COVID-GAN - Augmenting COVID-19 Diagnostic Data using Hybrid VAE-GANs

Shorya Sharma
Indian Institute of Technology Bhubaneswar
ss118@iitbbs.ac.in

Abstract

The COVID-19 pandemic has made clear the need to offload some of the work off the front-line workers. It is a known fact that medical image data for this disease suffers from skewness due to the limited availability of positive COVID-19 patient data, and the infeasibility of collection of additional data. Deep learning models rely heavily on the uniformity of the data to produce a good result and hence often overfit on highly skewed data. Increasing the number of training samples has been shown to provide better classification accuracy and more generalizability, and data augmentation on the available training samples has proven effective in reducing class imbalance. Many approaches to data augmentation have been proposed, from basic image transformations to using a Generative network to create new images. With this project, we propose three distinct generative networks to augment the COVID-19 data on which a CNN-based classifier is trained to achieve better performance. All of our models, a Variational Autoencoder (VAE), a Generative Adversarial Network (GAN) and a mixed VAE-GANs model lead to an increase in accuracy. The images developed by the third network provided the greatest marginal increase in performance for the CNN-based classifier

1 Introduction

Most widely used deep learning algorithms, such as convolutional neural networks (CNN), require a large number of data points to be employed successfully. Acquiring this data can be expensive and time consuming, and, especially in the given pandemic, it becomes infeasible. In the case of patients that have contracted COVID-19, the viral disease caused by severe acute respiratory syndrome coronavirus 2, data acquisition has been difficult due to the novelty of the disease and the strain it has put on healthcare systems around the world. An important step in the care for patients in critical conditions is the correct identification of COVID-19 in chest X-Rays. To ease the burden on healthcare professionals, and to improve the quality of care received by patients, the use of neural networks has been proposed to identify the sign of COVID-19 in chest X-Rays [1] While COVID-19 symptoms in lungs present distinct features [2], it has been difficult to identify the differences between COVID-19 and other lung diseases, such as pneumonia, when using neural networks, due to the lack of samples available for this disease.

An approach to this problem that has received increasing attention has been data augmentation. Various approaches have been proposed, with good results shown specifically by Auxiliary Classifier Generative Adversarial Networks. (ACGANs). In the domain of data augmentation, good results

have been shown by a combination of GANs and Variational Autoencoders (VAEs) in the creation of facial images [3]. We propose the use of a similar approach to augment the chest X-ray data available for COVID-19. We propose four baseline CNN models, used to evaluate the performance on the original dataset. We then plan to augment the data available using VAEs, GANs and our combination of the two techniques, and evaluate the performance of the four baseline models on the augmented dataset. The results will also be compared to those obtained by the ACGANS model built by Waheed et. al. [1].

2 Literature Review

Y. Wang et al [4] recently published that chest radiography is a better method to diagnose COVID-19 patients than laboratory testing. Therefore, it is critical to develop digital solutions that facilitate the interpretation and classification of chest x-ray scans. D.S Kermany et al [5] and O Stephen et al [6]. have recently presented deep neural networks that accurately classify pneumonia infected lung x-ray scans. The D.S Kermany neural network has a testing accuracy of 92.8 percent. O Stephen et al presented a network that consisted of 4 convolutional layers, 2 dense layers and a conventional image augmentation framework. The O Stephen neural network achieved a testing accuracy of 93.73 percent. Saraiva et al. [7] devised a convolutional neural network that classifies images of childhood pneumonia. The Saraiva network had 7 convolutional layers, 3 dense layers and achieved a testing accuracy of 95.30 percent. G.Liang and L. Zheng [8] presented a transfer learning method with a deep residual network for childhood pneumonia diagnosis. Liang and Zheng's network consisted of 49 convolutional layers, 2 dense layers and achieved a testing accuracy of 96.70 percent.

Most medical images training datasets are highly imbalanced and expensive to acquire. Therefore, in recent years, the approach of utilizing GAN's framework to augment medical image datasets has become prominent. Zhao et al [9] devised a multiscale VGG16 network, DCGAN model and a forward and backward GAN to generate images for lung-nodules classification. Dai et al [10] applied the GANS framework to produce segmented images of the lung and Nie et al [11] developed a patch GANS framework to convert brain CT pictures to corresponding MRI images. Beers et al [12] developed a progressively grown generative adversarial network to output fundus pictures that display premature retinopathic vascular pathology. A. Waheed et al [1] recently presented an auxiliary classifier GAN framework that augments COVID-19 infected chest x-ray data sets. Waheed's GAN framework consists of 4 convolutional transpose layers and multiple dense layers. Waheed's GAN framework was applied to a VGG16 convolutional neural network (CNN) that identifies COVID-19 inflicted lung x-ray scans. The GAN framework increased the CNN testing accuracy from 85 percent to 95 percent.

A. Larsen et. al. 2015recently presented a hybrid generative network that combined VAE and GANS frameworks to develop high quality images. A. Larsen et. al. 2015 applied the hybrid framework to images of faces and found that it outperforms conventional frameworks (VAE and GANS) with element-wise similarity measures in terms of visual fidelity. The promising results of the novel framework motivates us to apply the hybrid approach to the augmentation of medical images.

3 Contribution

3.1 Proposed Approach

Given the skewed nature of the COVID-19 Dataset, the traditional deep learning algorithms that often produce sophisticated results with large datasets, suffer from over-fitting and fail to generalize when they are trained in a small data set like this. Especially with medical images, this is a common occurrence due to the lack of robust datasets to train models on. In our case, as the COVID-19 itself has been a novel and recent disease, the data available is limited. This inherently arises the need for data-augmentation approaches to build and enhance the datasets and then enable deep learning models to run on these augmented datasets. Through our project, we would like to reiterate and demonstrate the need for data augmentation on the COVID-19 Dataset and identify the best way to achieve this, while evaluating various deep learning generative models and classifiers.

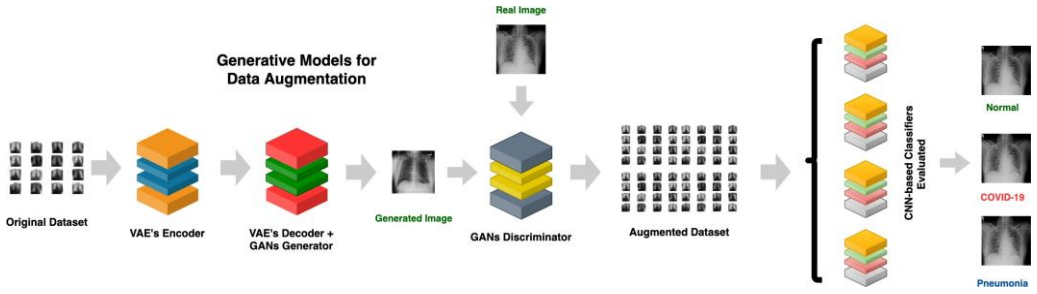

Figure 1: Proposed pipeline of COVID-GAN using a combined VAEs-GANs generative model

The proposed pipeline has the following components:

- **Generative Models for Data Augmentation**
  We propose multiple techniques to augment the original dataset that has fewer COVID-19 images. This includes simple techniques like Transformation like flipping, rotating, adding some noise to the original dataset and complex generative models like Variational Autoencoders (VAEs) and Generative Adversarial Networks (GANs) as in Boesen et al [13] , to generate more COVID-19 X-Ray images. This segment of the model is generate the image corresponding to the encoded feature vectors from the previous layer, by combining VAEs and GANs into an unsupervised generative model that simultaneously learns to encode, generate and compare dataset samples. This would clearly be the novelty in our project, adapting and retraining the model until the divergence (KL Divergence) between the true input image and the generate model is minimized so that the model is ready to be tested with the CNN-based frameworks without over-fitting.

- **CNN-based Classifiers for detecting the presence of COVID-19**
  The Classification problem aims to identify whether COVID-19 is present in an image, hence we have three main classes - COVID-19, Pneumonia and Normal. In our baseline, we have explored various CNN-based classifiers that are simpler models with low complexity suited for smaller datasets, namely vanilla ResNets, simplified ResNets, Inverted BottleNeck ResNets,and a VGG-based classifier on the original dataset. Although we intend to extend this list to AlexNets, GoogleNets and SqueezeNets in the coming phase of the project. Having evaluated these on the original dataset, we expect to see over-fitting and poor performance in general. Once the GANs based generative models are explored, we intend to evaluate the performance of the augmented dataset using the aforementioned classifiers.

3.2 Novelty in our Proposed Approach

As most of the existing work focuses on just using GANs alone for data augmentation, we decided to introduce novelty in the approach through our hybrid VAE/GANs model (inspired by A. Larsen et al 2015). There are a couple of reasons for proposing a hybrid-model to augment the dataset and enhance the performance of the baseline classifier.

- **Mitigating the training instability of GANs with VAEs**
  Given how difficult and unstable GANs are in training, we proposed VAEs and GANs into an unsupervised generative model that simultaneously learns to encode, generate and compare dataset samples. Also, we explored other techniques that weren't studied before in light of the COVID-19 Dataset. One of the main reasons we implemented DCGAN over ACGAN (from the paper) is to try other types of GANs models rather than reimplementing the same approach in the paper. With the aim of also evaluating VAEs, we noticed that VAEs do generate the images but they are of a poorer quality compared the the DCGANs model, this is both from a manual looking at the dataset and also how the augmented dataset with the VAE-generated images evaluated against the same baseline classifier. Although the VAE augmentation didn't really help boost accuracy, it's still a result that GANs outperform the VAEs and a combined model beats both of them. VAEs essentially use element-wise metrics like the squared error, which might be simple but not very suitable for image data,

as they do not model the properties of human visual perception. This clearly explains why VAEs fail to generate good quality images, we need a higher level representation of the images and sufficiently invariant representation of the images.

By combining VAEs with GANs, we can eliminate these element-wise metrics and utilize the feature-based metrics that's used in a GANs Decoder based on the features generated by the VAEs encoder. We collapse the VAE decoder and the GAN generator into one by letting them share parameters and training them jointly. This in turn makes the VAE-GANs model outperform the VAE model. Instead of forcing a low reconstruction error, VAE-GAN imposes a low error between intermediate features in the discriminator. The reconstructed image can be considered very similar to the original one their middle-representation in the discriminator should be similar too. We intend to show how the combined model would generate better image quality than the individual GANs and VAEs models. The stabilization of GANs with the help of the encoder step is one of the novel models we bring to the COVID-19 Dataset augmentation problem. We regard our method as an extension of the VAE framework. Though, it must be noted that the high quality of our generated images is due to the combined training of Dec as a both a VAE decoder and a GAN generator. This makes our method more of a hybrid between VAE and GAN.

- **Handling the over-fitting of Imbalanced Classes**
  As it is a known fact that a small dataset would render ineffective in training due to over-fitting of the classes, this would also be the case for training GANs too. We decided to use Deep Convolution GANs (DCGAN) over what the paper had used Auxiliary Conditioned GANs (ACGAN) despite the fact that the dataset is small. The intuition behind this is to evaluate other types of GANs rather than the one on the paper, we still achieved quality reconstructed images without the class labels passed to the discriminator. We used class-weights as a part of smoothening the imbalanced nature of the classes while evaluating our baseline CNN on the augmented data. We also used techniques like normalization and early stopping to prevent the over-fitting of data just when the validation loss starts increasing, which ensures that our model produces coherent results on the augmented data.

- **Augmentation of the COVID-19 Dataset**
  One of our noticeable contributions is taking the COVID-19 research in a direction where stable Deep learning algorithms can be applied through stabilizing the generative models that can help augment the dataset. We believe that by performing more augmentation, and as more data becomes available, the performance of the classifier can be boosted from 94% to 98-99%. Hence, once more COVID-19 data is out, we can augment the datasets and balance them even more and then this could truly be used in a medical setting.

4 Experimental Evaluation

4.1 All Experimental Models

To put our approach into an implementation phase, we have designed the following experimental models to achieve the COVID-GAN model.

1. **Baseline Evaluation of CNN-based Classifiers on Original Dataset**
   Various CNN-based classifiers that are simpler models with low complexity suited for smaller datasets, namely vanilla ResNets, simplified ResNets, Inverted BottleNeck ResNets, and a VGG-based classifier (extended to AlexNets, GoogleNets and SqueezeNets) will be evaluated on the Original Dataset that is unaugmented. Here, we expect to see over-fitting and poor performance in general. This experimental model is primarily to infer the best classifier suited for the classification task, as completed in our baseline model.

2. **Evaluation of a Generative model using only GANs**
   In this model, we intend to implement a GANs based generative model on the original dataset to achieve augmentation and evaluate the baseline classifiers on this type of data

augmentation. We intend to evaluate various types of GANs.

3. **Evaluation of a Generative Model using only VAEs**
   In this model, we intend to implement a VAEs based generative model on the original dataset to achieve augmentation and evaluate the baseline classifiers on this type of data augmentation.

4. **Evaluation of a Generative Model using a combined VAEs+GANs pipeline**
   In this model, we intend to implement the proposed model that combines the VAEs+GANs for encoding and generating the images and a GANs' discriminator to generate images based on the original dataset to achieve augmentation and evaluate the baseline classifiers on this type of data augmentation. **We hypothesize that this would give the best augmented dataset and expect to see a better performance of the classifiers on this data.**

4.2 Dataset Description

The dataset for the baseline model consists of Chest X-Ray images of 6544 subjects with three conditions - COVID-19, Pneumonia and Normal lung. The data is divided into 5882 training images, 27 validation images and 635 test images. 472 X-Rays belong to the COVID-19 class, 4489 belong to the Pneumonia class and 1583 normal lung X-Rays. The images are resized to 384x384 pixels to input to the classification model. A sample image is shown in Figure 8

4.3 Evaluation Metrics

In order to evaluate the performance of the generative models that augment the data, we have decided to use the following evaluation metrics on the classification task:

- **Testing Accuracy:**
  This is the evaluation metric we have used in our baseline evaluation owing to the simplicity. We have observed over-fitting when the training accuracy increases while the validating accuracy drops beyond a certain point. This would be more handy when we test these on the augmented data.

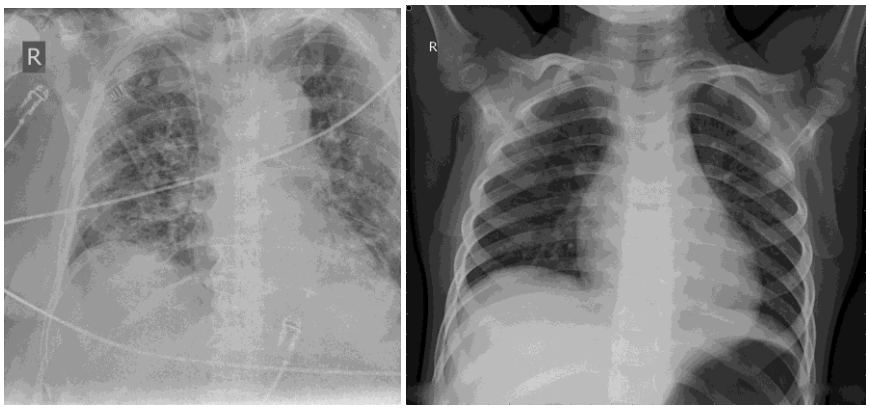

Figure 2: Sample Chest X-Ray images

5 The Baseline Model - Classifier Neural Nets

For our baseline model, due to the unprecedented pivot of the project, we finished our first experimental model - evaluation of CNN-based classifiers on the original dataset. The other experimental models explained previously will be tackled in the coming days on a rolling basis.

5.1 Baseline model using Vanilla ResNets - ResNet18 and ResNet34

Explored ResNet18 and ResNet34 for the classification of the original dataset. Each input was trimmed to a consistent shape of [3, 384, 384] before passing them through the ResNets as described below:

Network Architecture - ResNet18 and ResNet34:

- A very deep Residual Network with skip connections having Blocks of:
  Conv2d -> BatchNorm -> ReLU with Shortcuts, MaxPool after first Block
- Used Batchnorm2D (eps=1e-05 and momentum=0.5)
- ReLU and Leaky-ReLU used as the activation function
- Used SGD Optimizer and an Exponential LR scheduler with decay rate 0.9
- Used XELoss to compute the divergence between predicted and target labels of Classifier

Both the models were early-stopped just when the model's validation accuracy started to decrease steadily over 3 epochs or when it saturated.

5.2 Baseline model using a Simplified ResNet18

The simplified version of ResNet18 substituted max pooling in the last layer for average pooling, as well as capping the model at the 256 convolutional layer instead of continuing to the model's proposed 512 convolutional layer. For this variation, each convolutional layer was followed by a Batch Norm, ReLU and Max Pooling. In the first, third and fifth layers the Max Pooling is substituted with a Dropout with rate 0.3. The model is completed with a convolutional layer of size 256 followed by a linear layer of the same size. The Average Pooling is substituted with Max Pooling and no Fully Connected layer is implemented. The model was based upon literature and previously implemented homework.

5.3 Baseline model using Inverted Bottleneck ResNet

This network consists of four building blocks (classes):

- The first building block is named 'ConvBNReLU'. ConvBNReLU is a sequential class and contains the following layers: single 2D convolutional layer; 2D batch normalization layer; dropout layer; ReLU activation layer.
- The second building block is named 'Inverted Bottleneck'. Inverted Bottleneck is a module class and contains the following layers: three ConvBNReLU layers (with varying kernel sizes); single 2D convolutional layer; batch normalization layer. In the 'forward' function, the input first goes through all the convolutional and batch normalization layers. The cumulative output of said layers is then added to the original input of the Inverted Bottleneck class.
- The third building block is named 'Transit'. Transit is a module class and contains the following layers: single 2D convolutional layer; batch normalization layer; ReLU activation layer. In the 'forward' function the input of the class goes through all layers in a sequential manner
- The fourth building block is named 'Block'. Block is a module class and contains the following layers: two Inverted Bottleneck layers; single Transit class; single 2D convolutional layer. In the 'forward' function, the input first goes through all the Inverted Bottleneck and transit layers. The cumulative output of said layers is then added to the output of a 2D convolutional layer whose input is the original input of the 'Block' class.

The comprehensive network class consists of the following layers: single 2D convolutional layer; batch normalization layer; ReLU activation layer; multiple 'Block' layers; linear layer (outputs predicted labels of x-ray images).

The metrics used to measure the performance of the network was the classification accuracy of the training data set and the classification accuracy of the validation data set. The devised network was trained over 10 epochs with a batch size of 10.

5.4 Baseline model using VGG-16

The model has 13 2D Conv layers, divided into blocks of sizes [2,2,3,3]. Each block is followed by a maxpool layer. The final bloack is followed by three fully connected layers with 4096 nodes each. Every Conv 2D layer includes a BatchNorm and a ReLU layer. Dropout is used with a probability of 0.3. An adaptive average pool layer is used after the final Conv 2D block. The model starts with an initial learning rate of 0.005 and updates it every 2 epochs with a step scheduler (decay rate = 0.9). SGD and XELoss were used as the optimiser and loss function as in other approaches described above. The model gave the best performance after 17 epochs with a training batch size of 8.

6 The Generative networks

6.1 VAE Generative network

Before implementing the VAE, the training and testing data sets of the chest x-ray images underwent various transformations. The first transformation involved converting all images into PIL images. This was necessary as the data set had a wide variety of different image file types. Subsequently all images were resized to 100*100 and gray scaled to one output channel. By resizing and gray scaling all our images we reduced our computational overhead and brought greater consistency to our data set. The third transformation involved horizontally flipping some of our images. By implementing such a transformation, we hoped to diversify the type/orientation of images the network would train upon, hence making the network more generalizable and versatile. Our last transformations involved converting the data set's images to tensors and normalizing the images pixel values in the range of -1 to 1. By normalizing the pixel values we hoped to decrease bias in our network training procedure and increase the network's learning rate. Post transformations, all the image tensor shapes were [1,100,100].

The VAE module consisted of 4 functions. The initiation function consisted of 5 linear layers (fc1, fc21, fc22, fc3 and fc4). The first 3 linear layers (fc1, fc21, fc22) were utilized by the encoder function. 'fc1' had input dimensions of 10000 and output dimensions of 400. 'fc21' and 'fc22' have input dimensions of 400 and output dimensions of 20. In the encode function the output of 'fc1' goes through a 'relu' activation function. The output of the 'relu' activation function is inputted through 'fc21' and 'fc2'. The next function is the 'reparameterize' function. The arguments of the 'reparameterize' function are the mean and log-variance values outputted by the 'encode' function. The log-variance value and 'torch.exp' function is used to calculate the standard deviation. The standard deviation value and 'torch.randnlike' function (returns a tensor of the same shape as input but with random values) is used to calculate epsilon. The 'reparameterize' function outputs the aggregates the mean value and the product of the epsilon and standard deviation values. The next function is the 'decode' function. The decode function utilizes layer 'fc3' (20 input dimensions and 400 output dimensions) and layer 'fc4'(400 input dimensions and 10000 output dimensions). The output of 'fc3' goes through a 'relu' activation function. The output of the 'relu' activation function goes through layer 'fc4'. The output of 'fc4' goes through a sigmoid activation function. The output value of the sigmoid function is returned by the 'decode' function. The fourth function of the VAE module is the 'forward' function. In the 'forward' function the mean and log variance value is extracted from the encoder. The extracted mean and log variance values are used as arguments for the 'reparametrize' function. The output of the reparametrize function is used as the input for the 'decode' function. The forward function returns the output of the 'decode' function and the extracted mean and log-variance values. The VAE was based upon literature models

The loss function of the VAE network had two components. The first component is the reconstruction loss. The reconstruction loss is defined as the binary cross entropy loss between the generated and real images. The second component is the KL divergence loss. The KL divergence function is defined by the following equation (variance and mean extracted from the encoder):

$$0.5 * \sum(e^{variance} + \mu^2 - 1 - variance)$$

The final loss function is the sum of the reconstruction loss and the KL divergence loss function.

The VAE was trained in a two step sequential manner. The first step involved training the VAE on all classes of the training data set. The second step involved fine tuning the pre-trained VAE on either COVID-19 or Normal class training data set.

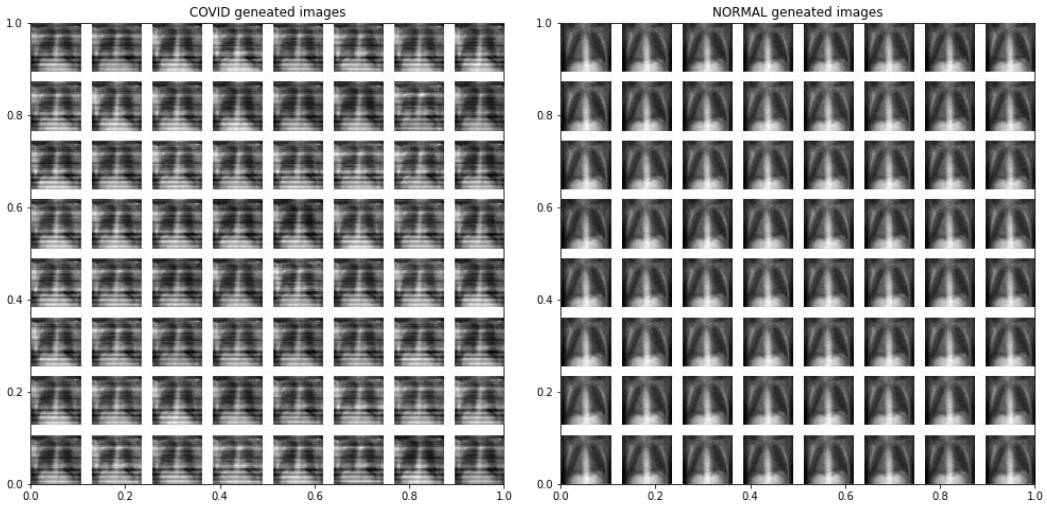

Figure 3: VAE Generated images for COVID class (left) and Normal class (right)

6.2 GANs as a Generative Model

Before implementing the GANs, the training and testing data sets of the chest x-ray images underwent various transformations. The first transformation involved converting all images into PIL images. This was necessary as the data set had a wide variety of different image file types.

Subsequently all images were resized to 64*64 images and converted to tensors and normalized the images pixel values in the range of -1 to 1 using a mean and std of 0.5 across all the channels. By resizing and normalizing all our images we reduced our computational overhead and brought greater consistency to our dataset. The third transformation involved horizontally flipping some of our images. By implementing such a transformation, we hoped to diversify the type/orientation of images the network would train upon, hence making the network more generalizable and versatile. By normalizing the pixel values, we hoped to decrease bias in our network training procedure and increase the network's learning rate. Post transformations, all the image tensor shapes were [3,64,64]. We chose an image size of 64 instead of 100 for the GANs training to make it computationally lighter, easier in terms of using the standard scaling up of the latent vector dimension of 100 to the images generated by the generator.

The GANs model was inspired by the Deep Convolution GAN architecture - The job of the generator is to spawn 'fake' images that look like the training images. The job of the discriminator is to look at an image and output whether or not it is a real training image or a fake image from the generator. During training, the generator is constantly trying to outsmart the discriminator by generating better and better fakes, while the discriminator is working to become a better detective and correctly classify the real and fake images. The equilibrium of this game is when the generator is generating perfect fakes that look as if they came directly from the training data, and the discriminator is left to always guess at 50% confidence that the generator output is real or fake. A DCGAN is a direct extension of the GAN described above, except that it explicitly uses convolutional and convolutional-transpose layers in the discriminator and generator, respectively. It was first described by Radford et. al. in the paper Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. The discriminator is made up of strided convolution layers, batch norm layers, and LeakyReLU activations. The input is a 3x64x64 input image and the output is a scalar probability that the input is from the real data distribution. The generator is comprised of convolutional-transpose layers, batch norm layers, and ReLU activations. The input is a latent vector, z, that is drawn from a standard normal distribution and the output is a 3x64x64 RGB image. The strided conv-transpose layers allow the latent vector to be transformed into a volume with the same shape as an image.

The DCGAN model consists of two parts - a generator (deeply convoluted) and a discriminator which acts like a CNN-based classifier. The generator, G, is designed to map the latent space vector (z) to data-space. Since our data are images, converting z to data-space means ultimately creating a RGB image with the same size as the training images (i.e., 3x64x64). In practice, this is accomplished through a series of strided two-dimensional convolutional transpose layers, each paired with a 2d batch norm layer and a Relu activation. The output of the generator is fed through a Tanh function to return it to the input data range of [1,1]. This can be illustrated as below.[14]

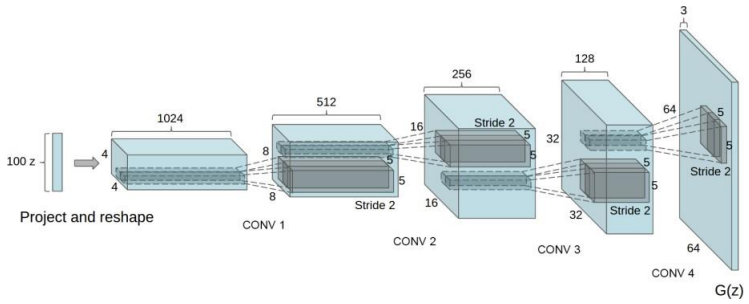

Figure 4: Deeply Convoluted Generator of the GANs Model

The discriminator is a binary classification network that takes an image as input and outputs a scalar probability that the input image is real (as opposed to fake). Here, D takes a 3x64x64 input image, processes it through a series of Conv2d, BatchNorm2d, and LeakyReLU layers, and outputs the final probability through a Sigmoid activation function. The implementation of the GAN pipeline based on the open-source code.[14]

The hyper-parameter tuning during the GANs training were batch-sizes based on the individual classes, smaller batch-sizes were easier to train and gave better quality results. The learning rate was kept small to make the model learn slowly but well, and mode-collapse was reduced by normalizing and transforming the images.

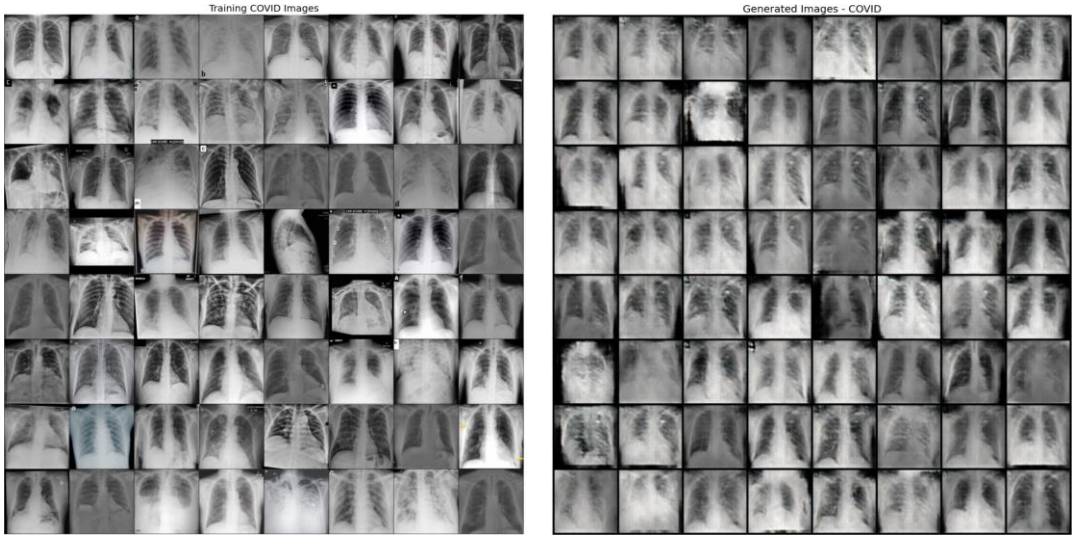

Figure 5: Training vs Generated Images for the COVID-19 Class using GANs

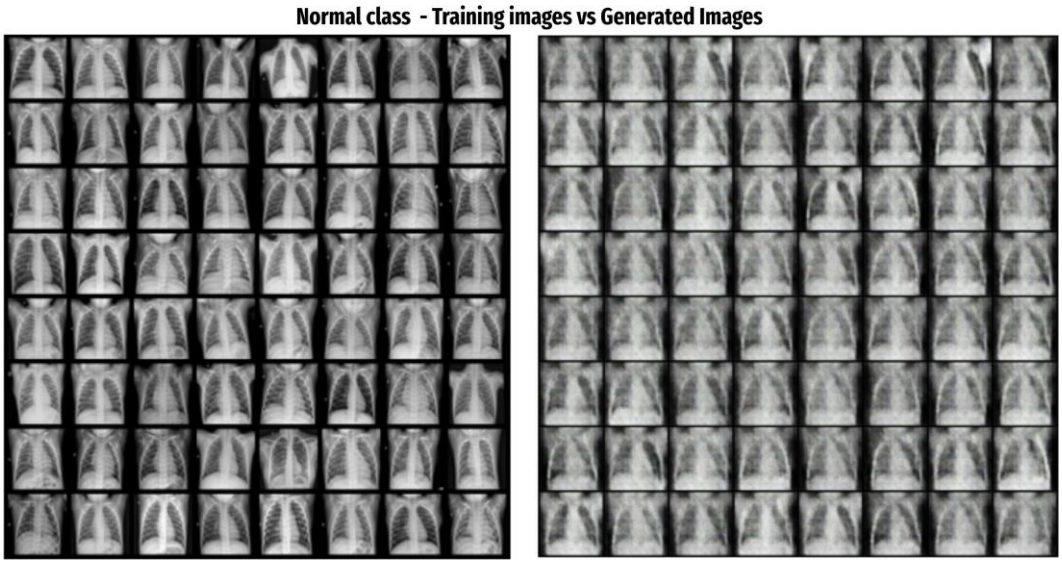

Figure 6: Training vs Generated Images for the 'Normal' Class using GANs

6.3 The hybrid VAE-GAN Model

Before implementing the VAE-GAN, the training and testing data sets of the chest x-ray images underwent various transformations. The first transformation involved converting all images into PIL images. This was necessary as the data set had a wide variety of different image file types. Subsequently all images were resized to 64*64 images (to reduce computational load) and converted to tensors and normalized. The image pixel values were constrained to a range of -1 to 1 using a mean and std of 0.5 across all the channels. By resizing and normalizing all our images we reduced our computational overhead and brought greater consistency to our dataset. The third transformation involved horizontally flipping some of our images. By implementing such a transformation, we hoped to diversify the type/orientation of images the network would train upon, hence making the network more generalizable and versatile. By normalizing the pixel values, we hoped to decrease bias in our network training procedure and increase the network's learning rate. Post transformations, all the image tensor shapes were [3,64,64].

The model consists of an encoder, a generator and a discriminator. The encoder is a RESNET-18 model and is trained with a KL-Divergence loss to ensure the latent space vector **z** is close to the standard normal distribution. The model consists of a pre-trained RESNET-18 with average pooling and two fully connected layers with 512 nodes each. The generator and discriminator models are inspired from DCGAN. The discriminator consists of Conv2D, BatchNorm and LeakyReLU layers. The generator consists of Transpose Conv2D layers with BatchNorm and and ReLU layers. The discriminator uses the standard Binary Cross Entropy loss, while the generator uses a heuristic non-saturating loss function. The combination of encoder and generator is further trained using the reconstruction loss and the discriminator's feature matching loss (Pixel wise XELoss). The combination is of the form of a weighted function, whose weights are hyperparameters and are varied to focus on specific module losses. The hyperparameter tuned for training was input batch size and the weights of the loss functions. The learning rate was set to 2e-4.

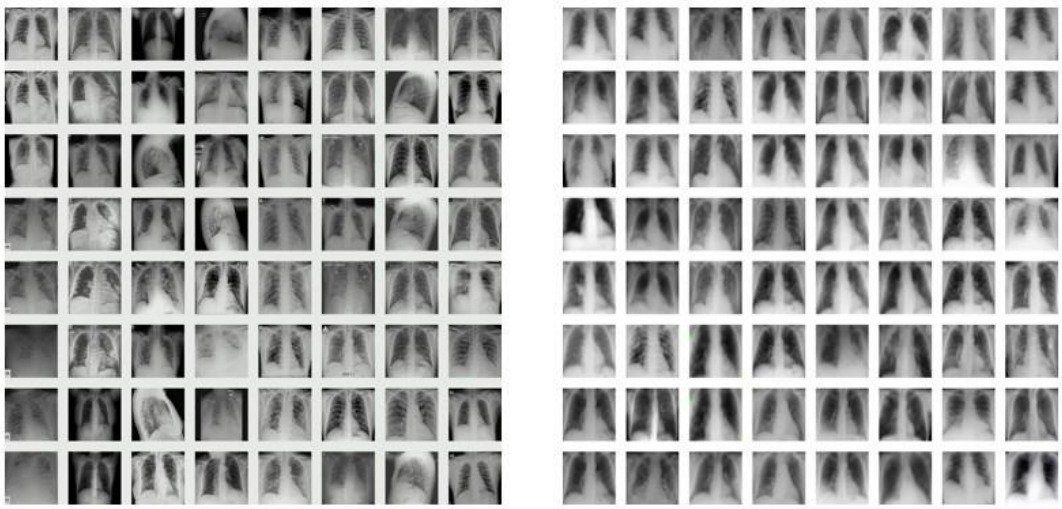

Figure 7: Training vs Generated Images for the COVID-19 Class using VAE-GAN

7 Results

7.1 Evaluation of Baseline Classifiers

Table 1: Baseline model performance comparison for various CNN-based classifiers

Baseline Classifier	Validation Accuracy
Vanilla ResNet18	77.78 %
Vanilla ResNet34	88.89 %
Simplified ResNet18	74.07 %
Inverted Bottleneck ResNet	74.00 %
VGG-16	75.00 %

From the above table, it is observed that **ResNet34 gave the best performance by early stopping just before it started to overfit with an accuracy near 89 % followed by ResNet18 that gave an accuracy of 78 % while training both to around 10 epochs.** Each epoch took roughly around 65

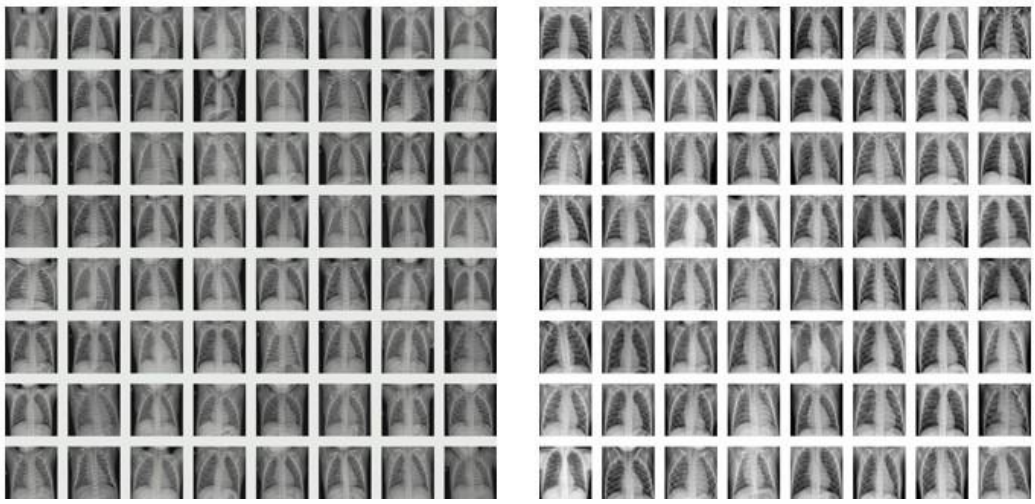

Figure 8: Training vs Generated Images for the Normal Class using VAE-GAN

seconds due to the small size of the data set. The training accuracy of the simplified ResNet reached 84 % after 7 epochs, while the testing accuracy stalled at 74 % for the same number of epochs. The training and validation accuracy continuously oscillated for the Inverted BottleNeck, while the model was being trained. The highest validation accuracy score of 74 percent was achieved at epoch 6 while the highest training accuracy score of 78 percent was achieved at epoch 10. With the VGG-16 Classifier, a lower accuracy of 74% for achieved. This diverging trend may imply the network is over-fitting towards the end of its training period.

7.2 Evaluation of the Image Quality of the Generated Images

Commenting on the images generated by all the three models, we observed **that the performance of the GANs model is better than that of the VAE model, and the hybrid VAE-GANs model outperforms both the individual GANs and VAE model.** VAE model produces grainy images and we do observe some mode collapse in the VAE model, mainly because of the element-wise metrics used by the VAE model. Taking a step further to the GANs model, out generated images are better than that generated in the previous literature mainly because of the image normalization added along with the convolutional layers within that boosted the training quality. As for the combined model, it performs the best by mitigating the training-instability of the GANs model and the poor performance of the VAE model. This is explained further in the next subsection.

7.3 Evaluation of ResNet34 on Augmented Data

The table below summarizes the performance of the ResNet 34 network when the classifier was trained on augmented data sets originating from specific generative models. The metric measuring the performance of ResNet 34 is classification accuracy of the validation data set.

Table 2: ResNet 34 network performance- Augmented Data set

Generative Network that Augmented Dataset	Validation Accuracy
Original Dataset with No Augmentation	88.89%
Augmented with VAE-generated Images	88.88%
Augmented with GANS-generated Images	92.29%
Augmented with Hybrid VAE-GANs-generated Images	94.26%

From the above numbers, it is evident that data-augmentation using generative models definitely boosted the performance of the classifier in case of the GANs based images than with the VAE generated images as the quality of the images generated by the GANs model is better and more informative enough that the same ResNet34 can distinguish between the classes. **The Hybrid VAE-GAN model produced the best quality of the images with a 94.26% accuracy because of the stability VAEs bring to the GANs model.**

The poor performance of the VAEs compared to GANs is because of the fact, VAEs essentially use element-wise metrics like the squared error, which might be simple but not very suitable for image data, as they do not model the properties of human visual perception. This clearly explains why VAEs fail to generate good quality images, we need a higher level representation of the images and sufficiently invariant representation of the images. By combining VAEs with GANs, we can eliminate these element-wise metrics and utilize the feature-based metrics that's used in a GANs Decoder based on the features generated by the VAEs encoder. We collapse the VAE decoder and the GAN generator into one by letting them share parameters and training them jointly. This in turn makes the VAE-GANs model outperform the VAE model. As for the performance of the VAE-GAN vs ACGAN, just by looking at the images generated in the paper, and the images we've generated with DCGAN alone and the combined model, we clearly see that VAE-GAN out-performed the individual GANs model both in terms of image quality and classifier performance. One of the things, we leave for the future work is to explore more classifiers and see if they perform well with our augmented data.

As to the overfitting, we enforced early-stopping while training to prevent overfitting while running the classifiers on the augmented data. So the augmented dataset in general helped prevent overfitting, and using patience parameters on top of that, favored the results even more.

8 Conclusion

In conclusion, we have developed four distinct classification baseline models. Three out of the four models are variants of the ResNet model and the fourth model is based upon the VGG model. The best performing baseline model is ResNet34, with an accuracy score of 88.88 %. All four models had a tendency to over fit the training data and are outputting accuracy scores inferior to scores outlined in the literature review section. We hypothesize that the 'tendency to over fit' is caused by a small and imbalanced training data set. This justifies our problem statement that there is a need for a suitable data augmentation technique to expand and balance the data set. In order to overcome this roadblock, we devised and developed a hybrid VAE and GANS based generative neural network that synthesizes lung x-ray images and in turn augments the training data set being used for the CNN classification/diagnostic model. By applying our unique generative model framework, we achieved an increased accuracy score of the original CNN classification model to 94%.

9 Future Work

Our innovative approach testifies to the possibility of using Deep Learning in a clinical setting, to both help ease off the load of healthcare workers and improve current diagnostic tools. Our improved accuracy of 94% for the novel VAE-GANs model can still be increased by further architecture development. As more data sets for COVID-19 chest X-rays become available, the proposed model can be tested to further provide information on the reliability of its current accuracy scores. As those results become available, our model could prove important in creating a novel method for the diagnosis of COVID-19, as well as potentially providing a blueprint for the development of X-ray diagnostic tools for multiple diseases.

References

[1] A. Waheed, M. Goyal, D. Gupta, A. Khanna, F. Al-Turjman and P. R. Pinheiro, "CovidGAN: Data Augmentation Using Auxiliary Classifier GAN for Improved Covid-19 Detection," in IEEE Access, vol. 8, pp. 91916-91923, 2020, doi: 10.1109/ACCESS.2020.2994762.

[2] Jacobi A, Chung M, Bernheim A, Eber C. Portable chest X-ray in coronavirus disease-19 (COVID-19): A pictorial review. Clin Imaging. 2020;64:35-42. doi:10.1016/j.clinimag.2020.04.001

[3] A. Larsen, S. Sønderby, H. Larochelle, O. Winther. Autoencoding beyond pixels using a learned similarity metric. arXiv, 2016. doi:1512.09300

[4] Y. Wang, C. Dong, Y. Hu, C. Li, Q. Ren, X. Zhang, H. Shi, and M. Zhou, "Temporal changes of CT findings in 90 patients with COVID-19 pneumonia: A longitudinal study," Radiology, Mar. 2020, Art. no. 200843. [Online]. Available: http://10.1148/radiol.2020200843

[5] D. Kermany, D. Kermany, M. and Dentistry, U. of C.-S. Diego, and M. G. Kang Zhang, "Labeled Optical Coherence Tomography (OCT) and Chest X-Ray Images for Classification," 2018. [Online]. Available: https://data.mendeley.com/datasets/rscbjbr9sj/2.

[6] O. Stephen, M. Sain, U. J. Maduh, and D. U. Jeong, "An Efficient Deep Learning Approach to Pneumonia Classification in Healthcare," J. Healthc. Eng., 2019.

[7] A. A. Saraiva et al., "Classification of images of childhood pneumonia using convolutional neural networks," BIOIMAGING 2019 - 6th Int. Conf. Bioimaging, Proceedings; Part 12th Int. Jt. Conf. Biomed. Eng. Syst. Technol. BIOSTEC 2019, pp. 112–119, 2019.

[8] G. Liang and L. Zheng, "A transfer learning method with deep residual network for pediatric pneumonia diagnosis," Comput. Methods Programs Biomed., 2020

[9] D. Zhao, D. Zhu, J. Lu, Y. Luo, and G. Zhang, "Synthetic medical images using FBGAN for improved lung nodules classification by multi-scale VGG16," Symmetry, vol. 10, no. 10, p. 519, 2018.

[10] W. Dai, J. Doyle, X. Liang, H. Zhang, N. Dong, Y. Li, and E. P. Xing, "SCAN: Structure correcting adversarial network for chest X-rays organ segmentation," 2017, arXiv:1703.08770. [Online]. Available: https://arxiv.org/abs/1703.08770

[11] D. Nie, R. Trullo, J. Lian, C. Petitjean, S. Ruan, Q. Wang, and D. Shen, "Medical image synthesis with context-aware generative adversarial networks," in Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent., vol. 10435, 2017, pp. 417–425.

[12] A. Beers, J. Brown, K. Chang, J. P. Campbell, S. Ostmo, M. F. Chiang, and J. Kalpathy-Cramer, "High-resolution medical image synthesis using progressively grown generative adversarial networks," 2018, arXiv:1805.03144. [Online]. Available: http://arxiv.org/abs/1805. 03144

[13] A. Boesen & L. Larsen & S. K. Sønderby & H. Larochelle & O. Winther (2016) Autoencoding beyond pixels using a learned similarity metric. *Proceedings of The 33rd International Conference on Machine Learning, PMLR 48:1558-1566*

[14] https://pytorch.org/tutorials/beginner/dcgan*faces*tutorial.html