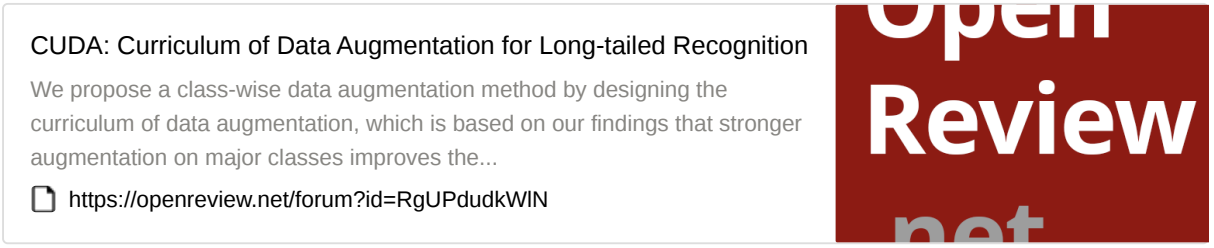
Reproducibility Track Proposal

Selected Research Paper : CUDA: Curriculum of Data Augmentation for Long-Tailed Recognition

Conference : ICLR 2023 notable top 25%

Link to Paper :

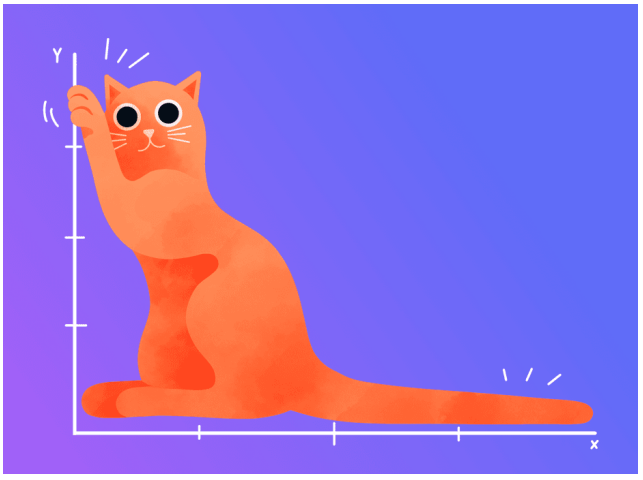


Objective:

Long tailed recognition is one of the most challenging problems in visual recognition,it aims to train well-performing models from a large number of images that follow a long-tailed class distribution.Cuda just proposes a data augmentation strategy that can be used on top of any other LTR models to improve overall performance for datasets like ImageNet-LT , CIFAR-100-LT**.**

The reasoning for choosing the paper is due to how fundamental the imbalence issue is in Computer vision tasks.Popular datasets like ImageNet are artificially balenced to solve it ,but this occurs at cost of downsampling our training data. The algorithm that they propose is very easy to understand and doesn't involve any complex code so I hope it will be easy to reproduce.

Understanding of the paper:



1 Long Tailed Recognition

In training data derived from real life statistics we may observe that there is a imbalence between the no of samples belonging to different classes,there will be a Head classes containing majority of the training instances and tail classes containing lesser no of instances.Example for this type of distribution will be a dataset for binary disease screening test where Non patients are in a majority(HEAD) and Patients are in a minority(TAIL).In this case,the performance of deep learning models is often dominated by the head classes while the learning of the tail classes is severely underdeveloped.(A Survey on Long-tailed Visual Recognition).

Solutions to Long Tailed recognition primary involves three methods:

- Resampling:Upsampling minority classes or Downsampling majority classes
- Reweighting:rebalancing the loss to give more weights to minority classes
- Transfer learning:enriching the information of minority classes by transferring information gathered from majority classes to the minority ones.

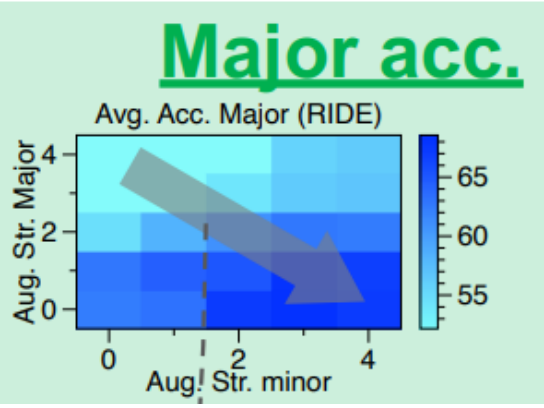
2 Key Finding of Paper

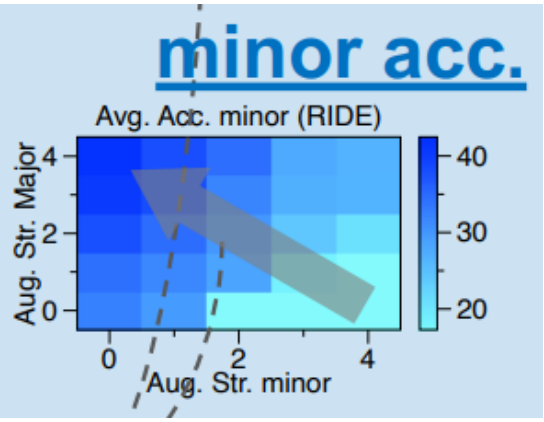
Data augmentation involves applying augmentation operations (shifting,rotating, brightening) to artificaially increase the amount of data.

Intuitive belief(what the previous paper trying to solve LTR follow) : To solve Long tail phenomeneon we would use data augmentation **(1)** to increase the no of samples in the minority class and **(2)** we would need to use a higher degree of augmentation (higher magnitude of rotation,higher magnitude of shifting etc) on the minority class to ensure that the copies are disinct from the original. We could also use data augmentation on the majority classes but both on a **(3)** lesser extent (lesser no of copies) and a **(4)** lesser degree(magnitude) of augmentation.

The key finding of the paper was against the **(2)** and **(4)** statements of the above para. ***"class-wise augmentation improves performance in the non-augmented classes while that for the augmented classes may not be significantly improved, and in some cases, performances may even decrease."***

Basically if we increase the augmenation strength on minority class(move along y axis) keeping augmenation strength on majority constant we see that in the left heatmap that the majority accuracy increases and in the right heatmap that minority accuracy decreases.





3 Why Cuda is needed?

From the above heat map we can see that if we:

- Use very strong augmentation on minority class,the accuracy of minority is low,lowering accuracy of test data(which has equal instances of all classes)
- Use very weak augmentation on minority class,the accuracy of majority is low,lowering accuracy of test data(which is balenced)
- So its basically like a trade off between minor acc and major acc.

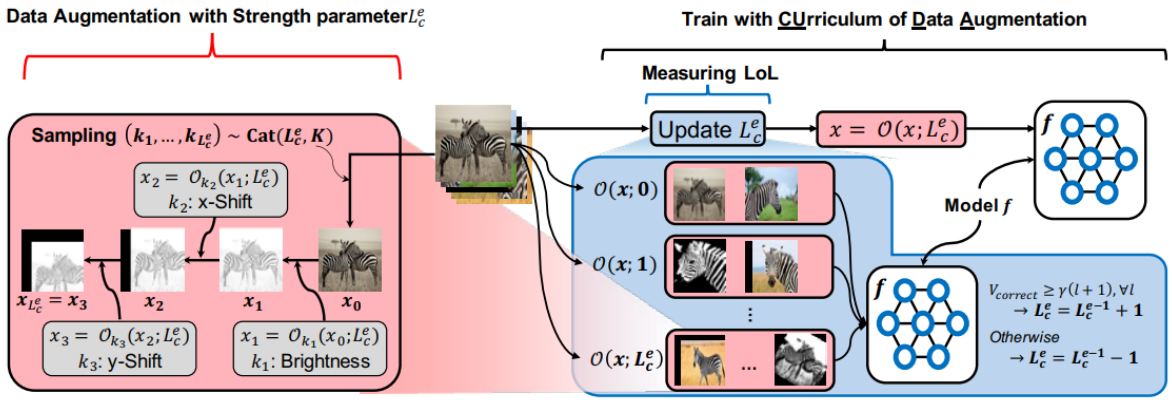
Hence we need a solution to find the appropriate augmentation strength for each class in the dataset which is (drum rolls)…… **CUDA:CUrriculum for Data Augmentation** a simple algorithm to
find the proper class-wise augmentation strength for long-tailed recognition. Based on our motivation,we have to increase the augmentation strength of majorities for the performance of minorities when the model successfully predicts the majorities. On the other hand, we have to lower the strength of majorities when the model makes wrong predictions about majorities.The paper was the first to suggest a class-wise augmentation method to find a proper augmentation strength for class imbalance problem.

The core philosophy of CUDA is to “generate an augmented sample that becomes the most difficult
sample without losing its original information.”

CUDA uses two main parts to achieve this:

(1) Strength-based augmentation

(2) Using Level-of-Learning (LoL) score.



4 Strength-Based augmentation

We need to define a metric to quantify how complex augmentations we are applying on an image.(there wasn’t any other well known metric when the paper was released)

Let us assume that there exist pre-defined K augmentation operations. We utilize visual augmentation operations which is indexed as k ∈ {1, · · · , K}, e.g., Gaussian blur, Rotation, Horizontal flip. However we won’t use all K operations always.Each augmentation operation $O_k^{mk(s)}$ has its own predefined augmentation magnitude function mk(s) with the strength parameter s ∈ {0, ..., S}. These operations are described in detail along with each magnitude functions in Appendix D of the paper.

Given an augmentation strength parameter s and an input image x, we model a sequence of augmentation operations O(x; s) as follows:

$$O(x; s) = O_{ks}^{mks(s)} \circ O_{ks-1}^{mks-1(s)} \circ \dots \circ O_{k1}^{mk1(s)}(x)$$

The sequential augmentation operation O(x; s) samples s operations from the categorical distribution when the probability of seeing the operations follows uniform distribution.Basically out of K possible operations we only take s operations and use them with magnitude : m(3) .Suppose we have s=3 ,let the 3 augmentations operations k1, k2, and k3 are brightness, X-shift, and Y-shift, respectively. Then, O(x; 3) outputs an image in

which bright is raised by mbright(3) and
moved by mx-shift(3) on the x-axis and shifted by my-shift(3) on the y-axis

When s increases both the no of augmentation operations applied to each image and the magnitude of each of these operation increases,that is the complexity of the augmentation increases.

5 Level of Learning

Algorithm 1: **CU**rriculum of **D**ata **A**ugmentation

Input: LTR algorithm $\mathcal{A}_{\text{LTR}}(f, \mathcal{D})$, training dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, train epochs E, aug. probability p_{aug}, threshold γ, number of sample coefficient T.
Output: trained model f_θ
Initialize: $L_c^0 = 0 \; \forall c \in \{1, ..., C\}$
for $e \leq E$ **do**
  Update $L_c^e = V_{\text{LoL}}(\mathcal{D}_c, L_c^{e-1}, f_\theta, \gamma, T) \quad \forall c$ `// Alg. 2`
  Generate $\mathcal{D}_{\text{CUDA}} = \{(\bar{x}_i, y_i) | (x_i, y_i) \in \mathcal{D}\}$ where

$$\bar{x}_i = \begin{cases} \mathcal{O}(x_i, L_{y_i}^e) & \text{with prob. } p_{\text{aug}} \\ x_i & \text{otherwise.} \end{cases}$$

  Run LTR algorithm using $\mathcal{D}_{\text{CUDA}}$, *i.e.*, $\mathcal{A}_{\text{LTR}}(f_\theta, \mathcal{D}_{\text{CUDA}})$.
end

Algorithm 2: V_{LoL}: Update LoL score

Input: $\mathcal{D}_c$, L, f_θ, γ, T
Output: updated L
Initialize: check $= 1$
for $l \leq L$ **do**
  /* $V_{\text{correct}}(\mathcal{D}_c, l, f_\theta, T)$ */
  Sample $\mathcal{D}'_c \subset \mathcal{D}_c$ s.t. $|\mathcal{D}'_c| = T(l+1)$
  Compute $v = \sum_{x \in \mathcal{D}'_c} \mathbb{1}_{\{f(\mathcal{O}(x;l))=c\}}$
  if $v \leq \gamma T(l+1)$ **then**
    check $\leftarrow 0$; **break**
  end
end
if check $= 1$ **then** $L \leftarrow L + 1$
else $L \leftarrow L - 1$

To control the strength of augmentation properly, we check whether the model can correctly predict augmented versions without losing the original information.To enable this, we define the LoL for each class c at epoch e, i.e., L_c^e which is adaptively updated as the training continues as follows:

Function call:

$$L_c^e = V_{\text{LoL}}(\mathcal{D}_c, L_c^{e-1}, f_\theta, \gamma, T),$$

Eq 1

$$V_{\text{LoL}}(\mathcal{D}_c, L_c^{e-1}, f_\theta, \gamma, T) = \begin{cases} L_c^{e-1} + 1 & \text{if } V_{\text{Correct}}(\mathcal{D}_c, l, f_\theta, T) \geq \gamma T(l+1) \quad \forall l \in \{0, ..., L_c^{e-1}\} \\ L_c^{e-1} - 1 & \text{otherwise} \end{cases}$$

Eq 2

$$V_{\text{Correct}}(\mathcal{D}_c, l, f_\theta, T) = \sum_{x \in \mathcal{D}'_c} \mathbb{1}_{\{f_\theta(\mathcal{O}(x;l))=c\}} \quad \text{where } \mathcal{D}'_c \subset \mathcal{D}_c.$$

- L_c^e is initialized to zero across all values of c before training.Lets say we start the third epoch with value of L_c^2 (LOL value for class c after second iteration) for one specific value of c to be 2
- Our first step would be update the value of lol for each class.We call the function V_{lol} which takes inputs: D_c all images belonging to class c , L_c^2 previous value of LOL, f_θ the model we are gonna use for prediction, γ ∈ [0, 1] is threshold hyperparameter , T is coefficient of the number of samples used to updating LoL.
- Inside the function V_{lol} we run the loop for all values of l lesser than L_c^2 =2 which mean we run it for l=0,1,2
    - We randomly sample T(l+1) samples out of D_c which is $D_c^{'}$ D′c ⊂ Dc s.t. |D′c| = T(l + 1)
    - **Eq2** ⇒ We compute $V_{correct} = v$ in which we make the model f_θ predict the class for all samples in $D_c^{'}$ and find the total no of correct predictions among the T(l+1) samples.
- **Eq1** ⇒ If the no of correct predictions is above the threshold γT(l+1) (i.e if the ratio of correct samples/total samples is above γ) for all values of l (0,1,2 in this case)increase the value of Lol by 1 (3 in this case) otherwise it would be decreased by 1 (1 in this case) for the next epoch
- Similarly if we run the function V_{lol} for all classes we could then define a sequence of augmentation operations $O(x_i, L_{y_i}^e)$ with each having their strengths defined by the respective $L_{y_i}^e$ where each y_i is a class .We create a new training dataset by

$$\mathcal{D}_{\text{CUDA}} = \{(\bar{x}_i, y_i) | (x_i, y_i) \in \mathcal{D}\} \text{ where}$$

$$\bar{x}_i = \begin{cases} \mathcal{O}(x_i, L_{y_i}^e) & \text{with prob. } p_{\text{aug}} \\ x_i & \text{otherwise.} \end{cases}$$

- We can train whatever LTR algorithm (the model + preprocessing) on the new dataset and get the overal accuracy for that epoch.

Challenges and Code Availability:

A complete code implementation of CUDA along with the SOTA LTR algorithms as of 2022 for Imagenet LT , Cifar -100 LT and inaturalist has been provided in github.However the author has not stated the dependencies anywhere.

All of the dependencies have been assumed from the depndencies of the LTR algorithms,some minor changes in code have been done (like np.int → int).

Dataset Cifar-100 LT contains only 60000 images but Imagenet LT and i naturalist consist of **115.8K and 675,170 images hence may take more time.**

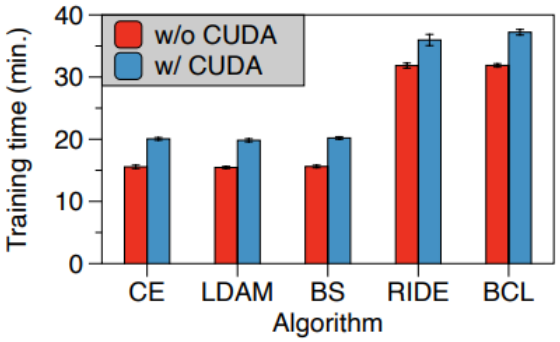


Figure 11: Training time.

The training time of various LTR Algorithms with CUDA is a constraint..

There is no code provided for ablation study but i guess it will be mostly doable..

Differnet LTR algorithms are paired with CUDA their requirements and dependecies must also be followed.

Experiment 1:

Cifar 100 LT

Algorithm	Imbalance Ratio (IR)			Statistics (IR 100)		
	100	50	10	Many	Med	Few
CE	$38.7_{\pm 0.4}$	$43.4_{\pm 0.3}$	$56.5_{\pm 0.6}$	$66.2_{\pm 0.5}$	$37.3_{\pm 0.6}$	$8.2_{\pm 0.3}$
CE + CMO (Park et al., 2022)	$42.0_{\pm 0.4}$	$47.0_{\pm 0.5}$	$60.0_{\pm 0.4}$	$69.1_{\pm 0.4}$	$41.2_{\pm 0.6}$	$11.3_{\pm 0.7}$
CE + CUDA	$42.7_{\pm 0.4}$	$47.2_{\pm 0.4}$	$59.6_{\pm 0.4}$	$\mathbf{71.6_{\pm 0.6}}$	$42.3_{\pm 0.3}$	$9.4_{\pm 0.7}$
CE + CMO + CUDA	$43.5_{\pm 0.5}$	$48.7_{\pm 0.6}$	$60.0_{\pm 0.3}$	$70.0_{\pm 0.7}$	$43.4_{\pm 0.5}$	$12.7_{\pm 0.8}$
CE-DRW (Cao et al., 2019)	$41.4_{\pm 0.2}$	$45.5_{\pm 0.6}$	$57.8_{\pm 0.6}$	$62.8_{\pm 0.5}$	$41.7_{\pm 0.7}$	$16.1_{\pm 0.4}$
CE-DRW + Remix (Chou et al., 2020)†	45.8	49.5	59.2	-	-	-
CE-DRW + CUDA	$47.7_{\pm 0.4}$	$52.4_{\pm 0.5}$	$61.6_{\pm 0.5}$	$64.3_{\pm 0.4}$	$49.2_{\pm 0.6}$	$26.7_{\pm 0.6}$
LDAM-DRW (Cao et al., 2019)	$42.5_{\pm 0.2}$	$47.4_{\pm 0.5}$	$57.6_{\pm 0.1}$	$62.8_{\pm 0.5}$	$42.3_{\pm 0.6}$	$19.0_{\pm 0.7}$
LDAM + M2m (Kim et al., 2020)‡	43.5	-	57.6	-	-	-
LDAM-DRW + CUDA	$47.6_{\pm 0.7}$	$51.1_{\pm 0.4}$	$58.4_{\pm 0.1}$	$67.3_{\pm 0.6}$	$50.4_{\pm 0.5}$	$21.4_{\pm 0.2}$
BS (Ren et al., 2020)	$43.3_{\pm 0.4}$	$46.9_{\pm 0.2}$	$58.3_{\pm 0.4}$	$61.6_{\pm 0.8}$	$42.3_{\pm 0.5}$	$23.0_{\pm 0.4}$
BS + CUDA	$47.7_{\pm 0.3}$	$52.1_{\pm 0.4}$	$61.7_{\pm 0.5}$	$63.3_{\pm 0.4}$	$48.4_{\pm 0.4}$	$28.7_{\pm 0.5}$
RIDE (3 experts) (Wang et al., 2021)†	48.6	51.4	59.8	-	-	-
RIDE (3 experts)	$49.7_{\pm 0.2}$	$52.7_{\pm 0.1}$	$60.2_{\pm 0.2}$	$67.7_{\pm 0.6}$	$51.5_{\pm 0.5}$	$26.7_{\pm 0.6}$
RIDE + CMO (Park et al., 2022)†	50.0	53.0	60.2	-	-	-
RIDE + CMO	$49.9_{\pm 0.1}$	$53.0_{\pm 0.1}$	$58.9_{\pm 0.3}$	$67.3_{\pm 0.3}$	$51.3_{\pm 0.6}$	$28.1_{\pm 0.4}$
RIDE (3 experts) + CUDA	$50.7_{\pm 0.2}$	$53.7_{\pm 0.4}$	$60.2_{\pm 0.1}$	$69.2_{\pm 0.3}$	$52.8_{\pm 0.2}$	$27.3_{\pm 0.8}$
BCL (Zhu et al., 2022)⋆	51.0	54.9	64.4	67.2	53.1	32.9
BCL + CUDA	$\mathbf{52.3_{\pm 0.2}}$	$\mathbf{56.2_{\pm 0.4}}$	$\mathbf{64.6_{\pm 0.1}}$	$66.4_{\pm 0.2}$	$\mathbf{54.2_{\pm 0.6}}$	$\mathbf{33.9_{\pm 0.8}}$

Experiment 2:

i naturalist 2018 and Image Net LT

Algorithm	ImageNet-LT				iNaturalist 2018			
	Many	Med	Few	All	Many	Med	Few	All
CE†	64.0	33.8	5.8	41.6	73.9	63.5	55.5	61.0
CE + CUDA	**67.2**$_{\pm 0.1}$	47.0$_{\pm 0.2}$	13.5$_{\pm 0.3}$	47.3$_{\pm 0.2}$	**74.6**$_{\pm 0.3}$	64.9$_{\pm 0.1}$	57.2$_{\pm 0.1}$	62.5$_{\pm 0.2}$
CE-DRW (Cao et al., 2019)	61.7$_{\pm 0.1}$	47.1$_{\pm 0.3}$	29.0$_{\pm 0.3}$	50.1$_{\pm 0.1}$	68.2$_{\pm 0.2}$	67.3$_{\pm 0.2}$	66.4$_{\pm 0.1}$	67.0$_{\pm 0.1}$
CE-DRW + CUDA	61.8$_{\pm 0.1}$	48.3$_{\pm 0.1}$	30.3$_{\pm 0.2}$	51.0$_{\pm 0.1}$	68.8$_{\pm 0.1}$	68.1$_{\pm 0.3}$	66.6$_{\pm 0.2}$	67.5$_{\pm 0.1}$
LWS (Kang et al., 2020)‡	57.1	45.2	29.3	47.7	65.0	66.3	65.5	65.9
cRT (Kang et al., 2020)‡	58.8	44.0	26.1	47.3	69.0	66.0	63.2	65.2
cRT + CUDA	62.3$_{\pm 0.1}$	47.2$_{\pm 0.2}$	28.4$_{\pm 0.5}$	50.2$_{\pm 0.2}$	68.2$_{\pm 0.1}$	67.8$_{\pm 0.2}$	66.4$_{\pm 0.1}$	67.3$_{\pm 0.1}$
LDAM-DRW (Cao et al., 2019)†	60.4	46.9	30.7	49.8	-	-	-	66.1
LDAM-DRW + CUDA	63.1$_{\pm 0.1}$	48.0$_{\pm 0.3}$	31.1$_{\pm 0.2}$	51.4$_{\pm 0.1}$	67.8$_{\pm 0.2}$	67.6$_{\pm 0.2}$	66.7$_{\pm 0.3}$	67.2$_{\pm 0.2}$
BS (Ren et al., 2020)	61.1$_{\pm 0.2}$	48.5$_{\pm 0.2}$	31.8$_{\pm 0.4}$	50.9$_{\pm 0.1}$	65.5$_{\pm 0.2}$	67.5$_{\pm 0.1}$	67.5$_{\pm 0.1}$	67.2$_{\pm 0.2}$
BS + CUDA	61.9$_{\pm 0.1}$	49.2$_{\pm 0.0}$	32.3$_{\pm 0.4}$	51.6$_{\pm 0.1}$	67.6$_{\pm 0.1}$	68.3$_{\pm 0.1}$	68.3$_{\pm 0.1}$	68.2$_{\pm 0.1}$
RIDE (3 experts) (Wang et al., 2021)⋆	64.8$_{\pm 0.1}$	50.8$_{\pm 0.2}$	34.6$_{\pm 0.2}$	53.6$_{\pm 0.1}$	70.4$_{\pm 0.1}$	71.8$_{\pm 0.1}$	71.7$_{\pm 0.1}$	71.6$_{\pm 0.1}$
RIDE + CMO (Park et al., 2022)⋆	65.6	50.6	34.8	54.0	68.0	70.6	72.0	70.9
RIDE (3 experts) + CUDA⋆	65.9$_{\pm 0.1}$	51.7$_{\pm 0.2}$	34.9$_{\pm 0.2}$	54.7$_{\pm 0.1}$	70.7$_{\pm 0.2}$	72.5$_{\pm 0.1}$	**72.7**$_{\pm 0.2}$	**72.4**$_{\pm 0.2}$
BCL (Zhu et al., 2022)	65.3$_{\pm 0.2}$	53.5$_{\pm 0.2}$	36.3$_{\pm 0.3}$	55.6$_{\pm 0.2}$	69.5$_{\pm 0.1}$	72.4$_{\pm 0.2}$	71.7$_{\pm 0.1}$	71.8$_{\pm 0.1}$
BCL + CUDA	66.8$_{\pm 0.1}$	**53.9**$_{\pm 0.3}$	**36.6**$_{\pm 0.2}$	**56.3**$_{\pm 0.1}$	70.9$_{\pm 0.2}$	**72.8**$_{\pm 0.1}$	72.0$_{\pm 0.1}$	72.3$_{\pm 0.1}$

Ablation Study:

1) How std of weight L1 norm between each class varies when we use Cuda against standard augmentation practices.

The classifier weight norm is usually used to measure how balanced the model consider the
input from a class-wise perspective (Kang et al., 2020; Alshammari et al., 2022).Basically if variance in L1 weight norm between each class deceases we have a more baleced dataset

2) How LOL scores look after training?

Does our inference on how accuracy varies with augmentation strength still hold.

3) How accuracy changes with hyperparameter T the coefficient of number of samples?

4) How accuracy changes with hyperparameter γ ∈ [0, 1] threshold ratio of correct predictions?

5) How accuracy changes with hyperparameter $p_{aug} \in$ [0, 1] probability of augmented images?

6) How accuracy changes with K the total number of available augmentation oprations.?

In the paper K=22 operations are used but what if we only have access to k=3 operations

7) How accuracy changes if we don't use curriculum learning?

Curriculum learning is where we first give the model easier images(lesser strength) to learn and then give harder images(higher strength) where when we update LOL we check the threshold for all l ≤ L rather than just l=L.

8) What happens if we don't apply a classwise LOL L_c^e but just have a common L^e which we optimize(which is totatally against our idea of classwise strength)?

Methodology:

First we will follow the code provided in the github implementation, set up the 2 given experiments in the paper, and compare the results.We will document any deviations, uncertainties observed during the reproduction process.

The code will be implemented in a Kaggle/Jupyter Notebook with 2 T4 GPU as accelerators (as of now), but I will try exploring better tech like AWS for deploying,Anaconda Navigator, Azure etc. Code Implementation of keeping a log of the arguments and the results must be done .

Next Ablation studies which only involve varying a parameter will be done and ther graphs plotted to systematically analyze the impact of individual components

Finally Ablation studies which require new code to be added or modified will be done .

After the deadline ends further analysis can be done on ViT (the current SOTA after this paper was released)to check wether Cuda can increase their performance.

Timeline:

27 december : Understanding the code implementation, Setting up

30 december : The first experiment

2 Jan : The second experiment

6 Jan : Ablation studies which only involve varying a parameter

13 Jan : Ablation studies which require new code to be added or modified

After 20 Jan : further analysis with ViT

Expected Outcomes:

1)Better performance than the SOTA for LTR (as of 2022 which are stated in the paper) Cifar-100-LT,ImageNet-LT, i-naturalist

2) Cuda decreases std of weight L1 norm compared to standard augmenation

3)After training majority class must have higher LOL score

4)Accuracy doesnt vary much with hyper parameter Num test T

5)The hyperparmater $p_{aug} \in [0, 1]$ shows a maxima inbetween 0 and 1 rather than at 1 where we basically only have augmented images and no original images.

6)Hyper parameter Accept rate γ has a maxima at 0.5 where the no of correct predictions is half of the samples we give it.

7)Without Curriculum learning accuracy must drop

8)Without Classwise LOL accuracy must decrease