CS230

Evaluating the Efficacy of Data Augmentation Using Generative Adversarial Networks For Identification of Leukemia Cells

Aditi Gaur
Computer Science
agaur@stanford.edu

Peter Nsaka
Computer Science
nsaka@stanford.edu

Victor Warlop Piers de Raveschoot
Materials Science
vwarlop@stanford.edu

Louise Zhuang
Electrical Engineering
llz@stanford.edu

Abstract

The classification of Leukemia cells using machine learning models are poised to have great benefits in healthcare such as reduced diagnosis time and improved efficiency. However, the lack of labeled training data sets due to hospital patient privacy rules and difficulty of obtaining professionally labeled images makes it nearly impossible to get access to sufficiently large datasets needed to train such models. In this project, we propose a method for novel data generation of cell images by leveraging the Wasserstein GAN with Gradient Penalty (WGAN-GP). Additionally, our technique proves that the accuracy of a leukemia cell binary classifier which was originally only trained on real cells can be improved when training data is augmented with generated images from our GAN generator .

1 Introduction

Leukemia is a cancer of the blood and bone marrow, of which early diagnosis is crucial for the recovery of patients. Several machine learning models have been trained over the years to classify blood smears [1,2]. However, one commonality amongst all models is the use of data augmentation techniques (such as image rotation) to make up for the lack of data. These models nevertheless still experience difficulties in classifying cells in more realistic environments where cells may have different backgrounds and may appear partially cropped [3].

We would like to explore whether generating augmented images using a Generative Adversarial Network (GAN) to augment our training dataset could improve the efficacy of such a model. This data augmentation technique for biological systems could be beneficial for a large range of cellular classification problems.

2 Dataset

We utilize a dataset containing 260 zoomed-in images of blood smears (ALL-IDB2) [2]. Of these blood smears, 130 are positive for leukemia and 130 of them are healthy, which results in having exactly balanced classes. On the images, one white blood cell (purple) is surrounded by approximately 5 red blood cells (pink). Given that this amount of data was not sufficient to train a model, we augmented our training data by rotating and flipping the images, thus yielding about 3,000 total images on which to train our GAN model. Reducing the size of the images to (128,128,3) and even (64,64,3) reduced the computational cost significantly but (256,256,3) images would allow the generated images to have significantly higher quality.

3 Methods

3.1 Hardware

Our models were trained on TPUs (Tensor Processing Units). TPUs are Google's custom-developed application-specific integrated circuits (ASICs) used to accelerate machine learning workloads that involve dense vector and matrix calculations [4]. The TPU v2-8 has 8 GB per core and 8 cores per device, with 64 GB of high bandwidth memory and 180 teraflops of compute performance [4,5]. By running on TPUs rather than GPUs, we were able to accelerate the performance of our training from days to hours.

CS230: Deep Learning, Spring 2021, Stanford University, CA. (LateX template borrowed from NIPS 2017.)

3.2 Training and selecting a GAN model

As we explored the existing literature about GANs, it became clear that no single GAN architecture performed best for all applications. Although it was not possible to implement all types of GANs available, we explored a couple of them with varying levels of success with regards to the quality of images generated. Our Deep Convolutional GAN (DC-GAN) model was implemented based an architecture presented in a demo done by the Tensorflow authors [6]. Our Wasserstein GAN with Gradient Penalty (WGAN-GP) implementation was based off of the work done by Nain [7]. Our InfoGAN model was implemented based on the work of research presented in [8].

While training our models, we utilized both Adam [9] and RMSprop [10] optimizers. We utilized Adam while training the DC model and InfoGAN, but tried both Adam and RMSprop while training the WGAN-GP since published literature claimed that momentum could destabilize WGAN training [11]. To search for our optimal learning rate, we maintained a lower search space for our generator, exponentially distributed between 10^{-5} and 10^{-3}, and maintained a slightly higher range for our discriminator, exponentially distributed between 10^{-4} and 10^{-2}. While tuning our parameters further, it quickly became clear to us that we were seeing much better performance from the WGAN-GP than the DC-GAN or InfoGAN. For this reason, we will continue to discuss only the WGAN-GP, henceforth referred to as our model, in more detail.

3.3 Optimizing our WGAN-GP model

While training our model, we found that utilizing a learning rate of 2×10^{-4} for both our discriminator (D) and generator (G) gave us the most stable performance and allowed our model to converge in a timely fashion. Additionally, upon trying both RMSprop and Adam optimizers and training the network for the same number of epochs, we did not see a significant difference in the quality of the resulting images. Thus, we opted to use Adam. Our loss functions for the discriminator and generator in our model can be summarized as follows:

$$\mathcal{L}_{\mathcal{D}} = D(x) - D(G(z)) + 10 \cdot (\left\|\nabla_{\hat{x}} D(\hat{x})\right\|_2 - 1)^2$$
$$\mathcal{L}_{\mathcal{G}} = D(G(z))$$

where z is the input random noise vector, x is the input image, ϵ is random between 0 and 1 and and $\hat{x}$:

$$\hat{x} = \epsilon \cdot x + (1 - \epsilon) \cdot G(z) \tag{1}$$

Note that the constant 10 in equation 1 is the value we chose for our gradient penalty, or λ as it is referred to in literature about WGAN-GP. Our discriminator uses the Earth-Mover (Wasserstein) distance for an improved cost function of the model, where the discriminator must satisfy the Lipschitz constraint [12]. This cost function is smoother and does not directly link the generator to the discriminator performance, which allows us to better maintain gradients even when the discriminator performs well. Thus our generator can continue learning when a normal GAN would not. Introducing a penalty on the gradient norm (gradient penalty) of the WGAN improved our performance even further, as this served as an alternative to gradient clipping and kept our gradients within an acceptable range to satisfy the Lipschitz constraint [13].

After tuning these hyperparameters and many more, we switched our attention to calibrating the number of extra steps we trained the discriminator. Since our critic is evaluating a continuous measure of quality of the produced image, training the critic for more steps as compared to the generator is common practice in a WGAN. Initially, we were attempting to train the discriminator 8 times for every time we trained the generator. This resulted in the discriminator becoming too skilled too quickly, and thus prohibiting the generator from learning. We found that lowering this number to 5 allowed the discriminator to get skilled enough such that the generator and discriminator could continue learning from each other. The architecture, adapted from [7] and [14] of the WGAN-GP can be found in Figure 1.

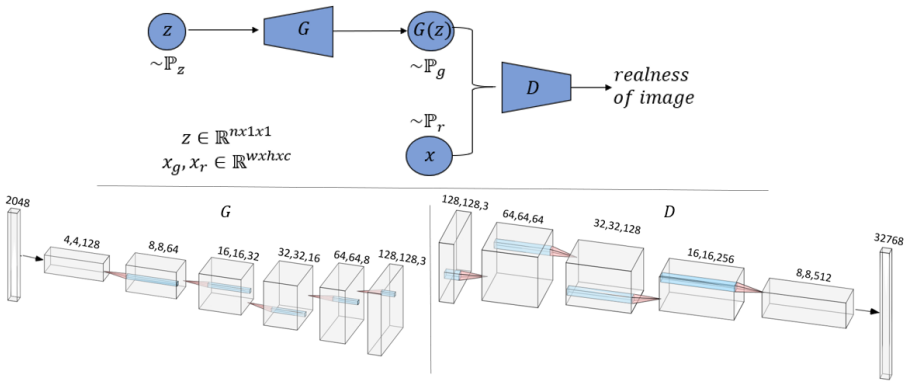


Figure 1: (width, height, channels). D uses convolution whilst G uses deconvolution. The kernel size is (5,5) and stride (2,2) for D. The kernel size is (3,3) and stride (1,1) for G. The padding is 'same'. LeakyRelu is used in both cases (G ends on a tanh function), with α of 0.2. A dropout of 0.3 is used for D. G uses batch normalisation whilst D does not.

3.4 Validating the accuracy of our GANs generated cell images

To evaluate the efficacy of the generated images, we used several binary classifiers. First, to check that the GAN-generated images of the healthy cells and leukemia cells were actually different, the classifiers were first trained to classify real healthy and leukemia cells and then used to classify the generated images. For successful generation of healthy and leukemia cell images, the classification accuracy would be similar or greater than the accuracy for the real test images. After confirming successful GAN performance, the GAN-generated images were then used for training the classifiers to compare the classification accuracy on a test set of real cell images when using only real images and using both real and 400 generated images to train the classifiers. The real images were split 80%:20% between training and testing. The classifiers that were used were a convolutional neural network, logistic regression, and support vector machine (SVM). The convolutional neural network used the VGG16 architecture with pre-trained weights from the ImageNet dataset. During the training process, only the weights for the last layer were adjusted for 40 epochs, and the learning rate was tuned to 0.00001. However, since the network had many layers, the model could overfit to our small dataset, so the two linear classifiers were used to provide a more realistic measure of accuracy. The logistic regression model had a regularization parameter tuned to $\lambda = 1$ and used 5000 maximum iterations. The SVM used a radial basis function kernel and a regularization parameter of $\lambda = 1$. The kernel coefficient is calculated as $\gamma = \frac{1}{1875 \cdot Var(traindata)}$.

4 Results and Analysis

Our model was trained on two different distributions of images: healthy blood smears and leukemia positive blood smears. Thus, we had two models that each generated healthy and diseased images respectively. Hand-picked samples of our generated images are contrasted against similar real cell images in Figure 2 and Figure 3 in order to point out the similarity of features. The 8 images of healthy and diseased cells do not represent the complete real distribution of the ALL-IDB2 dataset or our generated dataset.

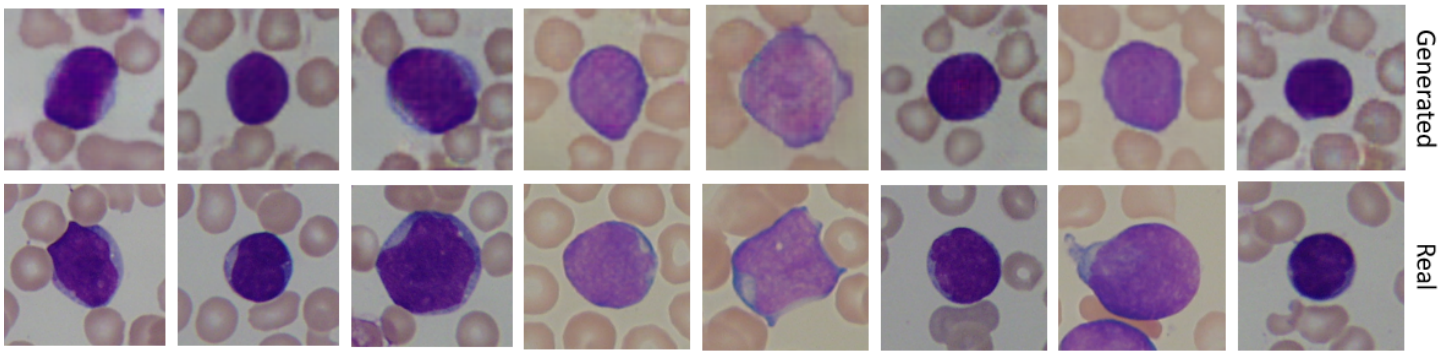


Figure 2: Leukemia Images: Real images are compared to generated images.

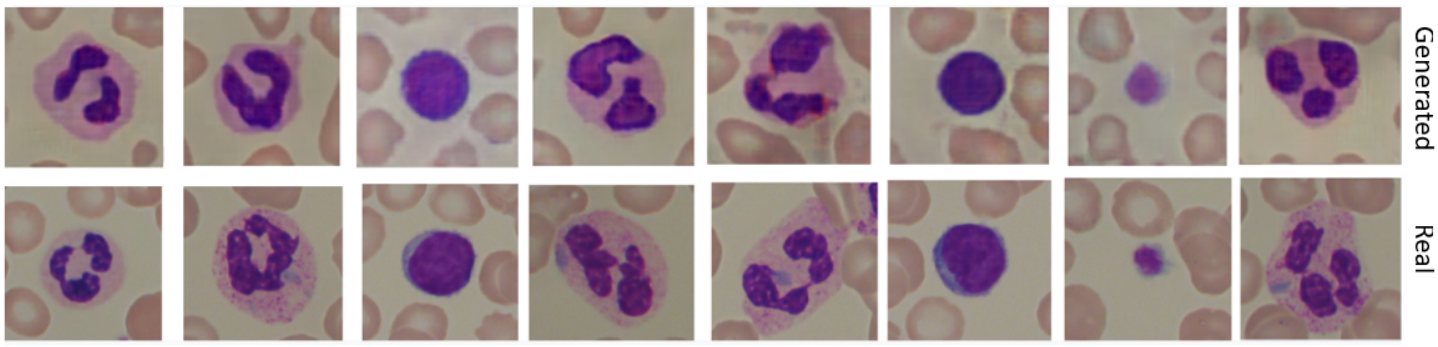


Figure 3: Healthy Images: Real images are compared to generated images.

A larger variety of our generated images, demonstrating the vast distribution of cells it is able to capture, are contained in subsection 8.1.

We can clearly see that our model captures many types of different healthy and leukemia cells, and that these images are close to indistinguishable to the untrained eye. The model, also seems to be able to generate white blood cells that might suggest differentiation into different types of white blood cells such as neutrophils and eosinophils. However, the model is not yet able to distinguish ultra high details such as intra-cellular granules and the shape of the red blood cells does not consistently form circles. To address this, and other potential solutions to improve our model, we will continue this discussion in the limitations section.

Having obtained generated images from the healthy and leukemia positive distributions, we then proceeded to evaluate them with several binary classifiers. We hand-selected 200 healthy generated images and 200 leukemia blast generated images and the data was augmented with consecutive 45°rotations. We chose to operate with 400 generated images so that we would not overwhelm the real dataset, which only consisted of 260 images.

First, we tried to have a binary classifier label our healthy and leukemia-positive generated images and measure the classification accuracy. The appropriate metrics are reflected in Table 1. Equations outlining these metrics are located in the appendix subsection 8.2. For the logistic regression and SVM models, the classification accuracies on a test set comprised of only generated images are similar or slightly higher than the classification accuracies for a test set of solely real images. This indicates that the generated healthy and leukemia cell images are reasonably different from each other and most likely resemble real healthy and leukemia cell images. The neural network, however, shows a drop in classification accuracy for the generated images, indicating that it may have overfit to the relatively small real image dataset. See Table 2 for accuracies of these models on real test sets.

Table 1: Performance of classifiers on GAN-generated images when trained on only real images.

Neural Network					
	Accuracy	Precision	Recall	F1	Specificity
Real training	72.91%	65.16%	99.02%	78.60%	46.53%
Real + aug. training	80.30%	72.14%	99.02%	83.47%	61.39%
Logistic Regression					
Real training	78.82%	75.65%	85.29%	80.18%	72.28%
Real + aug. training	84.24%	84.31%	84.31%	84.31%	84.16%
SVM					
Real training	83.25%	77.87%	93.14%	84.82%	73.27%
Real + aug. training	90.15%	84.75%	98.04%	90.91%	82.18%

4.1 Evaluating improvements in binary classifier accuracy when augmented using synthetically generated images

Having established that our generated images are representative of the distributions we hope to augment, we proceeded to evaluate whether augmenting our training data with our model-generated images would improve the performance of our binary classifiers on a real test set. The results of our investigation are summarized in Table 2. We used rotations as the traditional augmentation technique for the real images. We can see that the real and augmented image training set generally performed better on a test set of real images than the real data and generated data training set. However, real, augmented, and generated data combined for training consistently performed at or above the best accuracy we obtained for all three binary classifiers. This demonstrates to us that including synthetically generated images can improve our performance above traditional augmentation. It is worth noting that, of all our recorded metrics, our models performed the worst on the specificity score. This indicates that we have a significant number of false positives, which may mean that our classifier models are struggling with identifying negative image cases, but this result may be preferred clinically since false negatives should be prevented more than false positives.

Table 2: Performance of classifiers on real test images given different sets of training data and the same hyperparameters

Neural Network					
	Accuracy	Precision	Recall	F1	Specificity
Real training	90.38%	87.10%	96.43%	91.53%	83.33%
Real + aug. training	100.00%	100.00%	100.00%	100.00%	100.00%
Real + gen. training	98.08%	100.00%	96.43%	98.18%	100.00%
Real + aug. + gen. training	100.00%	100.00%	100.00%	100.00%	100.00%
Logistic Regression					
Real training	73.08%	70.59%	85.71%	77.42%	58.33%
Real + aug. training	80.77%	82.14%	87.14%	82.14%	79.17%
Real + gen. training	78.85%	74.29%	92.86%	82.54%	62.50%
Real + aug. + gen. training	82.69%	82.76%	85.71%	84.21%	79.71%
SVM					
Real training	76.92%	73.53%	89.29%	80.65%	62.50%
Real + aug. training	80.77%	75.00%	96.43%	84.37%	62.50%
Real + gen. training	69.23%	71.43%	71.43%	71.43%	66.67%
Real + aug. + gen. training	84.62%	74.41%	96.43%	87.10%	70.83%

After observing these metrics across our classifiers, we were also interested in whether generated images may not only improve the test accuracy of our model, but also the training speed (i.e. number of epochs required to train the model). We mapped out

the loss function of the neural network classifier by epoch in Figure 4. Using real, augmented, and generated images in the training set resulted in the fastest convergence when training the model.

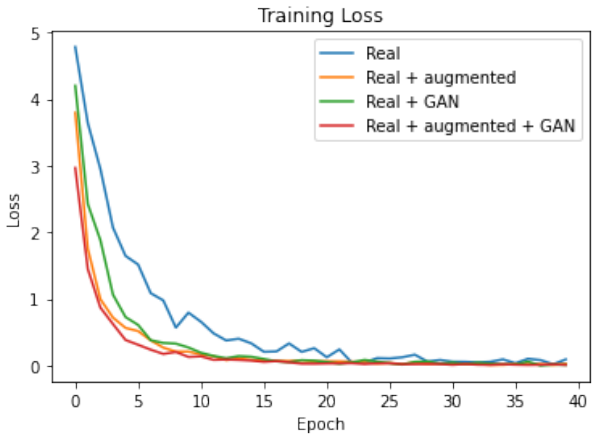


Figure 4: Neural network training loss for different training sets. Using real images, augmentations to the real images, and GAN-generated images led to the lowest loss over training, while using just real images led to the highest training loss.

5 Limitations of Study

Dataset: Based on our discussions with a subject matter expert, leukemia cell images are difficult to find mostly because of patient privacy rules many hospitals follow. While we were able to find a few labeled data sets, on which our GAN model trained pretty well, we believe the quality of our images would have been greatly enhanced by having access to more training data. We had originally trained on only 260 images but performed a few data augmentation techniques such as flipping and rotating which immediately gave us much better results.

Improving the Model Type and Architecture: Even though literature encourages the use of WGAN-GP, it is certainly not considered perfect, and other models should be attempted. One possible improvement would be to combine the advantages of InfoGAN with WGAN-GP, as it would teach the GAN to learn meaningful and interpretable representations of cellular structures [15]. Experimenting with a wider range of CNN architectures could also improve the model.

Exploring More Hyperparameter Tuning: While training our models, we explored hyperparameter values close to those suggested by the authors of the architectures we chose, however, we believe we may be able to get better results by trying a wider range of hyperparameter values.

Better Evaluation Metric: We would have loved to use a quantitative metric, such as SSIM or FID score, to determine how well each model does on a validation set. Note that we did attempt to experiment with implementing an FID score throughout training. However, due to the small size of our real image dataset, these estimates were incredibly noisy and inaccurate. Additionally, given that we are not experts at identifying leukemia cells, another quality metric could be to have a subject matter expert help us evaluate the accuracy of our leukemia cells which will serve as a benchmark on how well our model is performing.

6 Conclusion

In this paper, we determined that we are able to use a GAN model to generate realistic images of leukemia cells and healthy cells. Additionally, in agreement with most existing literature, after trying out multiple GANs models such as DC-GAN, WGAN-GP, and InfoGAN, we found that WGAN-GP gave us the best generated images.

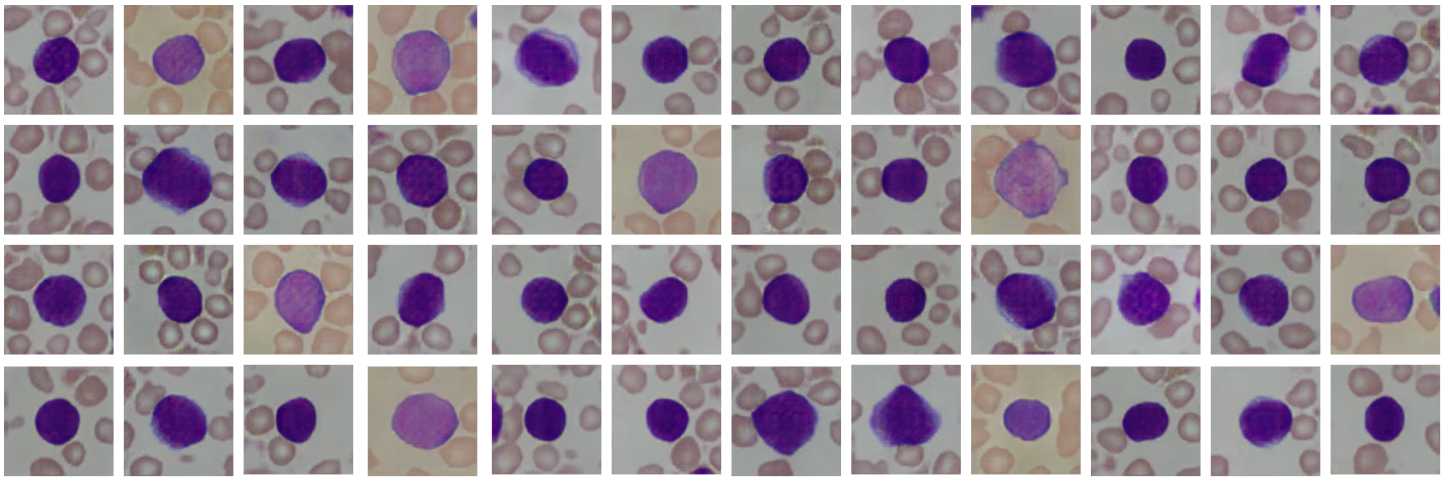
Secondly, we confirmed that by using generated images from our GANs, we are able to improve the accuracy of leukemia cell classification.

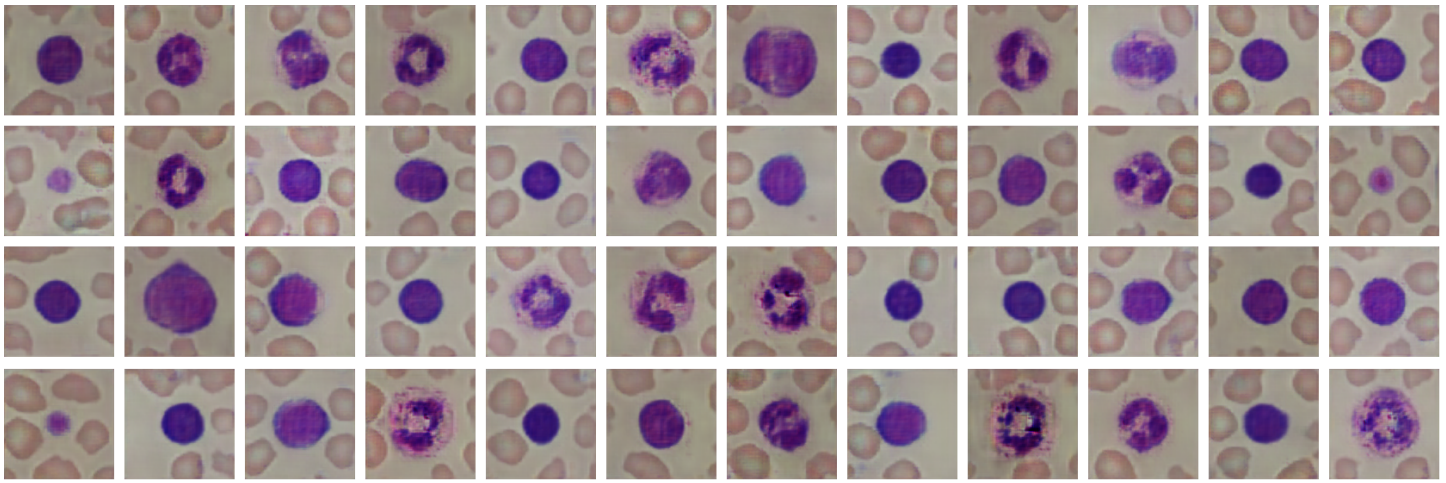
7 Contributions

All four members of the team contributed to each aspect of the project, but each focused more on different areas. Aditi focused on implementing and performing experiments on the DC-GAN model as well as generating synthetic images of cells using our trained generator. Peter focused on implementing the WGAN-GP and infoGAN model as well as reducing total run time by optimizing code to run on Tensor Processing units(TPUs). Victor focused on hyper-parameter tuning on the WGAN-GP and infoGAN model as well as generating synthetic images of cells using our trained generator. Louise focused on building out our binary classifiers and evaluating their performance on the real and augmented datasets. All team members contributed to the literature review, proposal, milestone, and final reports.

8 Appendix

8.1





8.2

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FP} \tag{3}$$

$$Recall = \frac{TP}{TP + FN} \tag{4}$$

$$F1\,Score = 2 \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{5}$$

$$Specificity = \frac{TN}{TN + FP} \tag{6}$$

References

[1] Ahmed, N., Yigit, A., Isik, Z., & Alpkocak, A. (2019). Identification of Leukemia Subtypes from Microscopic Images Using Convolutional Neural Network. Diagnostics (Basel, Switzerland), 9(3), 104. https://doi.org/10.3390/diagnostics9030104

[2] Sharma, N. (2020, September 28). Leukemia dataset. Retrieved April 17, 2021, from https://www.kaggle.com/nikhilsharma00/leukemia-dataset

[3] Marzahl, C., Aubreville, M., Voigt, J., & Maier, A. (2019). Classification of Leukemic B-Lymphoblast Cells from Blood Smear Microscopic Images with an Attention-Based Deep Learning Method and Advanced Augmentation Techniques, Singapore.

[4] Cloud tensor processing units (TPUs). (n.d.). Google Cloud; Google. Retrieved June 2, 2021, from https://cloud.google.com/tpu/docs/tpus

[5] Cloud TPU. (n.d.). Google Cloud; Google. Retrieved June 2, 2021, from https://cloud.google.com/tpu

[6] Deep Convolutional Generative Adversarial Network. (2019). Google Colab. https://colab.research.google.com/github/tensorflow/docs/blob/master/site/en/tutorials/generative/dcgan.ipynb

[7] Nain, A. K. (2020, May 9). WGAN-GP overriding 'Model.train_step'. Keras Documentation; Keras. https://keras.io/examples/generative/wgan_gp/

[8] Lucic, M., Kurach, K., Michalski, M., Bousquet, O., & Gelly, S. (2018). Are gans created equal? A large-scale study. 21. https://arxiv.org/pdf/1711.10337.pdf

[9] Kingma, D. P., Ba, J. (2017). Adam: A method for stochastic optimization. ArXiv:1412.6980 [Cs]. http://arxiv.org/abs/1412.6980

[10] Hinton, G., Srivastava, N., Swersky, K. (n.d.). Neural Networks for Machine Learning: Lecture 6a Overview of mini-batch gradient descent . Lec6; University of Toronto. https://www.cs.toronto.edu/ hinton/coursera/lecture6/lec6.pdf

[11] Arjovsky, M., Chintala, S., & Bottou, L. (2017). Wasserstein GAN. https://arxiv.org/abs/1701.07875

[12] M. Arjovsky, S. Chintala, & L. Bottou. Wasserstein gan. arXiv preprint arXiv:1701.07875, 2017.

[13] Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V. & Courville, A. Improved training of wasserstein GANs. Adv. Neural Inf. Process. Syst. 2017-December, 5768–5778 (2017).

[14] Yi, X., Walia, E. & Babyn, P. Generative adversarial network in medical imaging: A review. Med. Image Anal. 58, (2019).

[15] Hu, B. et al. Unsupervised learning for cell-level visual representation in histopathology images with generative adversarial networks. IEEE J. Biomed. Heal. Informatics 23, 1316–1328 (2019).