IMPROVING MODEL PERFORMANCE AND REMOVING THE CLASS IMBALANCE PROBLEM USING AUGMENTATION

Allena Venkata Sai Abhishek[1], Dr. Venkateswara Rao Gurrala [2]

[1] Research Scholar, Department of Computer Science and Engineering, GITAM University, Visakhapatnam, Andhra Pradesh, India

[2] Professor, Department of Computer Science and Engineering, GITAM University, Visakhapatnam, Andhra Pradesh, India

ABSTRACT

The data in the real world consists of various kinds of painful features. A majorly found one is the class imbalance in which the number of examples in different classes in a dataset is unequal. The class imbalance is being resolved using various sampling techniques on the data. Augmentation technique Augmentation is one of the essential steps in any machine learning pipeline and is used for oversampling the data of minority classes. This paper aims to improve the model performance is being enhanced with removing the class imbalance problem by using various Augmentation approaches to generate various balanced augmented datasets using various data augmentation techniques & random sampling. The accuracies are acquired for each augmentation technique using a RESNET18 model. The model is run up to 100 epochs for each case, and the best accuracies are compared. This iterative comparison of various augmentation techniques has shown stunning insights into the effectiveness of multiple datasets.

Key words: Class Imbalance, Augmentation, RESNET18, Random Sampling, Image Classification dataset.

1. INTRODUCTION

Image classification [11] is assigning the images to respective classes. It has an enormous repercussion in various domains of research. Deep learning has made the pipeline much more efficient and faster. With the widespread use of frameworks like PyTorch [18] and TensorFlow [19], neural networks are widely used in research work & companies like Google. The foremost step for training a model is collecting data. If the data is less, then the model cannot be able to learn enough and can lead to under-fitting or over-fitting [17]. Data augmentation [3][8] is used to generate new data using the existing data. This is cost-effective and time-saving as there is no need to extensively gathering data. The number of neural network models is exponentially increasing day by day with the increase in research in the field of neural networks. Neural networks are made up of neurons. Each layer consists of multiple nodes. And each node takes an input, applies a computation, and generates the output. Numerous types of neural networks are used for various purposes. Every kind of neural network model has a different function and gives varying perspectives on this ever-growing fast-changing domain. The study relates to analyzing the augmentation techniques that are effective and ineffective for various datasets while simultaneously improving the model performance with fixed test data and model. We incorporated a new approach to iteratively test the model with different augmented image datasets while removing class imbalance [1][4][10] with oversampling [13][20] is the last objective of this introduction. GANs [12] are also used for generating synthetic images.

2. BACKGROUND

There are many augmentations supported by various augmentation libraries that help in generating new samples. There are many image classification datasets. The improvement of the model performances varies for each dataset, augmentation, and model used. To find the best augmentation technique based on accuracy using a RESNET18 model [2][14] using a custom-built augmentation library that will generate an augmented balanced dataset, solving the class imbalance problem. The challenge is to help the model learn from fewer data. For which the data augmentation techniques are considered for oversampling. Many augmentation libraries and concepts like mixup augmentation [7], matting [9], torchvision transforms, augmentor [5], and albumentations [6] were compared to choose the efficient package to use the augmentation techniques as shown in Fig 1.

Figure 1. Augmentation Techniques

Residual networks preserve the data and help the model learn more data as they have skip connections. They have many variants based on the number of layers shown in fig 2.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	[3×3, 64; 3×3, 64] ×2	[3×3, 64; 3×3, 64] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3	[1×1, 64; 3×3, 64; 1×1, 256] ×3
conv3_x	28×28	[3×3, 128; 3×3, 128] ×2	[3×3, 128; 3×3, 128] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×4	[1×1, 128; 3×3, 128; 1×1, 512] ×8
conv4_x	14×14	[3×3, 256; 3×3, 256] ×2	[3×3, 256; 3×3, 256] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×6	[1×1, 256; 3×3, 256; 1×1, 1024] ×23	[1×1, 256; 3×3, 256; 1×1, 1024] ×36
conv5_x	7×7	[3×3, 512; 3×3, 512] ×2	[3×3, 512; 3×3, 512] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3	[1×1, 512; 3×3, 512; 1×1, 2048] ×3
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

ures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of block

Figure 2. Building blocks

3. PROPOSED METHODOLOGY

3.1. Data Collection

In order to compare the augmentation techniques used to improve the model performance while simultaneously removing the class imbalance problem, we created balanced datasets based on Intel Scene Dataset, CIFAR10, and Chest X-Ray Pneumonia datasets. Two classes were chosen to analyze the multiple classes in the datasets. For the Intel scene dataset, sea and buildings classes were chosen. For the Chest X-Ray Pneumonia dataset, Normal and Pneumonia classes were used. For the CIFAR10 dataset, cat and dog, classes were chosen.

3.2 Workflow

A classification image data set is used that has a train and test set. For the sole purpose of training a model, it is done with the usage of a train set. The model that is trained is tested upon a fixed test set, and the accuracies are stored in a log file for our records. The datasets are used to train the model, and the accuracies are acquired from the model by testing upon the test set.

Two classes are chosen out of the dataset & are balanced manually such that both the classes have an exact number of examples. We find the accuracy by passing it through the RESNET18 model, which acts as the ideal base for maximum accuracy.

An imbalance is created by choosing random samples from the minority class with different imbalance ratios. The accuracies are found using the RESNET18 model, which acts as the minimum base accuracy.

The class imbalance is resolved by balancing the dataset, i.e., making the number of samples in class equal to the number of examples in the minority class, i.e., the class with the lesser number of samples.

Over-sampling removes the imbalance by generating new images with existing images in the minority class. Various data augmentation is a method used for oversampling. For which transformations are made upon existing images such as rotation, scale, etc. For example, suppose there is a cat image, we make a new augmented image in which it has undergone some transformations like we can rotate or flip or increase the brightness or increase the contrast, etc. So, different types of augmentation techniques are applied to create new images that help the model learn even with less data.

Various augmentation techniques are applied to the unbalanced data set, and different augmented balanced data sets are generated. Accuracies are found using the RESNET18 model. Based on the accuracies of all types of augmented balanced datasets, the augmentation techniques are being analyzed and compared.

The model is run for 100 epochs, and the best test accuracies are recorded, then compared to analyze the effective augmentations for the dataset. The workflow is explained in fig 3.

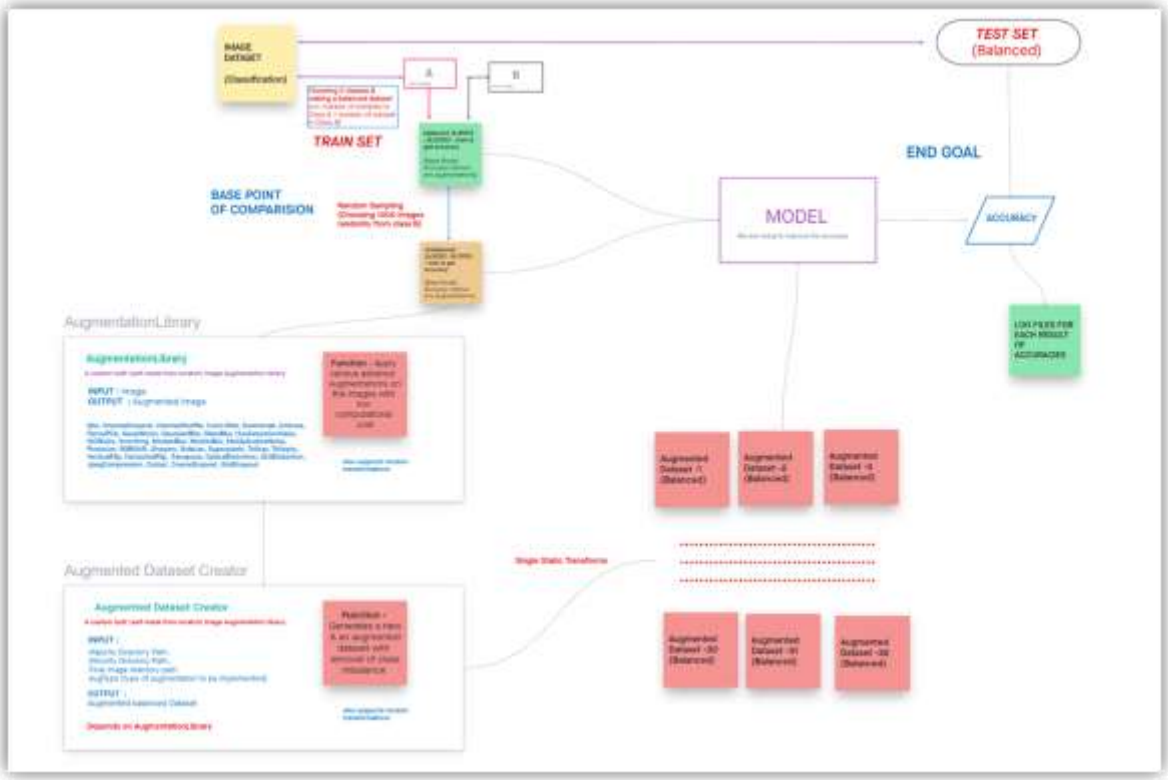

Figure 3. System Diagram

There are many types of augmentation techniques. And, each augmentation techniques generate a balanced dataset, removing the class imbalance.

The augmentations that were analyzed in this research are mentioned in fig 4.

- 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 4. List of Augmentations compared

3.3 Model Architecture

The neural network model used is RESNET18. It has 18 layers and consists of skip connections used to help the model learn more. The one residual block consists of two weight layers & has a skip connection for every residual block connecting the output of the second weight layer with a ReLU activation function [15]. An additional sequential layer [16] has been added for getting the accuracies in the range of 0 to 1. We use the PyTorch framework to build models. The RESNET 18 is not pre-trained. Adam optimizer is being used & negative log-likelihood loss function is used. The RESNET18 architecture is mentioned in fig 5 and 6.

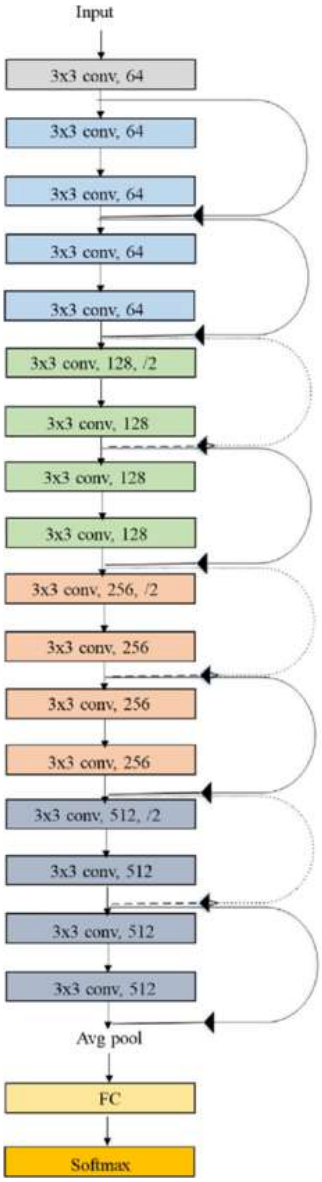

Figure 5. ResNet-18 Architecture

```
(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 6. Additional sequential layer

An additional sequential layer is added at the end, as mentioned in fig 6. The input will be passed to linear (512,512), whose output is fed to the first ReLU activation function. Then a dropout (0.2) layer is being used, followed by linear (512, 2). Finally, it is passed through a LogSoftmax to get the logarithm of the probabilities.

4. RESULTS AND DISCUSSIONS

This unbalanced dataset has a class imbalance ratio of 1:2, 1:4, 1:8, 1:16, and 1:20. The balanced datasets were created by oversampling the minority class using augmentation. These tests were run for three datasets. The results were recorded, analyzed & compared for the various augmentation techniques & for datasets that improved the model accuracy.

Augmentation techniques that are effective and ineffective in improving the model's accuracy by removing imbalance are mentioned in fig 7, fig 8, and fig 9.

a) For Intel Scene Dataset

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

Figure 7. Effective & Non Effective Augmentations for Intel Scene Dataset

b) For Chest Xray Pneumonia Dataset

INEFFECTIVE	EFFECTIVE
GaussianBlur Glass Blur ChannelDropout GridDropout Sharpen Solarize	VerticalFlip Transpose GaussNoise MotionBlur HueSaturationValue

Figure 8. Effective & Non Effective Augmentations for Chest Xray Pneumonia Dataset

c) For CIFAR 10 Dataset

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

Figure 9. Effective & Non Effective Augmentations for CIFAR 10 Dataset

Therefore, from the above tables, we can infer that each dataset has different types of augmentations that are effective. The ones like channel dropout are ineffective for the Intel scene as it has a sea that is blue in color. If we apply channel dropout, one of the channels is randomly dropped out of RGB, so the images become red or green. When we train using this, the model will learn the sea to be green or red. Thus, this type of augmentation is ineffective as it changes the GT.

The augmentation that is found effective for the dataset can be used for the respective datasets to improve the RESNET18 model performance, considering the apt optimizer is used. The change in the model can diverge the augmentation techniques effective & ineffective to the respective datasets.

5. CONCLUSIONS

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

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

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

[9] Shanchuan Lin, Linjie Yang, Imran Saleemi, Soumyadip Sengupta (2021) "Robust High-Resolution Video Matting with Temporal Guidance" arXiv:2108.11515v1 [cs.CV]

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

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

[12] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio "Generative Adversarial Networks " arXiv:1406.2661v1 [stat.ML]

[13] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer (2002) "SMOTE: Synthetic Minority Over-sampling Technique" Journal of Artificial Intelligence Research 16

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

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

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

[17] Shaeke Salman, Xiuwen Liu (2019) “Overfitting Mechanism and Avoidance in Deep Neural Networks” arXiv:1901.06566 [cs.LG]

[18] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia 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]

[19] 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]

[20] R. Mohammed, J. Rawashdeh and M. Abdullah, (2020) "Machine Learning with Oversampling and Undersampling Techniques: Overview Study and Experimental Results," 2020 11th International Conference on Information and Communication Systems (ICICS), 2020, pp. 243-248, doi: 10.1109/ICICS49469.2020.239556.