Revisiting Adversarial Training under Long-Tailed Distributions

Supplementary Material

A. Implementation Details of Experiments

A.1. Details of Table 1

All parameters in the experiments presented in Table 1 are consistent with the specifications of RoBal [45]. Key configurations include an initial learning rate of 0.1, with a decay factor of 10 applied at the 60th and 75th epochs, culminating in a total training period of 80 epochs. An SGD momentum optimizer is employed with a weight decay of 2×10^{-4}. The batch size is maintained at 64. For the adversarial training, we adopt a maximum perturbation of $8/255$ and a step size of $2/255$, with the internal maximization process involving 5 iterations, denoted as PGD-5. For CIFAR-10-LT, the parameters include $m_0 = 0.1$, scaling factor $s = 10$, bias term $\tau_b = 1.5$, and margin term $\tau_m = 0.3$; for CIFAR-100-LT, the parameters include $m_0 = 0.3$, $s = 10$, $\tau_b = 1.5$, and $\tau_m = 0.3$. The specific hyperparameters for each experiment are detailed in Table 8.

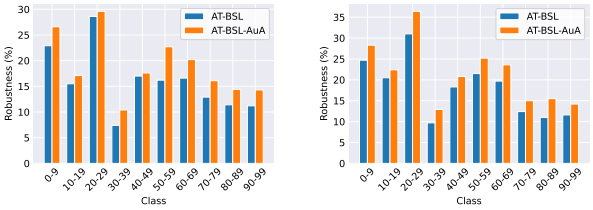
A.2. Details of Data Augmentation Approaches

Data augmentation techniques such as MixUp [51], Cutout [9], CutMix [49], Augmix [17], AutoAugment (AuA) [6], RandAugment (RA) [7], and TrivialAugmen (TA) [32] are employed utilizing the implementations provided in torchvision 0.16.0[1]. Regarding integrating data augmentation into the adversarial training pipeline, we follow the approach outlined in [34], whereby data augmentation precedes the generation of adversarial examples through adversarial attacks. It is observed that reversing this order, i.e., performing data augmentation after adversarial attacks, leads to the disruption of adversarial perturbations, significantly diminishing the effectiveness of the adversarial attacks.

A.3. Code References

For the defense methods compared in our paper, we utilize the official code releases, including AT [30][2], TRADES [52][3], MART [41][4], AWP [44][5], GAIRAT [53][6], LAS-AT [19][7], RoBal [45][8], and REAT [25][9]. Regarding the attacks used for evaluation, we implement them by referring to several official code repositories and the original papers, encompassing FGSM [12], PGD [30], CW [3], and AutoAttack [5][10].

[1] https://github.com/pytorch/pytorch
[2] https://github.com/MadryLab/cifar10_challenge
[3] https://github.com/yaodongyu/TRADES
[4] https://github.com/YisenWang/MART
[5] https://github.com/csdongxian/AWP
[6] https://github.com/zjfheart/Geometry-aware-Instance-reweighted-Adversarial-Training
[7] https://github.com/jiaxiaojunqaq/las-at
[8] https://github.com/wutong16/Adversarial_Long-Tail
[9] https://github.com/GuanlinLee/REAT



(a) (b)

Figure 8. The class-wise robustness under AA for various algorithms on CIFAR-100-LT at the best checkpoint. (a) ResNet-18; (b) WideResNet-34-10.

B. Extensive Experiments

B.1. More Ablation Studies of RoBal

In addition to the experiments conducted on ResNet-18 and CIFAR-10-LT as presented in Table 1, we extend our ablation studies to include WideResNet-34-10 and CIFAR-100-LT, as illustrated in Tables 9, 10, and 11. The results align with the conclusions drawn from Table 1, demonstrating that AT-BSL achieves comparable performance in terms of clean accuracy and robustness to the complete RoBal framework. Moreover, a significant advantage is observed regarding training time and memory consumption.

B.2. Experiments on CIFAR-100-LT

Tables 12 and 13 reveal that on CIFAR-100-LT, AT-BSL with data augmentation achieves the highest clean accuracy and adversarial robustness on both ResNet-18 and WideResNet-34-10. Compared to the improvement observed on CIFAR-10-LT, the improvements on CIFAR-100-LT are less pronounced, likely due to the more significant number of classes and fewer examples per class, making advancements more challenging.

In Fig. 8, we illustrate the robustness of different methods across each class. Given the extremely low robustness in most classes on CIFAR-100-LT and the presence of only 50 images per class in the test set, we report the average values for every 10 classes. Notably, AuA universally improves the robustness across all class groups.

[10] https://github.com/fra31/auto-attack

Table 8. The specific hyperparameters for each experiment following the integration of components from RoBal [45] into AT [30]. Cos: Cosine Classifier; BSL: Balanced Softmax Loss [35]; CM: Class-aware Margin [45]; TRADES: TRADES Regularization [52].

Method	Components				CIFAR-10-LT				CIFAR-100-LT			
	Cos	BSL	CM	TRADES	m_0	s	τ_b	τ_m	m_0	s	τ_b	τ_m
AT [30]					0	1	0	0	0	1	0	0
AT-BSL		✓			0	1	1	0	0	1	1	0
AT-BSL-Cos	✓	✓			0.1	10	1	0	0.3	10	1	0
AT-BSL-Cos-TRADES	✓	✓		✓	0.1	10	1.5	0	0.3	10	1.5	0
RoBal [45]	✓	✓	✓	✓	0.1	10	1.5	0.3	0.3	10	1.5	0.3

Table 9. The clean accuracy, robustness, time (average per epoch), and memory (GPU) using WideResNet-34-10 [50] on CIFAR-10-LT following the integration of components from RoBal [45] into AT [30]. The best results are **bolded**. The second best results are underlined. Cos: Cosine Classifier; BSL: Balanced Softmax Loss [35]; CM: Class-aware Margin [45]; TRADES: TRADES Regularization [52].

Method	Components				Accuracy						Efficiency	
	Cos	BSL	CM	TRADES	Clean	FGSM	PGD	CW	LSA	AA	Time (s)	Memory (MiB)
AT [30]					60.86	33.22	28.79	29.24	31.27	27.66	**160.01**	**2574**
AT-BSL		✓			73.84	39.13	32.02	32.29	**34.98**	30.21	162.01	**2574**
AT-BSL-Cos	✓	✓			**74.69**	40.86	34.77	31.14	30.50	29.22	163.71	**2574**
AT-BSL-Cos-TRADES	✓	✓		✓	73.34	41.28	**36.49**	31.79	31.55	30.05	302.62	6932
RoBal [45]	✓	✓	✓	✓	72.82	**41.34**	36.42	**32.48**	31.95	**30.49**	309.09	6932

B.3. Experiments on Tiny-ImageNet-LT

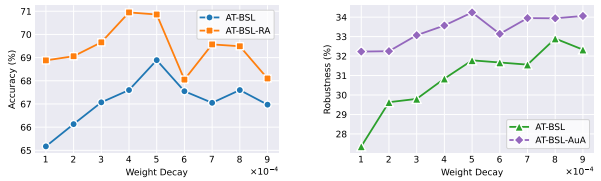
To analyze if BSL and data augmentation are as important for high-resolution datasets as they are for low-resolution datasets (such as CIFAR-10-LT and CIFAR-100-LT), we conduct experiments on Tiny-ImageNet [23]. Firstly, Tiny-ImageNet is a dataset consisting of 200 classes, with images sized 64*64 pixels, making it four times the resolution of CIFAR-10/100. We derive Tiny-ImageNet-LT using an IR of 0.1 from Tiny-ImageNet. Following [19, 24], we employ the PreActResNet-18 model [16]. Apart from the model, the experimental setup for Tiny-ImageNet-LT remains largely similar to that of CIFAR-10-LT. As observed from Table 14, both BSL and data augmentation are crucial for Tiny-ImageNet-LT.

B.4. Different PGD Steps

To investigate the impact of PGD steps on robustness, we assess the robustness achieved using different PGD steps following [45]. Table 15 indicates that RA consistently improves the robustness of AT-BSL regardless of PGD steps, and the clean accuracy also experiences improvement. Moreover, there is a trade-off between clean accuracy and robustness: as the PGD step increases, clean accuracy decreases while robustness improves. The optimal trade-off is attained at PGD-10. Hence, we employ PGD-10 in our experiments involving AT-BSL.

B.5. Different Weight Decays

During our replication of the experiments of REAT [25], we observe a discrepancy in the weight decay parameters used:



(a) (b)

Figure 9. The clean accuracy and robustness under AA for various algorithms with different weight decay using ResNet-18 on CIFAR-10-LT at the best checkpoint.

REAT employed a weight decay of 5×10^{-4}, contrasting with 2×10^{-4} used by RoBal [45]. This leads us to conduct experiments using varying values of weight decay. The results, depicted in Fig. 9, indicate that a weight decay of 5×10^{-4} offers a significant improvement over 2×10^{-4} in terms of both accuracy and robustness. However, further increasing the weight decay beyond 5×10^{-4} results in a noticeable decline in accuracy. Therefore, we employ a weight decay of 5×10^{-4} in our experiments.

B.6. Different Batch Sizes

While replicating the experiments of REAT [25], we note an inconsistency in the batch size settings: REAT utilized a batch size of 128, whereas RoBal utilized 64. To address this, we conduct experiments with different batch sizes, and the results are presented in Table 16. The findings indicate that the performance with batch sizes 64 and 128 are comparable, and both outperform larger batch sizes; however, 128 is more commonly used and helps speed up training.

Table 10. The clean accuracy, robustness, time (average per epoch) and memory (GPU) using ResNet-18 [15] on CIFAR-100-LT following the integration of components from RoBal [45] into AT [30]. The best results are **bolded**. The second best results are underlined. Cos: Cosine Classifier; BSL: Balanced Softmax Loss [35]; CM: Class-aware Margin [45]; TRADES: TRADES Regularization [52].

Method	Components				Accuracy						Efficiency	
	Cos	BSL	CM	TRADES	Clean	FGSM	PGD	CW	LSA	AA	Time (s)	Memory (MiB)
AT [30]					44.32	18.81	15.11	15.36	17.85	13.91	43.25	**946**
AT-BSL		✓			45.78	21.58	**18.96**	17.78	18.48	16.35	**41.99**	**946**
AT-BSL-Cos	✓	✓			41.83	17.95	14.69	14.22	14.87	13.14	43.86	**946**
AT-BSL-Cos-TRADES	✓	✓		✓	37.50	16.92	14.05	13.98	14.52	12.87	72.34	1724
RoBal [45]	✓	✓	✓	✓	**45.93**	**21.35**	17.40	**17.80**	**19.14**	**16.42**	72.93	1724

Table 11. The clean accuracy, robustness, time (average per epoch), and memory (GPU) using WideResNet-34-10 [50] on CIFAR-100-LT following the integration of components from RoBal [45] into AT [30]. The best results are **bolded**. The second best results are underlined. Cos: Cosine Classifier; BSL: Balanced Softmax Loss [35]; CM: Class-aware Margin [45]; TRADES: TRADES Regularization [52].

Method	Components				Accuracy						Efficiency	
	Cos	BSL	CM	TRADES	Clean	FGSM	PGD	CW	LSA	AA	Time (s)	Memory (MiB)
AT [30]					48.87	21.14	17.20	17.61	21.23	16.27	**319.33**	**2574**
AT-BSL		✓			49.68	**23.08**	**19.81**	**19.47**	21.19	17.84	323.66	**2574**
AT-BSL-Cos	✓	✓			48.29	20.25	16.34	16.43	17.90	15.09	327.17	**2574**
AT-BSL-Cos-TRADES	✓	✓		✓	44.37	18.94	15.48	15.70	17.02	14.43	603.99	6936
RoBal [45]	✓	✓	✓	✓	**50.08**	23.04	18.84	19.30	**21.87**	**17.90**	617.73	6936

Consequently, we employ a batch size of 128 in our experiments.

B.7. Different Training Epochs

As indicated in Table 17, the results between 80 and 100 training epochs show little difference without data augmentation. However, we observe that a higher number of training epochs leads to increased robustness with data augmentation. This improvement is likely attributable to the augmented diversity of examples, necessitating a more extended learning period for the model.

B.8. Hyperparameter Tuning of RoBal

Through hyperparameter tuning similar to those done for AT-BSL using ResNet-18 on CIFAR-10-LT, we find that RoBal achieves the best results with PGD-10, weight decay of 2×10^{-4}, batch size of 64, epochs of 60, and $\tau_b = 1.5$. The robustness under AA reaches 31.61%, which is close to the performance of AT-BSL.

B.9. Retraining RoBal and REAT

Compared to RoBal [45], our primary experiments employ different experimental settings, including previously discussed variables like PGD steps, weight decay, batch size, and training epochs. To facilitate a fairer comparison, we adapt these settings in our main experiments: changing PGD-5 to PGD-10, weight decay from 2×10^{-4} to 5×10^{-4}, batch size from 64 to 128, and increasing training epochs from 80 to 100, and then we retrain RoBal under these settings, referred to as RoBal (retraining). Compared with REAT [25], the only discrepancy is in the training epochs. Therefore, we adjusted REAT's training epochs to 100 and conducted a retraining called REAT (retraining). The results are presented in Table 18. The retrained RoBal is observed to achieve improved robustness, albeit at a slight cost to accuracy. Conversely, the retrained REAT displays even lower robustness than its initial version. Through this comparison, we note that the robustness achieved by the retrained RoBal and REAT is similar to that of the vanilla AT-BSL.

B.10. Other Data Augmentation Methods

Data Augmentations Designed for Long-Tailed Distributions. CUDA [2][11] initially discovered that an appropriate level of augmentation needs to be allocated for each class to mitigate class imbalance issues. Inspired by this finding, [2] introduces a curriculum to identify the appropriate data augmentation strength for each class, called CUDA: CUrriculum of Data Augmentation for long-tailed recognition. To assess CUDA's performance in adversarial training under long-tailed distributions, we augment AT-BSL with CUDA, referred to as AT-BSL-CUDA, and compared it with the vanilla AT-BSL, as shown in the Table 19. The results suggest that CUDA's performance in adversarial

[11] https://github.com/sumyeongahn/cuda_ltr

Table 12. The clean accuracy and robustness for various algorithms using ResNet-18 on CIFAR-100-LT. The best results are **bolded**.

Method	Best Checkpoint						Last Checkpoint					
	Clean	FGSM	PGD	CW	LSA	AA	Clean	FGSM	PGD	CW	LSA	AA
AT [30]	41.20	17.42	14.59	14.51	16.49	13.62	41.44	17.21	13.89	14.17	16.40	13.10
TRADES [52]	38.12	19.60	17.89	15.96	15.91	15.59	38.71	19.43	17.27	15.83	15.87	15.28
MART [41]	38.46	23.04	21.36	18.59	18.36	17.51	39.58	22.38	20.51	18.40	18.42	17.27
AWP [44]	41.53	23.47	21.79	19.68	19.73	18.61	43.57	22.91	20.72	19.11	19.30	17.82
GAIRAT [53]	38.99	19.73	18.05	16.59	16.80	15.61	39.70	14.66	11.87	11.57	12.28	10.48
LAS-AT [19]	44.33	22.02	19.59	17.18	17.11	16.15	44.70	22.11	19.23	16.93	17.03	15.77
RoBal [45]	45.93	21.35	17.40	17.80	19.14	16.42	45.78	19.97	15.37	15.75	18.67	14.51
REAT [25]	46.28	21.55	18.85	18.07	18.95	16.54	45.99	19.62	16.29	16.04	18.22	14.79
AT-BSL	45.59	21.14	18.05	17.34	18.14	15.97	45.35	18.96	15.52	15.59	17.78	14.41
AT-BSL-AuA	**48.39**	**25.81**	**22.96**	**20.73**	**21.30**	**18.90**	**50.66**	**25.89**	**22.43**	**20.62**	**21.43**	**18.79**

Table 13. The clean accuracy and robustness for various algorithms using WideResNet-34-10 on CIFAR-100-LT. The best results are **bolded**.

Method	Best Checkpoint						Last Checkpoint					
	Clean	FGSM	PGD	CW	LSA	AA	Clean	FGSM	PGD	CW	LSA	AA
AT [30]	45.18	19.25	16.36	16.43	19.00	15.60	44.86	19.01	15.65	15.89	19.12	15.08
TRADES [52]	41.71	21.91	19.85	18.46	18.39	17.91	43.22	20.28	17.46	17.34	17.56	16.69
MART [41]	41.32	25.01	23.27	20.89	20.77	19.98	43.67	22.84	19.88	18.80	19.45	17.77
AWP [44]	45.66	25.89	23.88	21.87	22.10	20.56	48.18	24.75	21.81	20.30	21.19	18.67
GAIRAT [53]	36.41	18.87	17.31	16.07	16.13	14.77	45.11	19.49	16.31	15.85	16.71	14.75
LAS-AT[19]	45.86	23.30	20.02	18.67	18.79	17.35	46.54	22.84	19.65	18.18	18.38	17.01
RoBal [45]	50.08	23.04	18.84	19.30	21.87	17.90	46.34	19.99	15.17	15.87	20.06	14.77
REAT [25]	50.29	23.99	20.82	20.25	21.93	18.65	49.22	20.89	16.57	17.08	20.89	15.49
AT-BSL	50.04	23.37	19.66	19.60	21.66	18.04	48.56	20.88	16.83	17.09	20.13	15.76
AT-BSL-AuA	**53.08**	**28.55**	**25.40**	**23.39**	**24.48**	**21.43**	**55.55**	**26.74**	**22.18**	**21.88**	**24.28**	**19.68**

Table 14. The robustness for various algorithms with different training epochs using PreActResNet-18 on Tiny-ImageNet-LT at the best checkpoint. Better results are **bolded**.

Method	Clean	FGSM	PGD	CW	LSA	AA
AT	36.30	16.58	14.52	12.65	13.16	11.37
RoBal	36.27	13.66	10.98	10.18	9.84	8.98
AT-BSL	38.83	17.47	15.34	13.35	14.08	11.83
AT-BSL-RA	**39.00**	**18.82**	**16.94**	**14.26**	**14.60**	**12.73**

training under long-tailed distributions appears less effective than RA.

Data Augmentations Designed for Adversarial Training. DAJAT [1][12] proposes a data augmentation technique designed explicitly for adversarial training. [1] initially conceptualizes data augmentation as a domain generalization problem during training. Subsequently, they introduce Diverse Augmentation-based Joint Adversarial Training (DAJAT). Since DAJAT's experiments are based on TRADES [52], it cannot be directly applied to augment AT-BSL. We conduct comparative analyses between vanilla TRADES and DAJAT. The comparison in Table 19 reveals that DAJAT still contributes to improved robustness in long-tailed adversarial training, showing comparable effectiveness to TRADES-RA.

IDBH [27][13] is another data augmentation technique that is specifically formulated for adversarial training. [27] discovers that the diversity and hardness of data augmentation play a crucial role in combating adversarial overfitting. Then, [27] proposes a new augmentation scheme called Improved Diversity and Balanced Hardness (IDBH). We utilize IDBH to augment AT-BSL, referred to as AT-BSL-IDBH. Upon comparison in Table 19, it is found that

[12] https://github.com/val-iisc/dajat

[13] https://github.com/treelli/da-alone-improves-at

Table 15. The clean accuracy and robustness for various algorithms using ResNet-18 on CIFAR-10-LT training with different PGD steps. Better results are **bolded**.

Steps	Method	Clean	FGSM	PGD	CW	LSA	AA
1	AT-BSL	77.15	23.15	12.05	13.06	24.80	11.27
	AT-BSL-RA	**82.16**	**28.28**	**14.25**	**15.34**	**26.30**	**13.21**
3	AT-BSL	72.37	36.61	28.95	28.79	30.23	26.64
	AT-BSL-RA	**74.20**	**40.39**	**32.63**	**32.25**	**33.31**	**29.79**
5	AT-BSL	68.62	39.11	33.67	32.47	32.62	30.49
	AT-BSL-RA	**69.39**	**41.86**	**36.81**	**34.33**	**33.89**	**32.62**
7	AT-BSL	68.28	39.55	34.62	32.94	32.68	31.16
	AT-BSL-RA	**68.79**	**42.45**	**37.78**	**35.31**	**34.98**	**33.57**
10	AT-BSL	68.89	40.08	35.27	33.47	33.46	31.78
	AT-BSL-RA	**70.86**	**43.06**	**37.94**	**36.24**	**36.04**	**34.24**
11	AT-BSL	67.89	39.78	35.21	33.15	33.20	31.57
	AT-BSL-RA	**68.46**	**42.10**	**37.63**	**34.58**	**34.26**	**33.12**
13	AT-BSL	**69.07**	39.82	35.19	33.12	32.91	31.44
	AT-BSL-RA	68.90	**42.18**	**37.89**	**34.93**	**34.58**	**33.35**

Table 16. The robustness for various algorithms with different batch sizes using ResNet-18 on CIFAR-10-LT at the best checkpoint. Better results are **bolded**. BS: Batch Size.

BS	Method	Clean	FGSM	PGD	CW	LSA	AA
64	AT-BSL	**67.82**	41.42	36.57	34.41	34.22	32.67
	AT-BSL-RA	66.70	**41.85**	**37.87**	**35.34**	**34.83**	**33.78**
128	AT-BSL	68.89	40.08	35.27	33.47	33.46	31.78
	AT-BSL-RA	**70.86**	**43.06**	**37.94**	**36.24**	**36.04**	**34.24**
256	AT-BSL	66.72	37.66	33.08	31.55	31.42	29.98
	AT-BSL-RA	**67.93**	**41.05**	**36.60**	**33.78**	**33.43**	**31.97**
512	AT-BSL	60.01	35.45	32.27	29.44	29.02	28.15
	AT-BSL-RA	**63.25**	**37.66**	**34.38**	**31.14**	**30.59**	**29.53**

Table 17. The robustness for various algorithms with different training epochs using ResNet-18 on CIFAR-10-LT at the best checkpoint. Better results are **bolded**.

Method	Clean	FGSM	PGD	CW	LSA	AA
AT-BSL-80	66.68	**40.18**	**36.11**	**33.87**	**33.64**	**31.95**
AT-BSL	**68.89**	40.08	35.27	33.47	33.46	31.78
AT-BSL-RA-80	69.39	41.93	37.20	34.82	34.36	32.92
AT-BSL-RA	**70.86**	**43.06**	**37.94**	**36.24**	**36.04**	**34.24**

IDBH's effectiveness is less pronounced than RA on long-tailed datasets.

Table 18. The robustness for various algorithms using ResNet-18 on CIFAR-10-LT at the best checkpoint. The best results are **bolded**.

Method	Clean	FGSM	PGD	CW	LSA	AA
RoBal	70.34	40.50	35.93	31.05	31.10	29.54
RoBal (retraining)	67.46	41.61	**38.04**	32.75	33.08	31.26
REAT	67.38	40.13	35.83	33.88	33.66	32.20
REAT (retraining)	67.38	39.51	35.15	33.53	33.31	31.77
AT-BSL	68.89	40.08	35.27	33.47	33.46	31.78
AT-BSL-RA	**70.86**	**43.06**	37.94	**36.24**	**36.04**	**34.24**

Table 19. The robustness for various algorithms with different data augmentations using ResNet-18 on CIFAR-10-LT at the best checkpoint. The best results are **bolded**.

Method	Clean	FGSM	PGD	CW	LSA	AA
TRADES	43.61	29.18	**27.81**	**26.73**	**26.58**	**26.41**
DAJAT	42.04	**29.34**	27.70	26.47	26.36	26.27
TRADES-RA	**44.45**	29.18	27.61	26.51	26.47	26.27
AT-BSL	68.89	40.08	35.27	33.47	33.46	31.78
AT-BSL-CUDA	68.05	40.06	36.48	33.07	32.75	31.49
AT-BSL-IDBH	70.80	39.54	33.30	32.56	33.01	31.24
AT-BSL-RA	**70.86**	**43.06**	**37.94**	**36.24**	**36.04**	**34.24**

Table 20. The robustness for various algorithms with different training epochs using ResNet-18 on CIFAR-10-LT at the best checkpoint. Better results are **bolded**.

Method	Clean	FGSM	PGD	CW	LSA	AA
AT-BSL	68.89	40.08	35.27	33.47	33.46	31.78
AT-BSL-RA	70.86	43.06	37.94	36.24	36.04	34.24
AT-BSL-DM	**72.61**	**47.09**	**42.01**	**41.56**	**41.89**	**39.48**

B.11. Using Data Generated by Diffusion Models

To investigate the potential of leveraging data generated by diffusion models to improve the robustness of AT-BSL, we train a diffusion model, DDPM, for CIFAR-10-LT, selecting the version with the best Fréchet Inception Distance (FID) of 6.92 after 18 sampling steps following [11, 21]. For the generation of 1 million data points, we produce 100,000 images per class, culminating in a total of 1 million images. Following [11], we set the proportion of unsupervised data to 0.7 and train a ResNet-18 using AT-BSL, which we refer to as AT-BSL-DM. The results presented in Table 20 clearly demonstrate the significant improvement in robustness by incorporating data generated by diffusion models.

B.12. Different Adversarial Training Methods

To further validate the hypothesis that data augmentation alone improves robustness under long-tailed distributions,

we conduct experiments across various adversarial training methods, employing AuA or RA. As evidenced in Table 21, with few exceptions, data augmentation is beneficial for robustness. This effect is common on CIFAR-100-LT, likely due to the reduced number of training examples per class in this dataset, leading to a more substantial reliance on data augmentation techniques.

B.13. Standard Deviation

We repeat AT-BSL and AT-BSL-RA five times using ResNet-18 on CIFAR-10-LT. Their mean and standard deviation of robustness under AA are 31.65 ± 0.45 and 34.12 ± 0.51, respectively. The relatively small variance indicates the stability of our training process.

B.14. Computational Cost Comparison

In this section, we compare the computational costs of AT-BSL and AT-BSL-RA/AuA regarding average training time per epoch and GPU memory usage. The detailed results are summarized in Table 22. The comparison indicates that the introduction of data augmentation incurs a negligible increase in time cost without imposing additional memory overhead.

C. Comparison with Concurrent Works

Concurrently and independently from our work, REAT [25] has also explored adversarial training under long-tailed distributions. [25] identifies that compared to conventional adversarial training on balanced datasets, adversarial training under long-tailed distributions tends to produce imbalanced adversarial examples and feature embedding spaces. To address this issue, [25] introduces a novel adversarial training framework: Re-balancing Adversarial Training (REAT). Notably, the experimental settings utilized in our paper are fundamentally consistent with those employed in REAT. Moreover, as shown in Tables 3, 4, 12, and 13, the robustness achieved by our implemented vanilla AT-BSL is comparable to that of REAT.

Table 21. The robustness for various algorithms with/without data augmentations at the best checkpoint. RA is employed on the combination of ResNet-18 and CIFAR-10-LT, whereas AuA is utilized for other model and dataset combinations. Better results are **bolded**.

Method	CIFAR-10-LT						CIFAR-100-LT					
	ResNet-18			WideResNet-34-10			ResNet-18			WideResNet-34-10		
	Clean	PGD	AA	Clean	PGD	AA	Clean	PGD	AA	Clean	PGD	AA
AT [30]	**49.35**	27.30	25.76	59.21	27.88	27.07	41.20	14.59	13.62	45.18	16.36	15.60
AT-RA/AuA	44.31	**27.81**	**25.90**	**62.98**	**33.40**	**31.64**	**45.17**	**19.78**	**17.22**	**50.00**	**21.87**	**19.44**
TRADES [52]	43.61	**27.81**	**26.41**	51.28	28.70	27.72	38.12	17.89	15.59	41.71	19.85	17.91
TRADES-RA/AuA	**44.45**	27.61	26.27	**55.89**	**31.53**	**29.77**	**42.14**	**19.69**	**16.12**	**46.23**	**22.78**	**19.52**
MART [41]	**48.61**	**30.29**	**27.73**	**49.13**	**32.32**	**29.60**	**38.46**	21.36	17.51	41.32	23.27	19.98
MART-RA/AuA	43.76	29.86	26.77	48.07	31.93	28.31	38.01	**22.64**	**18.68**	**43.43**	**25.41**	**21.26**
AWP [44]	**49.29**	**31.20**	**29.53**	**50.91**	**31.85**	**30.06**	**41.53**	21.79	18.61	**45.66**	23.88	20.56
AWP-RA/AuA	45.28	30.56	28.73	44.06	29.91	27.81	41.07	**23.02**	**19.37**	45.27	**25.76**	**21.60**
GAIRAT [53]	**50.83**	**27.46**	**20.41**	59.89	30.40	25.38	38.99	18.05	**15.61**	36.41	17.31	14.77
GAIRAT-RA/AuA	43.56	27.34	17.82	**66.43**	**37.96**	**25.53**	**41.94**	**19.18**	14.82	**49.75**	**22.19**	**18.24**
LAS-AT [19]	**52.81**	30.32	28.53	57.52	29.86	28.84	44.33	19.59	16.15	45.86	20.02	17.35
LAS-AT-RA/AuA	51.20	**31.20**	**29.18**	**59.14**	**34.51**	**32.54**	**45.18**	**22.78**	**18.61**	**49.73**	**24.09**	**20.79**
RoBal [45]	**70.34**	35.93	29.54	**72.82**	36.42	30.49	45.93	17.40	16.42	50.08	18.84	17.90
RoBal-RA/AuA	68.66	**37.50**	**30.06**	72.57	**40.54**	**31.87**	**47.75**	**19.93**	**18.04**	**54.12**	**21.41**	**19.66**
REAT [25]	**67.38**	35.83	**32.20**	**73.16**	35.94	33.20	46.28	18.85	16.54	50.29	20.82	18.65
REAT-RA/AuA	66.64	**36.97**	31.84	72.05	**40.05**	**35.74**	**47.65**	**22.86**	**18.48**	**50.10**	**25.07**	**20.81**
AT-BSL	68.89	35.27	31.78	73.19	35.60	32.80	45.59	18.05	15.97	50.04	19.66	18.04
AT-BSL-RA/AuA	**70.86**	**37.94**	**34.24**	**75.17**	**40.84**	**37.15**	**48.39**	**22.96**	**18.90**	**53.08**	**25.40**	**21.43**

Table 22. The time and memory for various algorithms. On the combination of ResNet-18 and CIFAR-10-LT, whereas AuA is utilized for other model and dataset combinations. All experiments are run on NVIDIA RTX 3090.

Dataset	Method	ResNet-18		WideResNet-34-10	
		Time (s)	Memory (MiB)	Time (s)	Memory (MiB)
CIFAR-10-LT	AT-BSL	22.37	1345	200.70	4293
	AT-BSL-RA/AuA	22.43	1345	201.42	4293
CIFAR-100-LT	AT-BSL	30.94	1347	277.82	4293
	AT-BSL-RA/AuA	31.25	1347	279.66	4293