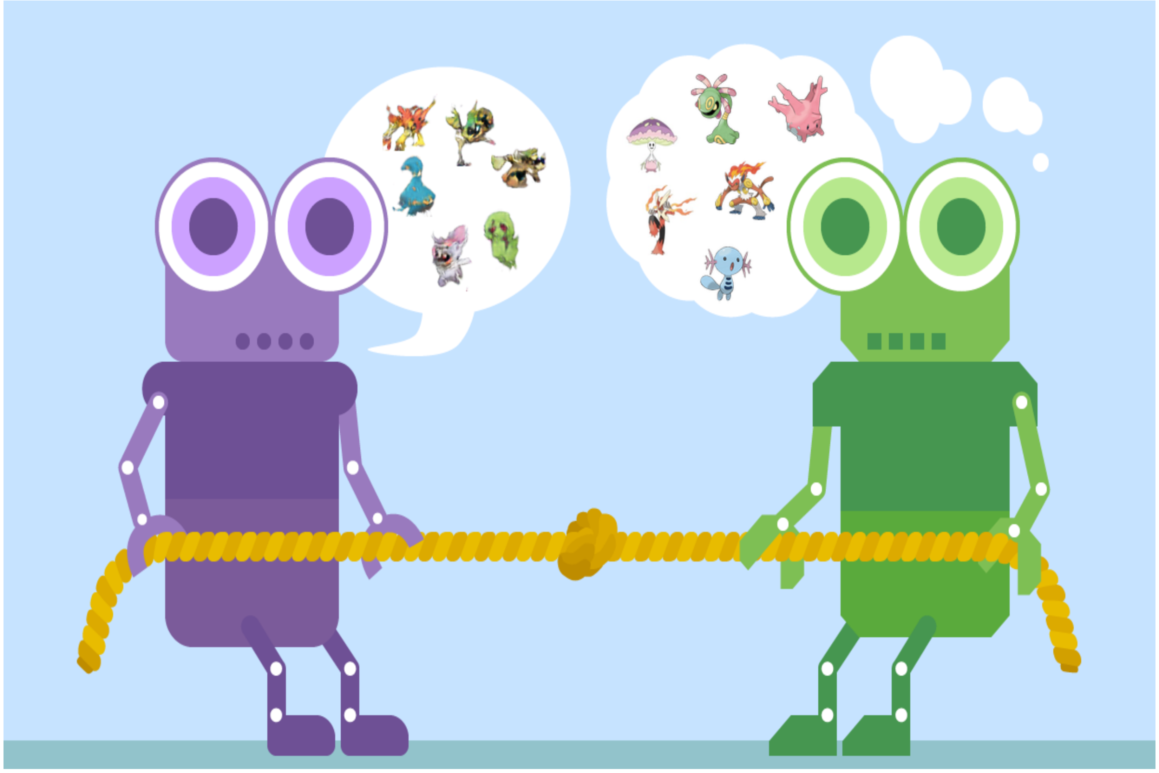
DATASET GENERATION WITH GANS

-Leveraging computational power to generate deepfakes



Divyansh Pandey

School of Engineering, Jawaharlal Nehru University

20/11/EE/053

TABLE OF CONTENTS

INTRODUCTION 1

Why do we need data? 1

Areas of use 2

Using Deep learning to generate synthetic data 3

LIBRARIES, FRAMEWORKS AND DATASET USED 2

Tensorflow 1

Pytorch 2

Matplotlib 3

Numpy and Pandas 4

About the datasets 5

THE GAN MODEL 3

About the model 1

Generator 2

Discriminator 3

PROFESSIONAL PRACTICES AND ETHICS 4

REFERENCES 5

INTRODUCTION

GANs or Generative Adversarial Networks have given out excellent synthetic picture outcomes. GAN's finest video is just a few seconds long, garbled, and low-resolution at the time of writing. I researched how to utilize several GAN models to generate images as part of my computational biology assignment. I constructed a deep convolutional GAN (DCGAN) to produce fake images that can be used in the medical domain in particular. Any image may be generated with this model. The GAN model in this project employed convolutional layers. I also looked at how GAN might be used with 2D convolutions.

WHY DO WE NEED DATA?

The efficacy of artificial intelligence in the medical areas is dependent on the availability of large amounts of high-quality training data. On the other side, medical image data sharing is frequently forbidden by restrictions such as doctor-patient confidentiality. Despite the fact that medical databases are freely available on the internet, their quality and quantity are not always enough. Furthermore, datasets are typically imbalanced, comprising just a tiny fraction of the images generated by scans in hospitals or clinics, and hence may only be used as training data for specific purposes. Generative adversarial networks, or GANs, produce fake images by training two convolutional networks. This paper describes a method for creating a large number of simulated images using GANs trained on medical data, which might be used to train other AI systems.

This method is a first step in addressing data privacy issues and allowing for the public sharing of data that preserves a considerable portion of the information contained in the original private data. The method has been tested on a number of public datasets and has produced positive results in both quantitative and qualitative evaluations.

Using Deep learning to generate synthetic data

Within a hospital, the variety of pictures made varies greatly, depending on the patient, the displayed content, and the methods employed. Artificial intelligence can minimize the labor of doctors and enhance the overall efficiency of hospitals and clinics by performing classifications, regression, and segmentation on pictures. To produce accurate findings, an effective implementation, such as the classifier to identify skin melanomas, needs more than 100,000 photos. However, huge businesses and hospitals are generally the only ones with access to such enormous volumes of photos, as this information is sensitive and cannot be shared with the public owing to doctor-patient confidentiality. GANs (generative adversarial networks) might be used on enormous datasets within hospitals to produce synthetic data that could be shared with the general public.

Medical pictures such as X-rays, MRIs, and CT scans are produced at hospitals. The photos are then put to their intended use like patient diagnosis, operation planning, and so on. Due to data privacy restrictions, the photographs cannot be uploaded to a cloud and shared with the public. The identical photographs are preserved on a hospital server which is usually not open to the public. Any authorized employee can launch a user application to train a model that reflects the images' statistical data. Any deep learning model that may be utilized in this stage is represented by a neural network. It translates the data it receives into statistical data. The model produces images that displays a wide range of attributes from the model's input photos while still displaying the same core item. It may be used to make a wide range of fake pictures. The produced photographs or trained model may then be uploaded to the cloud and shared with the general public without fear of data privacy concerns. Only a subset of the created data with a specific distance from the originals may be shared, so order to avoid sharing generated data that was too similar to any of the original photos. The potential to collect more thoughts and research findings regarding fixing difficult and frequent medical problems is made possible by the increased availability of such photographs.

LIBRARIES, FRAMEWORKS AND DATASET USED

Pandas

pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data strutures and data analysis tools for the Python programming language. pandas is a Python package providing fast, flexible, and expressive data structures designed to make working with “relational” or “labeled” data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real-world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis/manipulation tool available in any language. It is already well on its way toward this goal.

Numpy

NumPy forms the basis of powerful machine learning libraries like scikit-learn and SciPy. As machine learning grows, so does the list of libraries built on NumPy. TensorFlow’s deep learning capabilities have broad applications — among them speech and image recognition, text-based applications, time-series analysis, and video detection. PyTorch, another deep learning library, is popular among researchers in computer vision and natural language processing. MXNet is another AI package, providing blueprints and templates for deep learning.

Statistical techniques called ensemble methods such as binning, bagging, stacking, and boosting are among the ML algorithms implemented by tools such as XGBoost, LightGBM, and CatBoost — one of the fastest inference engines. Yellowbrick and Eli5 offer machine learning visualizations.

Matplotlib

Matplotlib is a Python tool that allows us to visualise data in a fixed, vivid, and synergistic way. Matplotlib is used by a number of different plotting libraries, including plotly and seaborn. Matplotlib simplifies the visualisation and interpretation of both basic and complex data. Without it, we would have the hassle of writing long codes for visualizing graphs and plots of the important features of data.

Pytorch

PyTorch is an open sourced machine learning framework which is primarily based on the Torch library that is used for numerous technologies such as computer vision and natural language processing. It was originally composed by Facebook's (now Meta) Artificial Intelligence Research department. It's open-source software and free to use. It uses GPU and CPU for computations and is used for deep learning similar as tensorflow, theano or keras. It is considered to be best for research oriented projects since models can be trained extremely fast using this.

TensorFlow

TensorFlow is an open source machine learning platform that runs from start to finish. It consists a large, flexible collection of tools, libraries, and community resources that allows researchers to advance the state-of-the-art in machine learning techniques while also allowing developers to quickly construct and deploy ML-powered apps.

TensorFlow consists of a set of procedures that allow both novices and professionals to develop machine learning models in a variety of languages using easy, high-level APIs. Models may be deployed on a variety of platforms, including servers, the cloud, mobile and edge devices, browsers, and a variety of other platforms. This makes it considerably easier for developers to move from model creation to training and deployment.

About the datasets

The datasets are taken from kaggle. They consist of a X-ray dataset containing pneumonical patients, a CT scan dataset of adinocarcinoma patients, a dataset of lung tissues, and a dataset containing gaussian filtered images used for diabetic retinopathy.

Below are the links of datasets-

https://www.kaggle.com/mohamedhanyyy/chest-ctscan-images

https://www.kaggle.com/tolgadincer/labeled-chest-xray-images

https://www.kaggle.com/andrewmvd/cancer-inst-segmentation-and-classification

https://www.kaggle.com/sovitrath/diabetic-retinopathy-224x224-gaussian-filtered

About the model

The type of dataset is identified using a function that employs structural similarity index as a measure to check similarity between images. Out of all the methods of error and similarity detection, it was experimentally found that SSIM was the beast measure to find the structural similarity between images.
The DCGAN approach allows for the creation of clear, realistic-looking pictures while avoiding mode collapse and training instability caused by imbalanced Discriminator and Generator. The functions utilized inside the equation must be 1-Lipschitz in order to solve it and get the minimum loss. This is accomplished by using the gradient penalty, which penalizes any function that deviates from the

intended norm value of 1. The GAN used in this paper is based on images with a resolution of 128, 128, 3. Both Generator and Discriminator are convolutional neural networks with the properties listed in. Generator receives a random noise vector as input. It is then upsampled by transposed convolutional layers with a stride of 2 utilizing 512, 256, 126, 64, and 32 distinct filters Discriminator receives a 4 L. Middel et al. 128,182, 3 picture, which can be real or created, and downsamples it using four convolutional layers with 64, 128, 256, and 512 filters each. Because there are presently no standard settings that perform well for all datasets, hyperparameters (learning rate, batch size, and kernel size) were determined through trial and error. With a learning rate of 3e-4, a batch size of 128, and a kernel size of 5, values of hyperparameters with overall satisfactory visual outcomes were attained. The models were trained for 150 epochs with all pictures accordingly. The names of the datasets are listed above. Although the number of epochs might be increased, the discriminator and generator losses were stabilized at the quantity set.

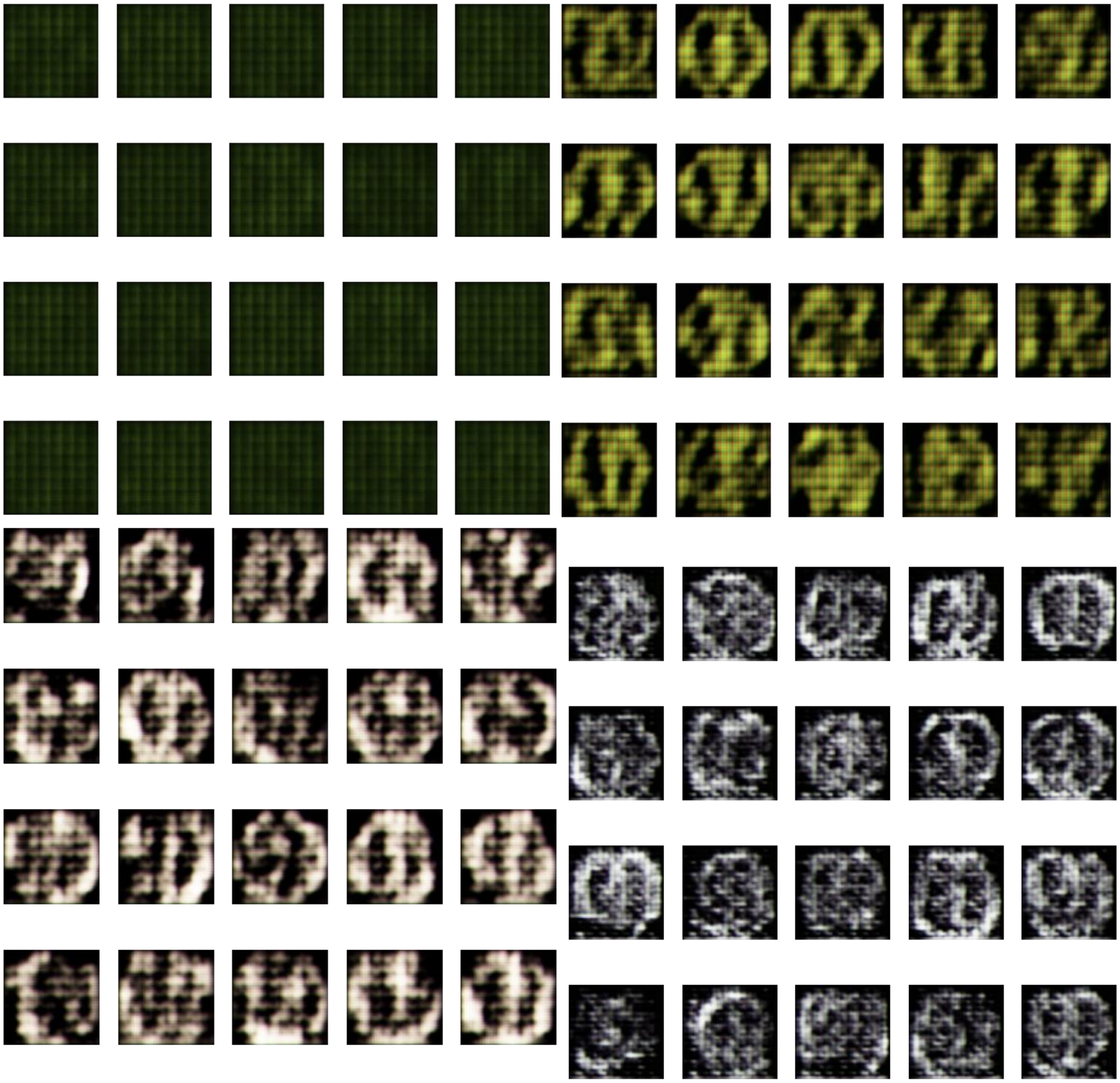
Discriminator

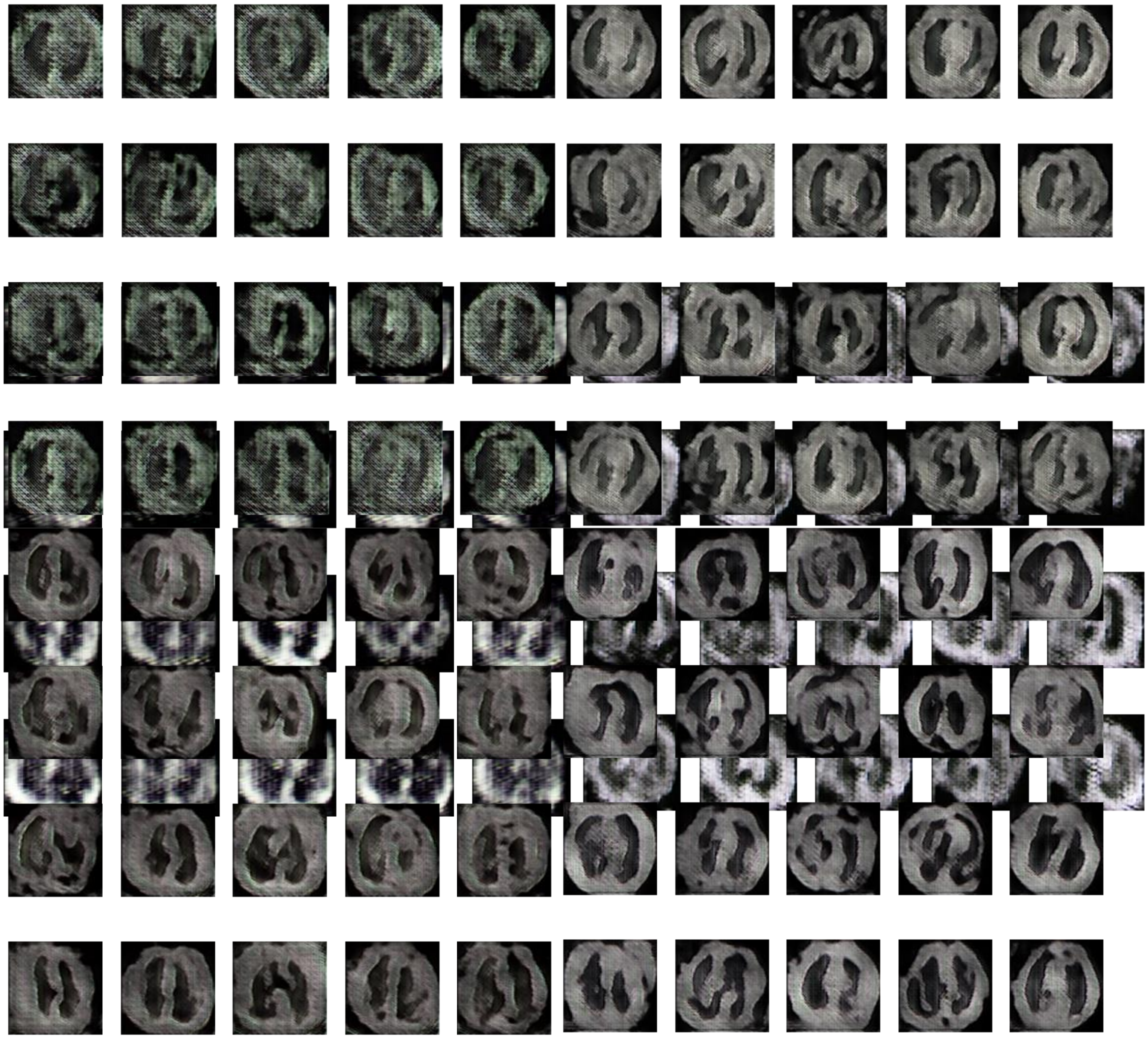
The discriminator is based on the sequential API and consists of 5 consecutive Conv2D layers followed by a LeakyRelu layer each with an alpha param of 0.2. Next It contains a flatten layer to flatten the output coming from the last leakyrelu layer followed by a Dropout and dense layer with a sigmoid activation. The optimizer used is Adam and the loss function employed here is binary_crossentropy and metrics, accuracy.

Generator

The generator is also a sequential model that begins with a dense layer with a input dimension equal to the latent dimension. It is then followed by a mixture of LeakyRelu and Conv2DTranspose layers and a final Conv2D layer with a tanh activation function

Below are the images of how the model evolves as the generator improves -





7

Ethics

Deepfake developers and distributors must guarantee that synthetic images is used and implemented responsibly. Big technological platforms like Microsoft, Google, and Amazon have a moral obligation since they supply tooling and cloud computing resources to generate deepfakes quickly and at scale. News media organizations and journalists, legislators and policymakers, and civil society must all demonstrate an ethical and social responsibility toward the use of deepfakes, as do social media platforms like Facebook, Twitter, LinkedIn, and TikTok, which offer the ability to distribute a deepfake at scale.

The social and technological platforms have an ethical commitment to avert damage. While users on these platforms have a duty to share and consume material, structural and informational inequalities make it difficult for users to effectively respond to harmful deepfakes. It's possible that transferring the responsibility for responding to malevolent synthetic media on users is ethically acceptable. Nonetheless, platforms must do the right thing and carry main responsibility for detecting and blocking the dissemination of false and altered information.

Most technological and social media platforms have procedures in place to deal with misinformation and hostile synthetic media, but they must be ethically sound. For example, if a deepfake has the potential to do considerable harm (reputational or otherwise), platforms must delete it. To prevent the spread of deepfakes on their networks, these platforms should include dissemination limits or differential promotional methods such as limited sharing or downranking. Another helpful approach is content labeling, which should be used honestly and clearly, without regard for political prejudice or business model issues.

Platforms have an ethical responsibility to establish and sustain their user community's dissemination rules. Content producers can be influenced by how community standards, community identity, and user submission limits are framed. Norms and community norms, such as examples of desired behavior and positive expectations of users as community members, can promote behavior that meets those goals. Terms of service and platform policies are important in avoiding the spread of dangerous fake news.

Institutions who want to counteract problems caused by misleading media owe it to their constituents to provide access to media literacy programs. To create resiliency and engage intelligently in the consumption, processing, and sharing of information, platforms must equip users with knowledge and critical media literacy abilities. While enjoying satire and parody, practical media knowledge may empower people to think critically about the context of media and become more involved citizens.

Conclusion

The model was able to successfully generate desired numbers of output images after feeding in a considerable amount of data. The images generated are of equal dimensions and are also downscaled as such they can be directly fed into the machine learning models and receive good results.

References

Img Src - https://www.google.com/url?sa=i&url=https%3A%2F%2Fblog.wolfram.com%2F2020%2F08%2F18%2Fgenerative-adversarial-networks-gans-in-the-wolfram-language%2F&psig=AOvVaw1ITEZrm8DS5-fueaMZffLT&ust=1646039324347000&source=images&cd=vfe&ved=0CAsQjRxqFwoTCKiGg8DEn_YCFQAAAAAdAAAAABAD

Deep Learning with Python – by Francois Chollet

https://www.orfonline.org/expert-speak/debating-the-ethics-of-deepfakes/

https://www.researchgate.net/publication/336430952_Synthesis_of_Medical_Images_Using_GANs