A Survey on Data Augmentation for Multi-Class Image Classification

Aastha Agrawal
Georgia Institute of Technology
aagrawal319@gatech.edu

Karthik Nama Anil
Georgia Institute of Technology
karthikna@gatech.edu

KEYWORDS

Data Augmentation, Multi-class Image Classification, Accuracy, Generative Adversarial Networks, Neural Networks

ACM Reference Format:
Aastha Agrawal and Karthik Nama Anil. 2019. A Survey on Data Augmentation for Multi-Class Image Classification. In *Proceedings of Project Proposal (CS 8803 DMM: Data Management and Machine Learning).* ACM, New York, NY, USA, 7 pages.

1 INTRODUCTION

Over past few years, the field of Computer Vision has heavily utilized Deep Learning for obtaining state-of-the-art results for problems like image classification, object detection and image reconstruction. One of the major challenges when applying neural networks to images is the scarcity of samples or put in other words, the small size of data set. With state-of-art neural networks for images, number of features are in millions. For an optimal performance of these models, we need more number of examples than the number of features. This is a challenging problem. Data augmentation is becoming the go-to-technique for increasing the size of data.

1.1 What is Data Augmentation?

Data augmentation [11] is a strategy that enables practitioners to significantly increase both size and diversity of available data set for training models, without actually collecting new data. Data augmentation techniques may be based on extrapolation heuristics, aggregation, tagging and probability. Specifically, for images, it would involve techniques like cropping, rotation, scaling and many more.

1.2 Why Data Augmentation?

With growing popularity and significance of deep learning in computer vision domain and limited data to train on, data augmentation is a crucial tool.

As highlighted earlier, one of the major challenge when running deep neural networks is the issue of scarcity. When utilizing images for training models, number of features scale up to millions. This implies that we need to feed at least few million examples to the model to get optimum results. Big data is required for achieving high accuracy of these models. This aspect of size of data set is thoroughly studied in [4] and [8]. In applications like medical image analysis, achieving high accuracy becomes very significant given the nature of domain. Esteva et al. [5] demonstrate that if data set is large, deep CNNs are very powerful for medical image analysis tasks like skin cancer classification. Medical image data set can be especially small where rare diseases, patient privacy or medical experts for labelling are involved. Even when using academic data sets like MNIST [28] and ImageNet [26], researchers generally take small subset of these big data sets as they want to simulate problems related to specific classes. Data Augmentation is highly useful in overcoming smaller data sets in such scenarios.

In real world, it is common to have image data set taken under limited conditions. However, our target application may exist in different conditions like different angle, orientation and scale. It is important to feed the model a more robust dataset covering all conditions. For example, in face recognition application, it might be possible that given images are collected from a single angle. However, in target application, we would most likely have to identify faces from different angles. This challenge could be overcome by synthetically creating new images from given images. Due to similar reasons, it is common to see class imbalance and over fitting problem with image data sets. Mikolajczyk et al. [32] highlights that data augmentation is able to address the lack of labeled data and class imbalance to a certain extent in image classification.

1.3 Multi-class Image Classification Problem

Image classification is an important problem in Computer Vision. The objective of this problem is to label each image of the test dataset. Image classification, can be binary or multi-class. For example, labelling an X-ray image as cancer or not would be an example of binary classification. Classifying a picture of a handwritten digit to its digit value is a multi-class image classification problem. For the project, we have limited the scope of our study of data augmentation techniques and their evaluation to this specific problem. We evaluated selected techniques with respect to multi-class image classification task.

1.4 Motivation

In this project, we performed comprehensive study of various data augmentation techniques and evaluated performance of selected techniques with respect to multi-class image classification task. The results from the study would aid researchers to select the correct image augmentation technique for a given image dataset.

2 RELATED WORK

In this study, we have broadly classified the data augmentation techniques as - Simple and Complex. In next two subsections, we discuss in detail the various simple and complex data augmentation tasks.

2.1 Simple Data Augmentation Techniques

In this section we cover the most popular simple data augmentation techniques for images. These techniques work on different attributes of a single image for generating a new image, uses less computation power, are easy to apply and have low execution time.

Color Inversion: Exchanges colour values to for certain colours to generate new images. For example, black drawing on a white screen becomes white drawing on a black screen[40].

Color Jittering: Change the color saturation, brightness and contrast in the image color space by a small random value[22].

Cropping: Crop a random or selected section of the original image and resize the cropped image to the original size or specific resolution. [14]. Random crop will crop the image at random location [1] while center crop will crop the image at the center[1]. Five crop is another variation where the four corners are cropped along with center crop[1].

Flipping: Flip the image in the horizontal or vertical direction to obtain a new image[14].

Gaussian Noise: Add noise to RGB channels of each pixel in the image based on the 2D colour distribution in the image to generate a new image[14].

Grayscale: Convert image to grayscale in one to three channels. Random grayscale will randomly convert image to grayscale with a probability[1].

Kernel Filters: Kernel filters are very popular in Computer Vision. These work by sliding a kernel which is a NxN matrix across an image to blur or sharpen it. Kang et al. [18] puts forward a kernel-filter based data augmentation technique called PatchShuffle Regularization.

Pad: Pad will add padding to the image with the specified pad value[1].

PCA Jittering: Perform PCA on the image to get the principal component, which is then added to the original image with a Gaussian disturbance of (0, 0.1) to generate the new image[22].

Random Affine: In geometry, an affine transformation, affine map[1] or an affinity (from the Latin, affinis, "connected with") is a function between affine spaces which preserves points, straight lines and planes. Also, sets of parallel lines remain parallel after an affine transformation. An affine transformation does not necessarily preserve angles between lines or distances between points, though it does preserve ratios of distances between points lying on a straight line. Random affine transformation of the image keeping center invariant[1].

Random Erase: The pixels within a random rectangular area are set to random RGB values. Less than 50% of all pixels must be changed within the selected box[42].

Random Perspective: Performs perspective transformation of the given image randomly with a given probability[1].

Rotation: A random rotation of maximum ±25°ensures severe cropping does not remove defining class features and rotation beyond this range may generate highly distorted unrealistic images[14].

Scale: The image can be scaled outward or inward. While scaling outward, the final image size will be larger than the original image size[14]. Sometimes, scaling is also referred to as resizing[1].

Shear: The image is stretched and sheared along the axial planes with a maximum shear of ±20°to ensure class preservation[40].

Skew Tilt: The image is tilted forwards, backwards, left or right to a maximum of 22.5°and thus creating an illusion of image being viewed from different perspectives[40].

Translation: Translation involves shifting the image along the either one or both the axes[14].

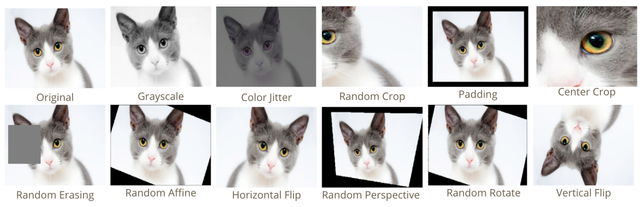


Figure 1: Simple Image Augmentation Techniques

2.2 Complex Data Augmentation Techniques

In this section we cover the complex data augmentation techniques for images. These techniques usually work on more than one image to augment the data set, generally require high computational power and have longer execution time.

AutoAugment provides a simple procedure to automatically search for improved data augmentation policies[13]. Each policy is split into multiple sub-policies and is applied to each image in a mini-batch. The sub-policies consists of two operations - an image processing function (like translation, rotation or shearing) and the probabilities and magnitudes with which the functions are applied. The search algorithm is used to find the best policy such that the neural network yields the highest validation accuracy on a target dataset.

Feature space Augmentation This technique uses neural networks to map high-dimensional inputs into lower-dimensional representations, then performs manipulation at these lower level representation and then finally, reconstructs these vectors to high-dimensional images[12].

Population Based Augmentation (PBA) is a novel formulation of data augmentation search which quickly and efficiently learns the best augmentation policy schedules. PBA[10] generates non-stationary augmentation policy schedules instead of a fixed augmentation policy. PBA leverages the Population Based Training algorithm to generate an augmentation schedule that defines the best augmentation policy for each epoch of training. The augmentation policy changes with the epoch.

Neural Style Transfer manipulates the representations of images created in CNNs with the goal of transferring the style of one image to another while preserving content its original content. Although this technique is useful for artistic creation but does not have many sceintific applications of data augmentation[29].

GAN-based Data Augmentation Generative Adversarial Networks (GANs) are a powerful class of neural networks that are used for unsupervised learning[39]. GANs are basically made up of a system of two competing neural network models which compete with each other and are able to analyze, capture and copy the variations within a dataset. These techniques use generative modelling to augment data sets.

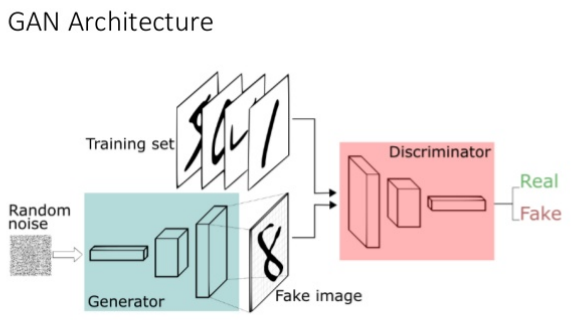


Figure 2: GAN Architecture

AAE (Adversarial Autoencoder) is a probabilistic autoencoder[3] that uses GANs to perform variational inference by matching the aggregated posterior of the hidden code vector of the autoencoder with an arbitrary prior distribution. This GAN is very quick in execution compared to others.

AugGAN is a structure-aware image-to-image translation network[20] that is composed of encoders, discriminators, generators, and parsing nets for the two domains, respectively, in a unified framework to generate hyper-realistic synthetic images.

ACGAN (Auxiliary Classifier GAN) employs label[37] conditioning to synthesise high-resolution images exhibiting global coherence.

BAGAN (Balancing Generative Adversarial Networks) is an augmentation tool to restore balance in imbalanced image datasets[15]. BAGAN uses class conditioning in the latent space to drive the generation process towards a target class. The generator in the GAN autoencoder is that enables the algorithm to learn an accurate class-conditioning in the latent space.

BGAN (Boundary-Seeking GAN) is method for training GANs with discrete data[38] that uses the estimated difference measure from the discriminator to compute importance weights for generated samples, thus providing a policy gradient for training the generator. The importance weights have a strong connection to the decision boundary of the discriminator, and hence the method is referred to as boundary-seeking GANs.

Conditional GAN is one of the earliest GANs and can be constructed by simply feeding data[34] by conditioning both the generator and discriminator. Conditional GANs could be used to learn a multi-modal model, and provide preliminary examples of an application to image tagging.

CoGAN (Coupled GAN) was developed for learning a joint distribution[33] of multi-domain images without any tuple of corresponding images. It can learn a joint distribution with just samples drawn from the marginal distributions by enforcing a weight-sharing constraint that limits the network capacity and favors a joint distribution solution over a product of marginal distributions one.

CycleGAN performs automatic training[23] of image-to-image translation models without paired examples in an unsupervised setting. CycleGANs embed the details of the source image in the augmented image in such a way that the source image can be recovered and thus satisfy the cyclic consistency requirement. The algorithm generates realistic images.

DAGAN (Data Augmentation GAN) is a variation of GANs that takes data[6] from a source domain and generalise it to generate images of the other classes with a small sample of new class of images.

DCGAN (Deep Convolutional GAN) is commonly used for image augmentation[2] tasks and is a topologically constrained variant of conditional GAN. DCGAN tries to address the scalability issues found in GANs by replacing pooling layers with strided convolutions and completely eliminating the hidden layers. DCGAN utilises an unique approach to spread the error from one convolution layer to the next to reduce the execution time of algorithm.

DRAGAN uses regret minimization and proposed the existence of undesirable local equilibria in this non-convex game to be responsible for mode collapse. DRAGAN[35] enables faster training, achieves improved stability with fewer mode collapses, and leads to generator networks with better modeling performance across a variety of architectures and objective functions.

EBGAN (Energy-based GAN) is similar to probablistic GANs - a generator is seen as being trained to[24] produce contrastive samples with minimal energies, while the discriminator is trained to assign high energies to these generated samples. The single scale architecture of the GAN can be trained to generate high-resolution images.

GAN (Generative Adversarial Network): This was the first GAN ever to be developed. GANs use unsupervised learning and are made up of a system of two competing neural network models which compete with each other. These networks can analyze, capture and copy the variations within a dataset. Most of the neural nets can be easily fooled into misclassifying things by adding only a small amount of noise into the original data.

InfoGAN (Interpretable Representation Learning by Information Maximizing GAN) is able to learn disentangled representations in a completely unsupervised manner and also maximizes the mutual information between a small subset of the latent variables and the observation. InfoGAN implements a variation of the Wake-Sleep algorithm. InfoGAN[41] successfully disentangles writing styles from digit shapes on the MNIST dataset and background digits from the central digit on the SVHN dataset.

LSGAN (Least Squares GAN) adopt the least squares loss function for the discriminator[31] to handle the problem of vanishing gradients problem during the learning process. LSGANs are able to generate high quality images and perform more stable during the learning process.

SGAN (Semi-Supervised Learning GAN) extend GANs to the semi-supervised context[36] by forcing the discriminator network to output class labels. SGAN trains a generative model G and a discriminator D on a dataset with inputs belonging to one of N classes. At training time, D is made to predict which of N+1 classes the input belongs to, where an extra class is added to correspond to the outputs of G. We show that this method can be used to create a more data-efficient classifier and that it allows for generating higher quality samples than a regular GAN.

Softmax GAN is an varient of the original GAN and it replaces[30] the classification loss in the original GAN with a softmax cross-entropy loss in the sample space of one single batch. Softmax GAN is the Importance Sampling version of original GAN and can be used to realistic images.

SRGAN (Super Resolution GAN) is used to generate high resolution images and is particularly useful in optimally up-scaling native low-resolution images to enhance its details minimizing errors while doing so [9].

Vanilla GAN is the simplest type of GAN where the generator and discriminator are simple multi-layer perceptrons and aims to optimize a mathematical equation using stochastic gradient descent[16].

VAE-GAN (Variational Autoencoder- Generative Adversarial Network) combines best of both VAEs and GANs. It replaces GAN discriminator in place of a VAE's decoder to learn the loss function to tackle *blurriness* of VAEs [27].

Wasserstein GAN is an extension to the GAN[7] that both improves the stability when training the model and provides a loss function that correlates with the quality of generated images.

Wasserstein GAN GP may generate low-quality samples or fail to converge due to the use of weight clipping in WGAN to enforce a Lipschitz constraint on the critic which is undesired. Wasserstein GAN GP[17] penalize the norm of gradient of the critic with respect to its input and generate superior quality images.

3 PERFORMANCE EVALUATION

In this section, we will talk about the evaluation set up and the results.

3.1 Simple Techniques

We evaluated performance of following simple data augmentation methods: Color Jitter, Gray Scale, Padding, Random Affine, Random Horizontal Flip, Random Perspective, Random Rotation, Random Vertical, Random Erasing, Center Crop, Random Crop and Resize. Our experiments used PyTorch implementation of these techniques. Experiment details and results are discussed in detail below:

Dataset Used: CIFAR-10 [25]

It consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images.

Metric Used: Accuracy and Execution time

Accuracy represents the fraction of total predictions the classifier correctly predicted. It can be defined as follows:

$$\text{Accuracy} = \frac{\text{Total number of correct predictions}}{\text{Total predictions}}$$

Classifier: (Pretrained) Resnet with 18 layers

ResNet[19] is the first neural network to train extremely deep neural networks with 150+layers successfully. ResNet explicitly reformulates the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions to increase the accuracy of image classification. We ran RESNET with 50 epoch and for data loader used 64 batches and 8 workers.

GPU: NVIDIA Tesla T4

Baseline: We set our baseline to accuracy results of pretrained Resnet classifier on original CIFAR-10 data set (without augmentation).

Results: From 3, we can see that Resize improved the accuracy significantly. On the other hand, techniques like Color Jitter, Gray Scale, Random Perspective, Random Vertical and Random erasing degraded the classifier performance. Decline in accuracy could be explained by loss of important information or attributes, for example, loss of colour attribute in case of gray scale data augmentation. Another example would be random erasing could potentially erase parts of picture which make it difficult to differentiate between car and truck.

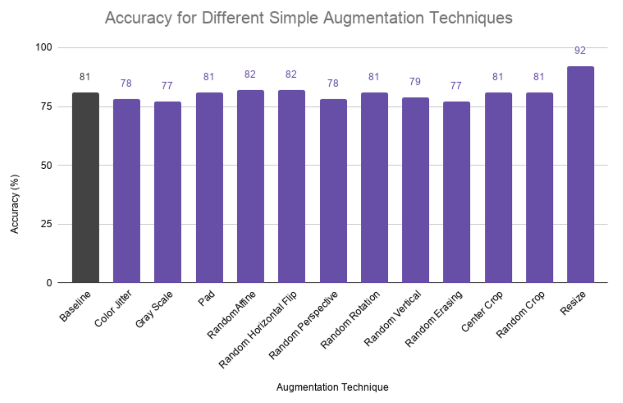


Figure 3: Accuracy for Different Simple Augmentation Techniques

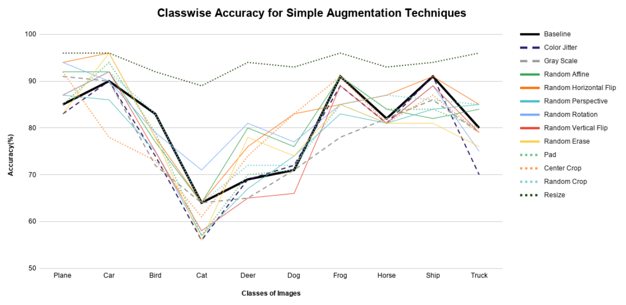


Figure 4: Classwise Accuracy for Simple Augmentation techniques

We also studied individual class accuracy for all these techniques. From 4, we can draw following inferences:

- Resize improves accuracy consistently for all categories.
- *Cat* class has lowest accuracy among all classes for all techniques.
- Random Horizontal Flip improves accuracy for all classes except *frog* and *bird* categories.
- Majority of the experimented augmentation methods improve accuracy for *Plane*, *Car*, *Deer*, *Dog* and *Truck* but degrades performance for classes *Bird*, *Frog* and *Ship*.

We also captured overall execution time for our experiments which includes time taken for data augmentation, classifier training and classifier predictions for test set. Figure 5 compares the execution time for all the techniques. It shows that majority of the techniques have execution time between 1200-1550 seconds (20-26 minutes). Resize took maximum time (approximately 40 minutes) to execute.

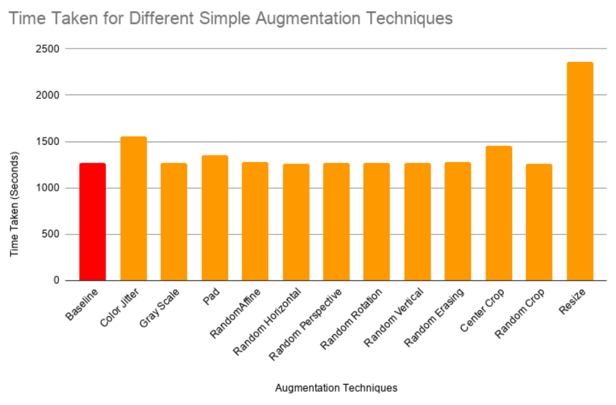


Figure 5: Time Taken for Different Simple Augmentation Techniques

3.2 Complex Techniques

This subsection will provide details of experiments on complex augmentation techniques and highlight the results obtained. We experimented with different GAN models for capturing their D loss, G loss and execution time only. All the tests have been performed on the open source implementation of the selected GANs.

Dataset Used: MNIST [28] This database of handwritten digits(0-9) comprises of 28x28 pixel grayscale images with a training set of 60,000 examples and a test set of 10,000 examples. The database is suitable for pattern recognition tasks on real-world data.

Metrics Used: D loss, G loss and Execution time

We train D to maximize the probability of assigning the correct label to both training examples and samples from G. Described mathematically, the discriminator seeks to maximize the average of the log probability for real images and the log of the inverted probabilities of fake images.

maximize $\log D(x) + \log(1 - D(G(z)))$

Execution time is the time taken to complete the augmentation task.

CPU: 2.3 GHz Intel Core i5

RAM: 16 GB 2133 MHz LPDDR3

Epochs: 50

Number of batches: 64

Number of workers: 8

Baseline: D loss and G loss of the original GAN[21] was taken as the baseline for evaluation. We also compared the execution time of different GANs with the original GAN.

Fig 6 shows the execution times of different GANs in comparison with original GAN. AAE, BGAN, CGAN and WGAN have execution time lesser or comparable with original GAN. All other implementations take at least twice or thrice the time taken by original GAN. Other GANs take longer due to the internal complexity of the neural nets the discriminator and generator are using. The way the

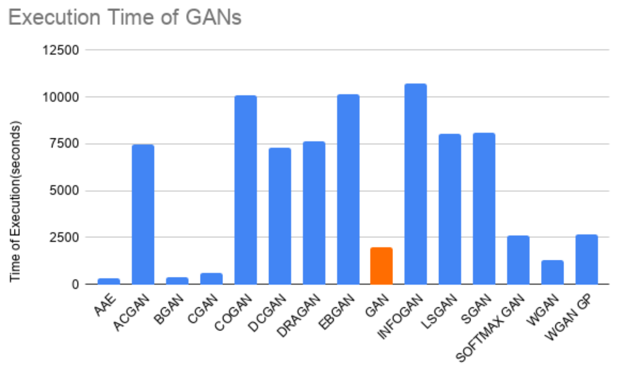


Figure 6: Time Taken for Different GANs

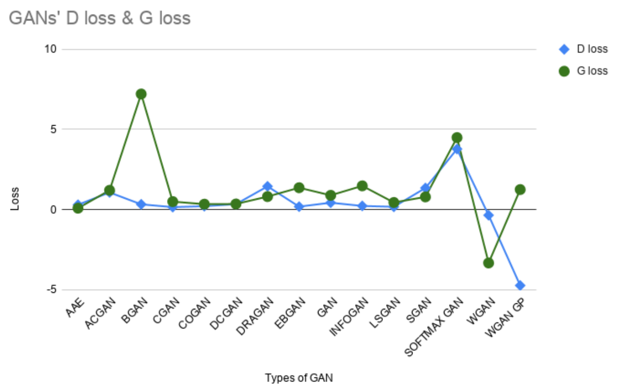


Figure 7: GANs D loss and G loss

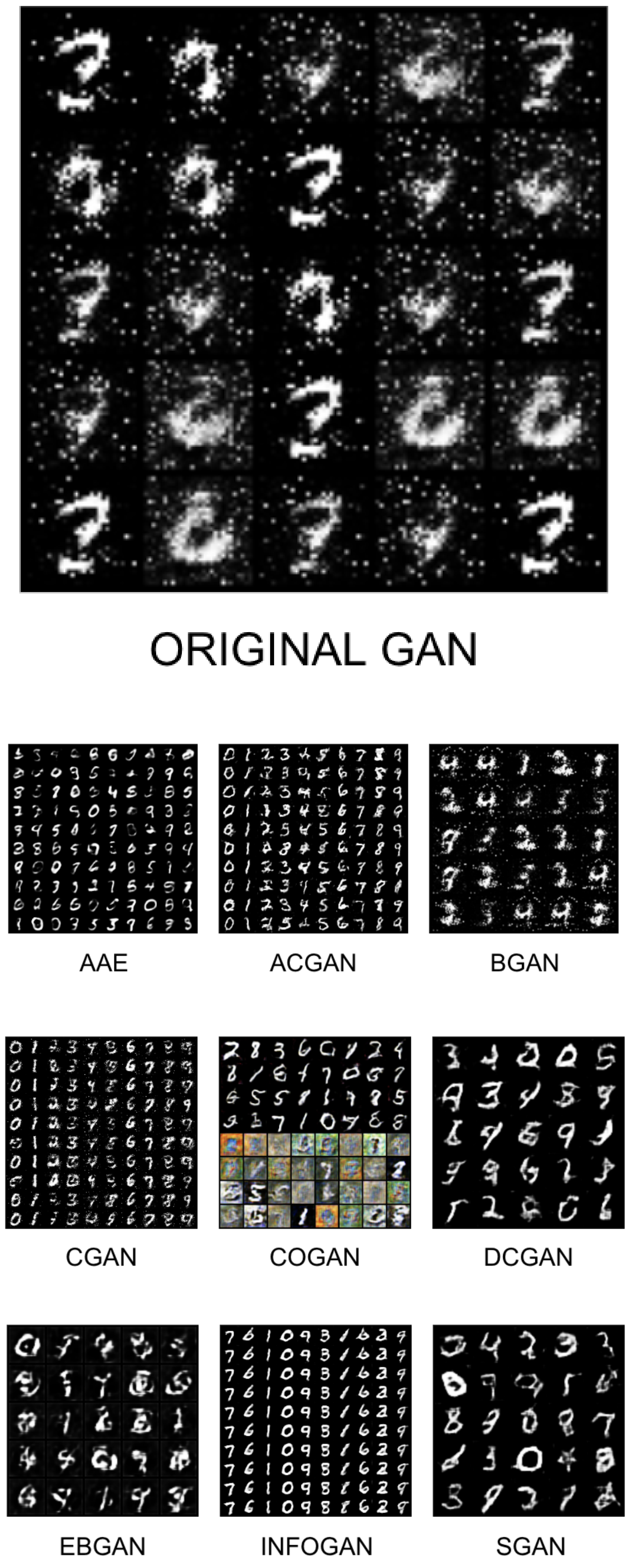
loss is propagated through the network also determines the time taken for execution.

The D loss and G loss must be as small as possible. In Fig 7, we observe BGAN, Softmax GAN, WGAN and WGAN GP have very high losses and thus the generated images may not be of high quality. Fig 8. shows the output of the various GANs and it is evident that digits "0" and "1" are more accurately generated compared to other digits. The output of InfoGAN is easily interpretable and the output is of higher quilty than other GANs. All GANs performed better than the baseline original GAN interms of the quality of the output generated.

4 CONCLUSION

Simple image augmentation techniques had short execution time and are easy to apply, visualize and map generated images to the original image. We noticed that not all augmentation techniques would improve classification results. Selecting augmentation method depends both on the domain and classification task.

Complex image augmentation techniques had longer execution time, required high computation power and it is hard to map output image to the original set of images. Quite a few techniques resulted in high quality augmented images. GANs require large number of



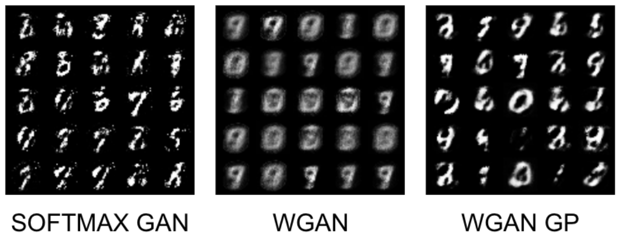


Figure 8: Output of GANs after 50 epochs of execution

training images to generate high resolution images. This is counter intuitive since we are using GANs to handle the problem of scarcity of data. Techniques like Neural style transfer are creative but has few scientific applications.

In general, image augmentation requires domain specific knowledge to generate high quality images. The proposed models are hard to generalise and a model developed for one task may generate invalid augmented images for another task or in a different context.

Our evaluation methodology covered only a subset of all available image augmentation techniques. In future, one can evaluate more techniques and come up with another approach to classify different augmentation methods based on their approach, algorithm used, performance and other metrics.

ACKNOWLEDGEMENTS

We are grateful for the continuous support provided by Professor Xu Chu during the project. We would like to thank Google for providing access to the GPUs NVIDIA Tesla T4 and NVIDIA Tesla K80 for computation through their Google Colab platform.

REFERENCES

[1] 2019. Torchvision Transforms Documentation. (2019). https://pytorch.org/docs/stable/torchvision/transforms.html
[2] Soumith Chintala Alec Radford, Luke Metz. 2015. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. arXiv:cs.LG/1511.06434
[3] Navdeep Jaitly Ian Goodfellow Brendan Frey Alireza Makhzani, Jonathon Shlens. 2015. Adversarial Autoencoders. arXiv:cs.LG/1511.05644
[4] Peter Norvig Alon Halevy and Fernando Pereira. 2009. The Unreasonable Effectiveness of Data. *IEEE Intelligent Systems* 24, 2 (March 2009), 8–12. https://doi.org/10.1109/MIS.2009.36
[5] Roberto A. Novoa Justin Ko Susan M. Swetter-Helen M. Blau Sebastian Thrun Andre Esteva, Brett Kuprel. 2017. Dermatologist-level classification of skin cancer with deep neural networks. *Nature* (February 2017), 115–118. https://doi.org/10.1038/nature21056
[6] Antreas Antoniou, Amos Storkey, and Harrison Edwards. 2017. Data Augmentation Generative Adversarial Networks. (2017). arXiv:stat.ML/1711.04340 https://arxiv.org/abs/1711.04340
[7] Martin Arjovsky, Soumith Chintala, and Léon Bottou. 2017. Wasserstein GAN. arXiv:stat.ML/1701.07875
[8] Saurabh S Chen S, Abhinav S and Abhinav G. 2017. Revisting unreasonable effectivness of data in deep learning era. In *Proceedings of the 2017 International Conference on Computer Vision.*
[9] Ferenc Huszar Jose Caballero et al Christian Ledig, Lucas Theis. 2016. Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. arXiv:cs.CV/1609.04802
[10] Ion Stoica Pieter Abbeel Xi Chen Daniel Ho, Eric Liang. 2019. Population Based Augmentation: Efficient Learning of Augmentation Policy Schedules. (2019). arXiv:cs.CV/1905.05393 https://arxiv.org/pdf/1905.05393
[11] Richard Liaw Daniel Ho, Eric Liang. 2019. *1000x Faster Data Augmentation.* Retrieved Oct 15, 2019 from https://bair.berkeley.edu/blog/2019/06/07/data_aug/
[12] Terrance DeVries and Graham W. Taylor. 2017. Dataset Augmentation in Feature Space. arXiv:stat.ML/1702.05538

[13] Dandelion Mane Vijay Vasudevan Quoc V. Le Ekin D. Cubuk, Barret Zoph. 2018. AutoAugment: Learning Augmentation Policies from Data. (2018). arXiv:cs.CV/1805.09501 https://arxiv.org/abs/1805.09501

[14] Arun Gandhi. 2018. *Augmentation | How to use Deep Learning when you have Limited Data — Part 2.* https://nanonets.com/blog/data-augmentation-how-to-use-deep-learning-when-you-have-limited-data-part-2/

[15] Roxana Istrate Costas Bekas Cristiano Malossi Giovanni Mariani, Florian Scheidegger. 2018. BAGAN: Data Augmentation with Balancing GAN. arXiv:cs.CV/1803.09655 https://arxiv.org/abs/1803.09655

[16] Ian J. Goodfellow and Yoshua Bengio. 2014. Generative Adversarial Networks. arXiv:stat.ML/1406.2661

[17] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, and Aaron Courville. 2017. Improved Training of Wasserstein GANs. arXiv:cs.LG/1704.00028

[18] Liang Zheng Guoliang Kang, Xuanyi Dong and Yi Yang. 2017. PatchShuffle Regularization. *CoRR* abs/1707.07103 (2017). arXiv:1707.07103 http://arxiv.org/abs/1707.07103

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. (2015). arXiv:cs.CV/1512.03385 https://arxiv.org/abs/1512.03385

[20] Sheng-Wei Huang, Alex Lin, Shu-Ping Chen, Yen-Yi Wu, Po-Hao Hsu, and Shang-Hong Lai. 2018. AugGAN: Cross Domain Adaptation with GAN-based Data Augmentation.

[21] Mehdi Mirza Bing Xu David Warde-Farley Sherjil Ozair Aaron Courville Yoshua Bengio Ian J. Goodfellow, Jean Pouget-Abadie. 2014. Generative Adversarial Networks. arXiv:stat.ML/1406.2661

[22] Jia Peiyi Hu Siping Jia Shijie, Wang Ping. 2017. Research on data augmentation for image classification based on convolution neural networks. (2017). https://doi.org/10.1109/CAC.2017.8243510

[23] Phillip Isola Alexei Efros Jun-Yan Zhu, Taesung Park. 2017. Unpaired Image-To-Image Translation Using Cycle-Consistent Adversarial Networks. In *The IEEE International Conference on Computer Vision (ICCV)*.

[24] Yann LeCun Junbo Zhao, Michael Mathieu. 2016. Energy-based Generative Adversarial Network. arXiv:cs.LG/1609.03126

[25] Alex Krizhevsky. 2009. CIFAR-10. (2009). https://www.cs.toronto.edu/~kriz/cifar.html

[26] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1 (NIPS'12)*. Curran Associates Inc., USA, 1097–1105. http://dl.acm.org/citation.cfm?id=2999134.2999257

[27] Anders Boesen Lindbo Larsen, Søren Kaae Sønderby, Hugo Larochelle, and Ole Winther. 2015. Autoencoding beyond pixels using a learned similarity metric. arXiv:cs.LG/1512.09300

[28] Yann LeCun and Corinna Cortes. 1999. The MNIST Dataset Of Handwritten Digits (Images). (1999). http://www.pymvpa.org/datadb/mnist.html

[29] Alexander S. Ecker Leon A. Gatys and Matthias Bethge. 2015. A Neural Algorithm of Artistic Style. *CoRR* abs/1508.06576 (2015). arXiv:1508.06576 http://arxiv.org/abs/1508.06576

[30] Min Lin. 2017. Softmax GAN. arXiv:cs.LG/1704.06191

[31] Xudong Mao, Qing Li, Haoran Xie, Raymond Y. K. Lau, Zhen Wang, and Stephen Paul Smolley. 2016. Least Squares Generative Adversarial Networks. arXiv:cs.CV/1611.04076

[32] Agnieszka Mikołajczyk Michal Grochowski. 2018. Data augmentation for improving deep learning in image classification problem. *IEEE Xplore* (October 2018). https://ieeexplore.ieee.org/document/8388338

[33] Oncel Tuzel Ming-Yu Liu. 2016. Coupled Generative Adversarial Networks. arXiv:cs.CV/1606.07536

[34] Mehdi Mirza and Simon Osindero. 2014. Conditional Generative Adversarial Nets. arXiv:cs.LG/1411.1784

[35] James Hays Zsolt Kira Naveen Kodali, Jacob Abernethy. 2017. On Convergence and Stability of GANs. arXiv:cs.AI/1705.07215

[36] Augustus Odena. 2016. Semi-Supervised Learning with Generative Adversarial Networks. arXiv:stat.ML/1606.01583

[37] Augustus Odena, Christopher Olah, and Jonathon Shlens. 2016. Conditional Image Synthesis With Auxiliary Classifier GANs. (2016). arXiv:stat.ML/1610.09585 https://arxiv.org/abs/1610.09585

[38] Tong Che Yoshua Bengio R Devon Hjelm, Athul Paul Jacob. 2017. Boundary-Seeking Generative Adversarial Networks. arXiv:stat.ML/1702.08431

[39] Rahul Roy. [n.d.]. Generative Adversarial Network (GAN). ([n. d.]). https://www.geeksforgeeks.org/generative-adversarial-network-gan/

[40] Kevin McGuinness Sarah O'Gara. 2019. Comparing Data Augmentation Strategies for Deep Image Classification. (2019). https://doi.org/148b-ar75

[41] Rein Houthooft John Schulman Ilya Sutskever Pieter Abbeel Xi Chen, Yan Duan. 2016. InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets. arXiv:cs.LG/1606.03657

[42] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. 2017. Random Erasing Data Augmentation. *ArXiv* abs/1708.04896 (2017).