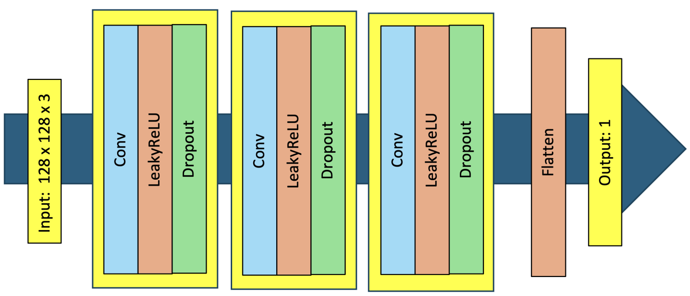

Fig. 4. The basic architecture of discriminator

The convolutional layer is added, which will transform the input channels into 64 feature maps. Kernel size is kept at 5 with a stride of 2 and padding of 2. The LeakyReLU activation function is used to introduce non-linearity and to avoid dead-neuron conditions (a condition when some of the neurons stop updation during the training process). After this, the Dropout layer is used, which prevents overfitting by introducing noise into the model's learning.

Again, the same set of layers is applied three times so that the model downsamples the input from 32*32 to 16*16 while capturing even more complex, high-level features. Finally, a flattened layer is used, which converts the 3D feature map into a 1D vector to find the output size and prepare it for fully connected layers to follow.

The adversarial loss of the discriminator model is defined based on the original GAN model[3]:

$$L_{adv} = -\mathbb{E}_{x\sim P_{data}} \log D(x) - \mathbb{E}_{z\sim P_G} \log(1 - D(G(z)))$$

C. Optimizer

The generator and discriminator need to adapt and learn so that the models provide realistic and optimal results. Adam Optimizer is used, which facilitates effective learning of parameters through gradient descents, which leads to faster convergence and brings stability to the model. Adam(Adaptive Moment Estimation) is based on two popular optimization algorithms, AdaGrad and RMSProp. The former adapts learning rates based on the history of gradients, and later adjuncts learning rates based on recent gradients. Momentum terms used by Adam Optimizer are controlled by Beta, a crucial hyperparameter, along with learning rates.

The generator and discriminator need a balance between them as they completely differ in their objectives. The generator typically benefits from slower adjustments to avoid drastic shifts in learning. We need to reduce Beta, which is kept at 0.5 in this model. This also helps to prevent the mode collapse. Second Beta is chosen at 0.999, which helps in slower convergence of the generator if the discriminator is updated more frequently. If ignored, then this will lead to situations where the generator will fail behind the discriminator, eventually leading to mode collapse.

V. Performance Analysis of DCGAN using Fréchet Inception Distance

A pre-trained image classification model trained on a massive dataset is used for calculating the Fréchet inception distance (FID) score; for the purposes of this paper, the Inception V3 model provided by PyTorch has been used. The final classification layer is removed, and the output from the penultimate layer, known as the image's embedding or feature vector, is used to capture the features of the images provided to the classification model. The image classification model has been repurposed; instead of using it to classify the image, we use it to extract its features.

Once the real and generated images' feature vectors (image embeddings) are obtained from the penultimate layer, we calculate their respective mean and covariances. These mean and covariances defined two multivariate Gaussian distributions in an n (2048 for the Inception V3 model) dimensional feature space (2048 $\times$ 2048 matrix). FID scores are then calculated based on the formula below.

$$FID(x, x^*) = ||\mu_x - \mu_{x^*}||_2^2 + Tr(C_x + C_{x^*} - 2(C_x C_{x^*})^{\frac{1}{2}})$$

Generated images that more accurately capture the underlying features of the original set of images should have a lower Fréchet inception distance. FID is less sensitive to minor pixel-level differences between images in contrast to other pixel-based metrics such as MSE, PSNR, or SSIM; this makes it resilient to noise and other artifacts generally present in generated images.

Furthermore, FID is particularly well-suited to this paper as only 16 artificial images are being generated. Given the limited number of generated images, observable improvements in overall image classification performance are unlikely. The FID score allows us to compare each image with the data distribution of all the images in the original dataset and assess the DCGAN model's performance.

VI. Results and Ablation Study

A. Learning Rate Scheduler

The initial learning rate scheduler used was CosineAnnealingLR which resulted in the graph of FID scores over training having unexpected peaks at certain epochs although the FID scores were really low at other points.

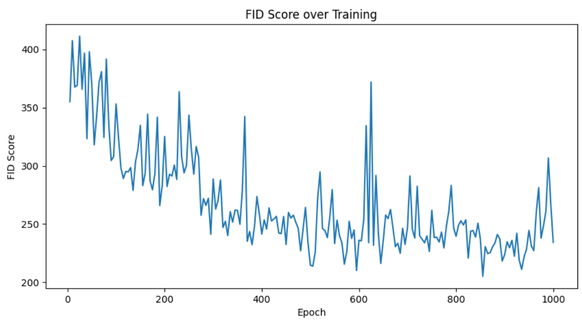

Fig. 5. FID scores: CosineAnnealingLR

Fig. 6. 16 Generated Images

To address the issues of the CosineAnneallingLR learning rate scheduler ReduceLRonPlateau was used which although reduced the undesired peaks resulted in FID scores being considerably higher than those obtained previously.

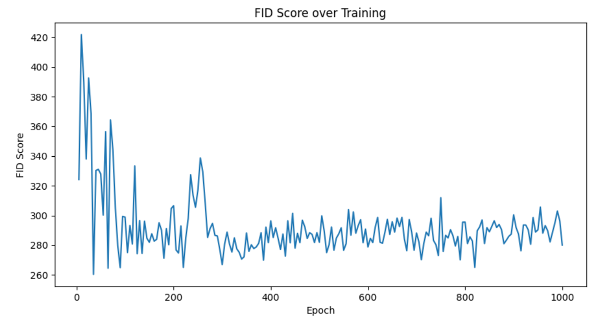

Fig. 7. FID scores: ReduceLRonPlateau

References

[1] A. Radford, L. Metz, and S. Chintala, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks,” arXiv preprint arXiv:1511.06434, 2015.

[2] J. Arun Pandian, G. Geetharamani and B. Annette, ”Data Augmentation on Plant Leaf Disease Image Dataset Using Image Manipulation and Deep Learning Techniques,” 2019 IEEE 9th International Conference on Advanced Computing (IACC), Tiruchirappalli, India, 2019, pp. 199-204, doi: 10.1109/IACC48062.2019.8971580.

[3] B. Liu, C. Tan, S. Li, J. He and H. Wang, ”A Data Augmentation Method Based on Generative Adversarial Networks for Grape Leaf Disease Identification,” in IEEE Access, vol. 8, pp. 102188-102198, 2020, doi: 10.1109/ACCESS.2020.2998839

[4] https://www.kaggle.com/datasets/zienabesam/grape-plant-from-plant-village-dataset

[5] Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[6] M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

[7] S. Dhar, M. T. Akhter, P. Banerjee, N. D. Jana and S. Das, ”FID-RPRGAN-VC: Fréchet Inception Distance Loss based Region-wise Position Normalized Relativistic GAN for Non-Parallel Voice Conversion,” 2023 Asia Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC), Taipei, Taiwan, 2023, pp. 350-356, doi: 10.1109/APSIPAASC58517.2023.10317438.

[8] E. Kim, H. -h. Cho, E. Ko and H. Park, ”Generative Adversarial Network with Local Discriminator for Synthesizing Breast Contrast-Enhanced MRI,” 2021 IEEE EMBS International Conference on Biomedical and Health Informatics (BHI), Athens, Greece, 2021, pp. 1-4, doi: 10.1109/BHI50953.2021.9508579.

[9] arXiv:1511.06434v2 [cs.LG] 7 Jan 2016

[10] D. Ulyanov, A. Vedaldi, and V. Lempitsky, Instance normalization: The missing ingredient for fast stylization, 2016, arXiv:1607.08022. [Online]. Available: http://arxiv.org/abs/1607.08022