Augmentation Techniques Analysis with Removal of Class Imbalance Using PyTorch for Intel Scene Dataset

ANUSHA KANUKOLANU[1], DR. S PHANI KUMAR[2], ALLENA VENKATA SAI ABHISHEK[3]
[1]*Research Scholar, GITAM University*
[2]*Professor and HOD, GITAM University*
[3]*Research Scholar, IIIT Bangalore*

Abstract*— although best-in-class AI can deliver extraordinary outcomes in experimentation, data scientists struggle to duplicate these outcomes on actual-world data. It's nothing unexpected - actual data mirrors the messy world that made it, containing many biases and gaps. A painful element of real data is that it tends to be imbalanced. An imbalanced dataset is a dataset with a lot more examples in one class than others. This exploration features the broad study about taking care of class Imbalance issues utilizing Random Sampling and Data Augmentation Techniques. The critical angle featured is to grasp how Under-Sampling, Over-Sampling, and Data Augmentation use images and custom datasets. The model performance is improved with the expulsion of class imbalance issue utilizing different Augmentation approaches utilizing an augmentation library. The accuracy contrasts with taking care of the Class imbalance issue to boost accuracy, lessen error, and track down an ideal technique to tackle it.

Indexed Terms-- Class Imbalance, Intel Scene Dataset, Augmentation, RESNET, Classification.

I. INTRODUCTION

Image classification [16] is doling out the images to respective classes. It has a massive repercussion in different areas of exploration and research. Deep learning [3] [6] has made the pipeline much more efficient and faster. With the widespread use of frameworks like PyTorch [20] and TensorFlow [21], neural networks are widely used in research work & companies like Google. There are numerous augmentations [1] [5] [7] [11] [12] [13] upheld by different augmentation libraries that assist in the progress of the accuracies of the various models. To find the best increase strategy because of accuracy utilizing a model using AugStatic [7] [9] library/package/module, which will produce an expanded adjusted dataset, like this, tackling the Class imbalance [1] [10] [15] issue. Each neural network model has an alternate capacity and gives shifting viewpoints on this steadily developing, quick evolving space. The review connects with investigating the augmentation techniques procedures that are compelling and incapable for different datasets while at the same time working on the model execution with fixed test information and model.

II. RELATED WORK

There are numerous augmentations upheld by different augmentation libraries that assist in creating new examples. There are many image order datasets. The improvement of the model exhibitions differs for each dataset, augmentation, and model utilized. To find the best augmentation strategy in light of accuracy using a RESNET18 [8] [17] model utilizing an exclusively constructed augmentation library that will produce an augmented balanced dataset, tackling the class imbalance issue. The test is to assist the model with gaining from fewer data. For which the data augmentation strategies are considered for oversampling. Numerous augmentation libraries and ideas like mixup augmentation [12], matting [14], torchvision changes, augmentor [5], and albumentations [11] were contrasted with picking the productive bundle to involve the augmentation methods.

Over-sampling eliminates the imbalance by producing new images with existing pictures in the minority class. Different data augmentation is a strategy utilized for oversampling. Changes are made after existing images like rotation, scale, and so on. For instance,

assume there is a cat image. We make another augmented image in which it has gone through certain transformations like we can rotate or flip or increment the brilliance or increment the differentiation, and so on. In this way, various augmentation techniques are applied to make new pictures that assist the model with learning even with fewer data.

III. RESEARCH METHODOLOGY

The neural network model utilized is RESNET18. It has 18 layers and comprises skip connections used to assist the model with finding out more. The one residual block [2] contains two weight layers and has a skip connection for each remaining square, interfacing the result of the subsequent weight layer with a ReLU actuation function [18]. An additional sequential layer [19] has been added to get the accuracy of 0 to 1. We utilize the PyTorch structure to construct models. The RESNET 18 isn't a pre-trained Adam optimizer utilized, and a negative log-probability likelihood loss function is used. The RESNET18 engineering is referenced in fig 5 and 6.

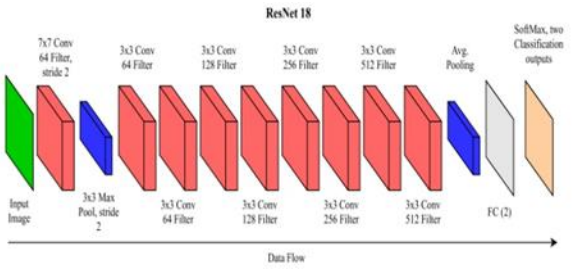

Figure 1. ResNet-18 Architecture

```
(fc): Sequential(
  (0): Linear(in_features=512, out_features=512, bias=True)
  (1): ReLU()
  (2): Dropout(p=0.2, inplace=False)
  (3): Linear(in_features=512, out_features=2, bias=True)
  (4): LogSoftmax(dim=1)
```

Figure 2. Additional sequential layer

An extra successive layer is added toward the end, as referenced in fig 6. The info will be passed to linear (512,512), whose result is taken care of by the first ReLU activation function. Then a dropout (0.2) layer is being utilized, trailed by linear (512, 2). At last, it is gone through a LogSoftmax to get the logarithm of the probabilities.

A classification image data set is utilized with a train and test set. For the sole motivation behind preparing a model, it is finished using a train set. The ready model is tried upon a fixed test set, and they are put away in a log document for our records. The datasets are used to train the model, and the model's accuracy is acquired by testing upon the test set.

Two classes are picked out of the dataset and are adjusted physically with the end goal that both the classes have a definite number of models. We find the precision through the RESNET18 model, which goes about as the best base for most extreme accuracy.

An imbalance is made by picking arbitrary examples from the minority class with various irregularity proportions. The accuracy is found utilizing the RESNET18 model, which is the base accuracy.

The class imbalance is settled by adjusting the dataset, i.e., making the number of examples in class equivalent to the number of samples in the minority class, i.e., the class with fewer examples.

Different augmentation strategies are applied to the unbalanced data set, and different augmented balanced data sets are created. Accuracies are found utilizing the RESNET18 model. In light of the accuracies of a wide range of augmented balanced datasets, the augmentation strategies are being analyzed and looked at.

The model is run for 100 epochs, and the best test accuracies are recorded, then, at that point, contrasted with an investigation of the robust expansions for the dataset.

The augstatic library takes an input image, applies advanced augmentation methods to these images, and returns an augmented image. The library is superior to others as it has low computational expense dissimilar to TensorFlow or PyTorch increase libraries that convert pictures to a tensor, applying the transformation on the tensor and again transforming the array into an image.

Thus, to lessen this, we straightforwardly used the conversion on the image array and returned an augmented image.

It upholds many kinds of augmentations as follows:

- Blur
- CLAHE
- ChannelDropout
- ChannelShuffle
- ColorJitter
- Downscale
- Emboss
- FancyPCA
- GaussNoise
- GaussianBlur
- GlassBlur
- HueSaturationValue
- ISONoise
- InvertImg
- MedianBlur
- MotionBlur
- MultiplicativeNoise
- Posterize
- RGBShift
- Sharpen
- Solarize
- Superpixels
- ToGray
- ToSepia
- VerticalFlip
- HorizontalFlip
- Transpose
- OpticalDistortion
- GridDistortion
- JpegCompression
- Cutout
- CoarseDropout
- GridDropout

Figure 3. Effective & Non Effective Augmentations for Intel Scene Dataset

The description about each augmentations are as follows –

- Blur - Blur the input image using a random-sized kernel.
- CLAHE - Apply Contrast Limited Adaptive Histogram Equalization to the input image.
- ChannelDropout - Drops out a channel based on a range
- ChannelShuffle - Randomly rearrange channels of the input RGB image.
- ColorJitter - Randomly changes the brightness, contrast, and saturation of an image.
- Downscale - Decreases image quality by downscaling and upscaling back.
- Emboss - Emboss the input image and overlays the result with the original image.
- FancyPCA - Augment RGB image using FancyPCA from Krizhevsky's paper "ImageNet Classification with Deep Convolutional Neural Networks"
- GaussNoise - Apply gaussian noise to the input image.

Figure 4. Augmentations brief -1

- GaussianBlur - Blur the input image using a Gaussian filter with a random kernel size.
- GlassBlur - Apply glass noise to
- the input image.
- HueSaturationValue - Randomly change hue, saturation and value of the input image.
- ISONoise - Apply camera sensor noise.
- InvertImg - Invert the input image by subtracting pixel values from 255.
- MedianBlur - Blur the input image using a median filter with a random aperture linear size.
- MotionBlur - Apply motion blur to the input image using a random-sized kernel.
- MultiplicativeNoise - Multiply image to random number or array of numbers.
- Posterize - Reduce the number of bits for each color channel.
- RGBShift - Randomly shift values for each channel of the input RGB image.
- Sharpen - Sharpen the input image and overlays the result with the original image.
- Solarize - Invert all pixel values above a threshold.

Figure 5. Augmentations brief -2

- Superpixels - Transform images partially /completely to their superpixel representation. This implementation uses skimage's version of the SLIC algorithm.
- ToGray - Convert the input RGB image to grayscale. If the mean pixel value for the resulting image is greater than 127, invert the resulting grayscale image.
- ToSepia - Applies sepia filter to the input RGB image
- VerticalFlip - Flip the input vertically around the x-axis.
- HorizontalFlip - Flip the input horizontally around the y-axis.
- Transpose - Transpose the input by swapping rows and columns.
- OpticalDistortion - Distortion can be thought of as the difference in magnification across a field of view. This is usually calculated as a percentage of image size. By taking the measured distance in the image and comparing it to the predicted distance, we can calculate the optical distortion

Figure 6. Augmentations brief -3

- GridDistortion - Grid-distortion is an image warping technique which is driven by the mapping between equivalent families of curves, arranged in a grid structure. Until recently only curve sets arranged in a regular rectangular grid were considered.
- JpegCompression - Decrease Jpeg compression of an image.
- Cutout - CoarseDropout of the square regions in the image
- CoarseDropout - CoarseDropout of the rectangular regions in the image
- GridDropout - GridDropout, drops out rectangular regions of an image and the corresponding mask in a grid fashion.

Figure 7. Augmentations brief -2

IV. RESULTS AND DISCUSSION

This unbalanced dataset has a class imbalance proportion of 1:2. The balanced datasets were made by oversampling the minority class utilizing augmentation. These tests were run for Intel scene datasets. The outcomes were recorded, examined, and thought about for the different augmentation procedures and datasets that better the model accuracy. Augmentation techniques that are effective and ineffective in improving the model's accuracy by removing imbalance for Intel Scene Dataset are mentioned in fig 7.

INEFFECTIVE	EFFECTIVE
GaussianBlur ChannelDropout GridDropout Sharpen Solarize	FancyPCA GaussNoise InvertImg RGBShift HueSaturationValue JpegCompression MultiplicativeNoise

Figure 8. Effective & Non Effective Augmentations for Intel Scene Dataset

Therefore, from the above tables, we can derive that each dataset has various kinds of expansions that are viable. The ones like channel dropout are insufficient for the Intel scene as it has an ocean that is blue in variety. In the event that we apply channel dropout, one of the channels is haphazardly exited RGB, so the images become red or green. At the point when we train utilizing this, the model learn the ocean to be green or red. Accordingly, this sort of augmentation is ineffective as it changes the GT. The augmentation that is seen as effective for the dataset can be utilized for the individual datasets to further improve the RESNET18 model performance, considering the adam optimizer is utilized. The change of the model can diverge the augmentation strategies effective & ineffective to the respective datasets.

V. CONCLUSION AND FUTURE SCOPE

We aimed to simultaneously improve the model performance and resolve class imbalance problems using random sampling and advanced data augmentation techniques. The effective and non-effective augmentat1`ion techniques vary for each dataset for a single model. The augmented dataset included comparing many augmentation libraries. The effective ones are based on various parameters, building a new Augmentation library, and analyzing

each type of augmentation that the library supports. We balanced the two classes of original datasets by developing a Dataset Balance & created an imbalance randomly by using an unbalance creator. An augmentation Dataset Creator was built that used the Augmentation library. The datasets generated are used to train the RESNET18 model with additional layers to get the accuracy metrics – train & test accuracies. The model is run for 100 epochs for each test case & choosing the best-trained model using the test accuracies. These tests were run for three datasets. The results were analyzed & compared for the various augmentation techniques and three datasets. The resultant augmentations were recorded that improved the model accuracy. Therefore, the data scientist & ML Engineers can use the system & workflow to improve the model performance and help the model learn more with less data by implementing the effective augmentation for the datasets based on insights derived from this research.

REFERENCES

[1] Mateusz Buda, Atsuto Maki, Maciej A. Mazurowski (2018) “A systematic study of the class imbalance problem in convolutional neural networks” arXiv:1710.05381v2 [cs.CV]

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun (2015) “Deep Residual Learning for Image Recognition” arXiv:1512.03385v1 [cs.CV]

[3] Connor Taghi M. Khoshgoftaar (2019) “A survey on Image Data Augmentation for Deep Learning” Shorten and Khoshgoftaar J Big Data

[4] Michał Koziarski (2021) “Radial-Based Undersampling for Imbalanced Data Classification” arXiv:1906.00452v2 [cs.LG]

[5] Marcus D. Bloice, Christof Stocker, Andreas Holzinger (2017) “Augmentor: An Image Augmentation Library for Machine Learning” arXiv:1708.04680v1 [cs.CV]

[6] Allena Venkata Sai Abhishek, Sonali Kotni, 2021, Detectron2 Object Detection & Manipulating Images using Cartoonization, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 10, Issue 08 (August 2021),

[7] Allena Venkata Sai Abhishek, Venkateswara Rao Gurrala "AugStatic - A Light-Weight Image Augmentation Library", International Journal of Emerging Technologies and Innovative Research (www.jetir.org), ISSN: 2349-5162, Vol.9, Issue 5, page no.b735-b742, May-2022, Available :http://www.jetir.org/papers/JETIR2205199.pdf

[8] Allena Venkata Sai Abhishek, Dr. Venkateswara Rao Gurrala, Dr. Laxman Sahoo, "RESNET18 MODEL WITH SEQUENTIAL LAYER FOR COMPUTING ACCURACY ON IMAGE CLASSIFICATION DATASET", International Journal of Creative Research Thoughts (IJCRT), ISSN:2320-2882, Volume.10, Issue 5, pp.c176-c181, May 2022, Available at :http://www.ijcrt.org/papers/IJCRT2205235.pdf

[9] Allena Venkata Sai Abhishek, Dr. Venkateswara Rao Gurrala, "AUGMENTED BALANCED IMAGE DATASET GENERATOR USING AUGSTATIC LIBRARY", IJRAR - International Journal of Research and Analytical Reviews (IJRAR), E-ISSN 2348-1269, P- ISSN 2349-5138, Volume.9, Issue 2, Page No pp.536-543, May 2022, Available at : http://www.ijrar.org/IJRAR22B1901.pdf

[10] Allena Venkata Sai Abhishek and Venkateswara Rao Gurrala, Improving Model Performance and Removing the Class Imbalance Problem Using Augmentation, International Journal of Advanced Research in Engineering and Technology (IJARET). 13(5), 2022, pp. 14-22 doi: https://doi.org/10.17605/OSF.IO/CQMY5

[11] Alexandr A. Kalinin, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Alexander Buslaev (2020) “Albumentations: fast and flexible image Augmentations”

[12] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, David Lopez-Paz (2018) “MixUp augmentation for image classification”arXiv:1710.09412v2 [cs.LG]

[13] Zhiting Hu, Bowen Tan, Ruslan Salakhutdinov, Tom Mitchell, Eric P. Xing (2019) “Learning Data Manipulation for Augmentation and Weighting “ arXiv:1910.12795v1 [cs.LG]

[14] Shanchuan Lin, Linjie Yang, Imran Saleemi, Soumyadip Sengupta (2021) “Robust High-

Resolution Video Matting with Temporal Guidance" arXiv:2108.11515v1 [cs.CV]

[15] Wanwan Zheng, Mingzhe Jin (2020) "The Effects of Class Imbalance and Training Data Size on Classifier Learning" Springer

[16] Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton (2012) "ImageNet Classification with Deep Convolutional Neural Networks " NeurIPS Proceedings

[17] Elena Limonova, Daniil Alfonso, Dmitry Nikolaev, Vladimir V. Arlazarov (2020) "ResNet-like Architecture with Low Hardware Requirements" arXiv:2009.07190v2 [cs.CV]

[18] Abien Fred Agarap (2019) "Deep Learning using Rectified Linear Units (ReLU)" arXiv:1803.08375 [cs.NE]

[19] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, Ole Winther (2016) "Sequential Neural Models with Stochastic Layers" arXiv:1605.07571 [stat.ML]

[20] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natal ia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, Soumith Chintala (2019) "PyTorch: An Imperative Style, High-Performance Deep Learning Library" arXiv:1912.01703 [cs.LG]

[21] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, Xiaoqiang Zheng (2016) "TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems" arXiv:1603.04467 [cs.DC]