CS221 Spring 2021 Project Proposal

SUNet ID: [rpatwari, varoly]
Names: [Rajeev Patwari, Varo Ly]

Introduction

Supervised learning algorithms require adequate amount of data in order to learn the patterns and behaviors of the observed data. In the case of sparsity in availability of data, data-augmentation is one of the key methods to generate synthetic data to train the network. We would like to address this problem by using Generative Adversarial Network (GAN) as a synthetic data generator. We propose to use this synthetic data in order retrain the classifier to perform better.

In particular, this project aims to train a classifier network on the Colorectal Histology dataset and show that the addition of synthetic data using GAN generated images improves the accuracy of the classifier. In addition, to compare the effectiveness of GAN generated images, we will also generate new images based on standard image manipulation techniques, such as blur/interpolation/noise-insertion, etc., on the existing images and show that the GAN generated images improved the classifier accuracy more than the image manipulation generated images.

Literature Review

Amongst the references as our starting point for our work. The work represented by [3] tries to address similar task, but on MNIST dataset. The GAN used in this work is fully connected network, not a CNN. Since the images are black and white, the simple fully connected layer based GAN suffices this paper's need. The work represented by [5] also addresses similar work, but again, on MNIST dataset. These provide a good motivation, but are not a baseline for our work. Our work involves dataset which has a higher dimension of data distribution, i.e. colored images. Thus, our work requires a GAN based on CNNs capable of extracting and generating information from deep channels.

The reference, [1], is an excellent inspiration for our work, but uses deep Resnets, required a larger compute power for training. For example, the discriminator in this model is a DenseNet and generator is 8 blocks of ResNet with 4 CNNs in each ResNet block. This model has excellent performance as shown in the Omniglot as well as VGG face dataset. However, given compute resources available to us, we are unable to recreate the exact complex model.

The reference [6] is a good reference but they focus on higher dimention datasets such as Breast cancer dataset from University of Wisconsin, with as many as 32 features. However, their GAN implemntation is relatively simple with less than 5 layers of convolutions, with

a maximum of 1024 channels. THis was a good reference for us. We also found a [2] that partially implemented the work similar to [6], but on Cifar-10 dataset. This became the baseline for GAN.

However, we have not found a single reference which is able to generate dataset, along with labels for synthetic data. We aim to solve this problem either by developing a novel GAN or by ensemble of GANs for each class. We have a baseline implemented with ensemble of GANs per data class as described below. We aim to publish our work after our project is done.

Dataset

This project uses the colorectal histology dataset [4] which is available through the TensorFlow dataset catalog. The dataset consists of 5000 150x150 pixel color images distributed evenly into 8 classes, 625 images each, as shown in Figure 1. 80% of the dataset was used for classifier training, 10% for validation, and the remaining 10% for testing.

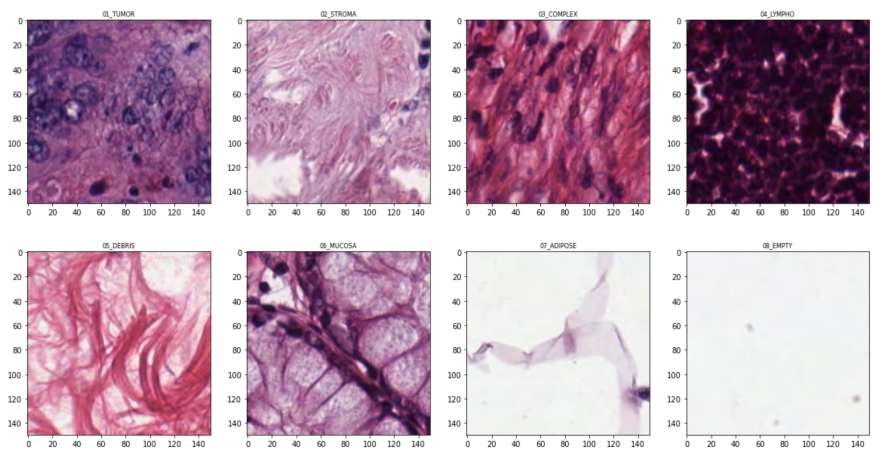


Figure 1: Samples of Colorectal Histology dataset for each Label

Baseline

Classifier

The TensorFlow Keras built-in VGG16 classifier, which is implemented as a multi-layer CNN, was used as the baseline classifier of the histology dataset as the classifier is not the main goal of this project. The VGG16 model was implemented with the following parameters:

- weights: random initialization, not Imagenet pre-trained
- optimizer: SGD

- loss: categorical cross entropy
- metric: categorical accuracy

The VGG16 classifier was trained on batches of 32 images and after 15 epochs, the validation accuracy fluctuated around 50% and the classifier achieved 54% accuracy on the withheld test dataset (10% of the full dataset).

For baseline comparison of GAN generated images we will generate synthetic data by modifying real input images using techniques such as blur/interpolation/noise-insertion, etc. Thus, the baseline can provide an image that is a distorted version of real data image, not generated by GAN but hand-generated. This can be used as part of the training dataset which we will then compare with our GAN augmented training dataset to benchmark the benefits of GAN generated images to improve the VGG16 classifier accuracy.

GAN

A good baseline starting point was obtained from [2]. The author has implemented a GAN for Cifar-10 dataset, without any distinction for individual class of the images being generated synthetically. In other words, the author has provided entire dataset to the GAN, resulting in GAN generating a dataset that resembles similar distribution of original data, but no clear adherence to which class the generated data belongs to.

In our flow, we categorized data in to sub directories, each directory with a dataset belonging to distinct data such. So, we had 8 directories with 8 classes of data as per our dataset. We trained GAN individually 8 different times, thereby producing 8 different GAN training parameters, each GAN trained to fit and generate synthetic data according to the distribution of the dataset. This enabled us to generated unique synthetic data that can be labelled into respective categories.

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 150, 150, 64)	1792
leaky_re_lu (LeakyReLU)	(None, 150, 150, 64)	0
conv2d_1 (Conv2D)	(None, 75, 75, 128)	73856
leaky_re_lu_1 (LeakyReLU)	(None, 75, 75, 128)	0
conv2d_2 (Conv2D)	(None, 38, 38, 128)	147584
leaky_re_lu_2 (LeakyReLU)	(None, 38, 38, 128)	0
conv2d_3 (Conv2D)	(None, 19, 19, 256)	295168
leaky_re_lu_3 (LeakyReLU)	(None, 19, 19, 256)	0
flatten (Flatten)	(None, 92416)	0
dropout (Dropout)	(None, 92416)	0
dense (Dense)	(None, 1)	92417

Total params: 610,817
Trainable params: 610,817
Non-trainable params: 0

(a) Discriminator

Model: "sequential_3"

Layer (type)	Output Shape	Param #
dense_2 (Dense)	(None, 92416)	9334016
leaky_re_lu_8 (LeakyReLU)	(None, 92416)	0
reshape_1 (Reshape)	(None, 19, 19, 256)	0
conv2d_transpose_3 (Conv2DTr	(None, 38, 38, 128)	524416
leaky_re_lu_9 (LeakyReLU)	(None, 38, 38, 128)	0
conv2d_transpose_4 (Conv2DTr	(None, 76, 76, 128)	262272
leaky_re_lu_10 (LeakyReLU)	(None, 76, 76, 128)	0
conv2d_transpose_5 (Conv2DTr	(None, 152, 152, 128)	262272
leaky_re_lu_11 (LeakyReLU)	(None, 152, 152, 128)	0
conv2d_5 (Conv2D)	(None, 152, 152, 3)	3459

Total params: 10,386,435
Trainable params: 10,386,435
Non-trainable params: 0

(b) Generator

Figure 2: GAN model structure

Prior to training of the GAN, discriminator was trained to distinguish between real and fake images, fake images being random noise from latent vector. During the training of the generator, the weight update for discriminator section has been disabled. This ensured generator to be trained as intended.

Main Approach

The GAN consists of a discriminator and generator neural network that are trained against each other. The discriminator is implemented as a 4 layer CNN with leaky RELU activation, and then a fully connected layer followed by a dropout layer. The generator is similarly implemented with 4 CNN layers with the final layer having a tanh activation.

Since the dataset had 8 classification labels, we implemented 8 different GANs in order to generate additional images for each class. A single GAN neural network is depicting in 2.

Evaluation metrics

For the GAN, we use cross entropy loss as an optimization metric, i.e.

Discriminator Loss: **maximize** $logD(x) + log(1–D(G(z)))$

Generator Loss: **minimize** $log(1 - D(G(z))$

In general, it is harder to ascertain the success of GAN based on the loss function of the discriminator and generator. As the generator improves and the discriminator is unable to detect and classify the generated image as a fake image, the generator then learns based on spurious data from the discriminator. As such we saved the model at each epoch so that we were able to find the best image that the generator created based on human input.

Our evaluation metric for the VGG16 classifier is categorical accuracy. Improvement of this classification accuracy when the model is trained with the addition of the GAN generated images also provides metric for the success of the GAN.

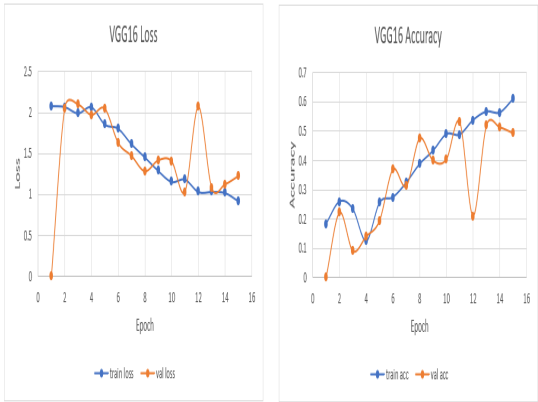
Results and Analysis

The classifier and GAN training loss is shown in the **??**

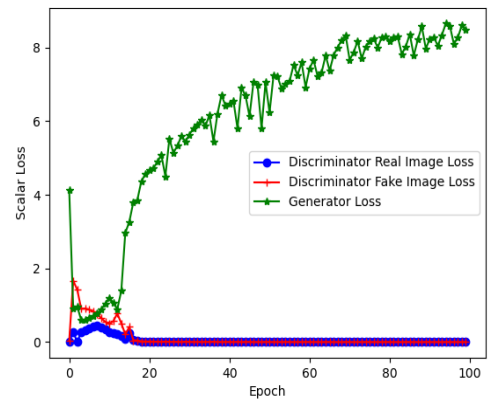
The synthetic data is shown in the appendix.

Future Work

Implement alternate synthetic data generation by augmenting the real image data using standard image modification techniques such as blur/interpolation/noise-insertion [7] and



(a) VGG16 classifier loss



(b) GAN training loss vs epochs - Discriminator and Generator

Figure 3: Classifier and GAN training vs epochs

compare to GAN generated images in improving classifier accuracy.

Study latent space values and their impact on generating good GAN images.

References

[1] Antreas Antoniou, Amos Storkey, and Harrison Edwards. *Data Augmentation Generative Adversarial Networks*. 2018. arXiv: `1711.04340 [stat.ML]`.

[2] Jason Brownlee. *Jason Brownlee: How to develop GAN for Cifar-10*. URL: `https://machinelearningmastery.com/how-to-develop-a-generative-adversarial-network-for-a-cifar-10-small-object-photographs-from-scratch/`.

[3] Prateik Harwalker. *Prateik Harwalker: CS230 GAN MNIST*. URL: `http://cs230.stanford.edu/projects_fall_2019/reports/26256833.pdf`.

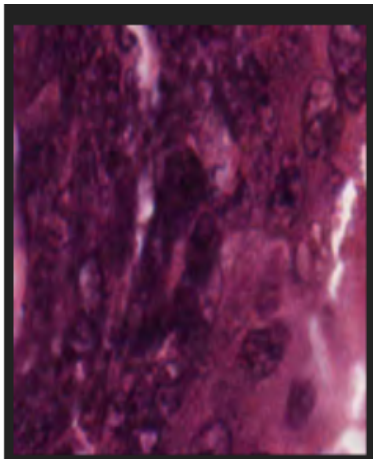
[4] Jakob Nikolas Kather et al. "Multi-class texture analysis in colorectal cancer histology". In: *Scientific reports* 6 (2016), p. 27988.

[5] David Liu and N. Hu. "GAN-Based Image Data Augmentation". In: 2020.

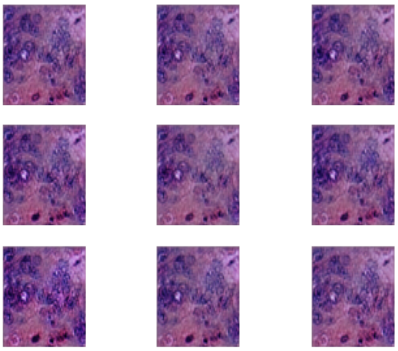
[6] Fabio Henrique Kiyoiti dos Santos Tanaka and C. Aranha. "Data Augmentation Using GANs". In: *ArXiv* abs/1904.09135 (2019).

[7] Tensorflow. *Tensorflow Keras API*. URL: `https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/ImageDataGenerator`.

Appendix

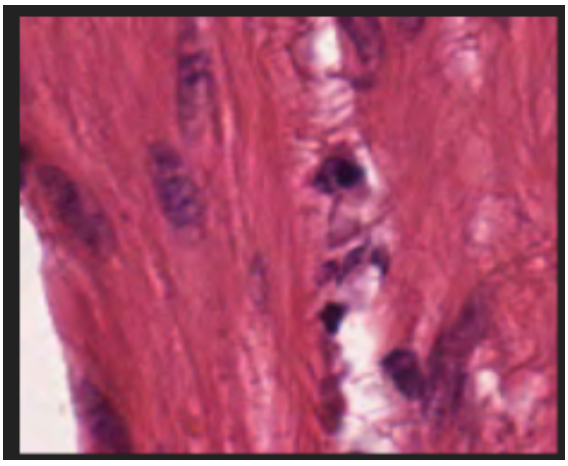


(a) Original

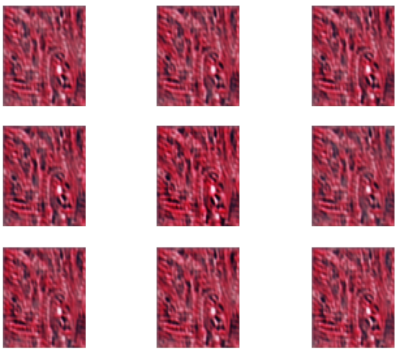


(b) Synthetic generated from GAN

Figure 4: GAN results on dataset for Class-1

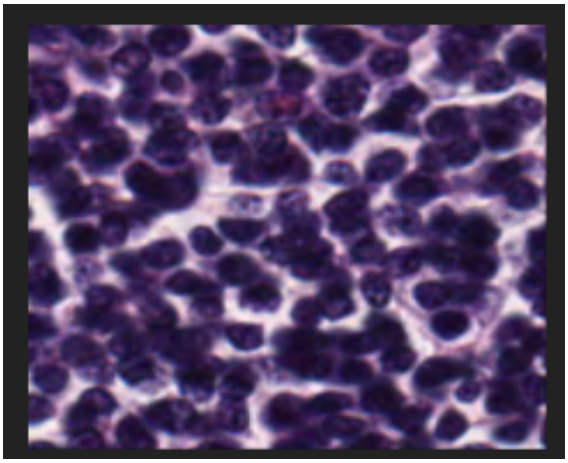


(a) Original

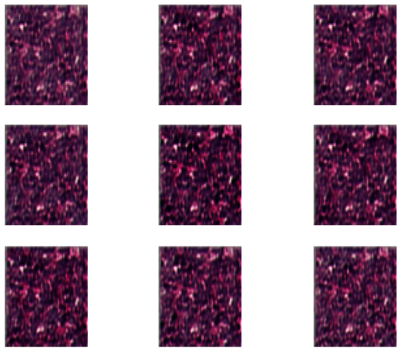


(b) Synthetic generated from GAN

Figure 5: GAN results on Class-2 image



(a) Original



(b) Synthetic generated from GAN

Figure 6: GAN results on Class-4 image