Towards Adversarially Robust Dataset Distillation by Curvature Regularization

Eric Xue[1], Yijiang Li[2], Haoyang Liu[3], Peiran Wang[4], Yifan Shen[3], Haohan Wang[3]
[1]UToronto [2]UC San Diego [3]UIUC [4]UCLA

Scan Here for Website:

AAAI-25

Introduction

• Background: The **computational demands** for training deep learning models are continuously growing due to the increasing volume of data, prohibitive to entities with limited computational resources.

• Limitation: **Dataset distillation** offers a means to reduce the size of the data while maintaining its utility. However, little attention has been given to the **adversarial robustness** of models trained on distilled datasets, which is important to practical applications.

• Research question: *How can we embed adversarial robustness into the dataset distillation process, thereby generating datasets that lead to more robust models?*

Preliminaries

• Problem formulation: Robust dataset distillation can be defined as a tri-level optimization problem, where the goal is to find the optimal distilled dataset $\mathcal{S}$, such that training a model $\boldsymbol{\theta}$ on $\mathcal{S}$ minimizes the *maximum adversarial loss* of $\boldsymbol{\theta}$ on the original data distribution $\mathcal{D}_{\mathcal{T}}$:

$$\mathcal{S}^* = \arg\min_{\mathcal{S}} \mathop{\mathbb{E}}_{(\mathbf{x},y)\sim\mathcal{D}_{\mathcal{T}}} \left(\max_{\|\mathbf{v}\|\leq\rho} \ell\left(\mathbf{x}+\mathbf{v}, y; \boldsymbol{\theta}(\mathcal{S})\right)\right)$$
$$\text{subject to } \boldsymbol{\theta}(\mathcal{S}) = \arg\min_{\boldsymbol{\theta}} \mathcal{L}(\mathcal{S};\boldsymbol{\theta}).$$

• A natural approach is to integrate **adversarial training** into the distillation process. However, our empirical studies suggest this approach is ineffective. Even mild adversarial training during distillation significantly degrades clean accuracy of the trained model, which in turn limits robust accuracy substantially.

IPC	Attack	GUARD	SRe^{2}L	SRe^{2}L +Adv
1	None (Clean)	**37.49**	27.97	11.61
	PGD100	**16.22**	12.05	10.03
	Square	**26.74**	18.62	11.18
	AutoAttack	**15.81**	12.12	10.03
	CW	**29.14**	20.38	10.31
	MIM	**16.32**	12.05	10.03
10	None (Clean)	**57.93**	42.42	12.81
	PGD100	**23.87**	4.76	9.93
	Square	**44.07**	22.77	11.46
	AutoAttack	**19.69**	4.99	9.96
	CW	**58.67**	22.11	10.90
	MIM	**21.80**	4.76	9.96

Table 1: Performance comparison between our method and SRe^{2}L w/wo adversarial training

• Possible reason:
◦ Cross-over mixture problem: adversarial examples can cross over the decision boundary, providing misleading training signals
◦ Create irrelevant artifacts as a kind of data augmentation (DA), consistent with the observation in SRe^{2}L that DA can hurt performance

Theory

In the paper, we show that the adversarial loss on a distilled image $\tilde{\ell}_\rho^{adv}(\mathbf{x}')$ is upper bounded as below:

$$\tilde{\ell}_\rho^{adv}(\mathbf{x}') \leq \mathop{\mathbb{E}}_{\mathbf{x}\sim D_c} \ell(\mathbf{x}) + \frac{1}{2}\rho^2 \mathop{\mathbb{E}}_{\mathbf{x}\sim D_c} \lambda_1(\mathbf{x}) + L\sigma.$$

Through further analysis, we determine that the term $\frac{1}{2}\rho^2 \mathop{\mathbb{E}}_{\mathbf{x}\sim D_c} \lambda_1(\mathbf{x})$ is the least constrained term and requires optimization, where

• D_c denotes the distribution of data in class c, and
• λ_1 is the largest eigenvalue of the Hessian matrix $H(\ell(\mathbf{x}))$

Method

We present **GUARD** (**G**eometric Reg**U**larization for **A**dversarially **R**obust **D**ataset).

• Since the adversarial loss of distilled datasets is largely upper-bounded by the curvature of the loss function, our goal is to reduce the value of λ_1 by incorporating it into the loss function.

• To bypass the expensive calculation of the Hessian matrix, we use an efficient approximate of lambda as below:

$$\lambda_1 = \|\lambda_1 \mathbf{v}_1\| \approx \|\frac{\nabla\ell(\mathbf{x}+h\mathbf{v}_1) - \nabla\ell(\mathbf{x})}{h}\|.$$

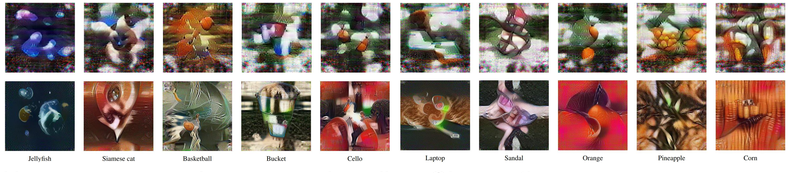
Where $\mathbf{v}_1$ is the unit eigenvector corresponding to λ_1.

• To make it more efficient, we use the normalized gradient $\mathbf{z} = \frac{\nabla\ell(\mathbf{x}))}{\|\nabla\ell(\mathbf{x}))\|}$ to replace $\mathbf{v}_1$, inspired by previous works.

The regularized loss function is finally defined as:

$$\ell_R(\mathbf{x}) = \ell(\mathbf{x}) + \lambda\|\nabla\ell(\mathbf{x}+h\mathbf{z}) - \nabla\ell(\mathbf{x})\|^2$$

Visualization



Upper row – ours; lower row – baseline. Observations:
• Our synthetic data have more distinct object outlines; exhibits high-frequency noise similar to those generated by adversarial attacks

Results

Dataset	IPC	Attack	Methods			
			GUARD	SRe^{2}L	MTT	TESLA
TinyImageNet	10	None (Clean)	**37.00**	33.18	8.14	14.06
		PGD100	6.39	1.08	4.08	**8.40**
		Square	**19.53**	15.85	2.48	6.31
		AutoAttack	4.91	0.79	2.44	**6.16**
		CW	**8.40**	3.24	2.50	6.26
		MIM	6.51	1.10	4.08	**8.40**
	50	None (Clean)	55.61	**56.42**	17.84	28.24
		PGD100	**15.63**	0.27	5.62	12.12
		Square	**36.93**	15.50	3.84	10.39
		AutoAttack	**13.84**	0.16	3.52	10.01
		CW	**20.46**	12.12	3.66	10.13
		MIM	**16.09**	0.29	5.64	12.12
ImageNet-1K	10	None (Clean)	**27.25**	21.30	-	-
		PGD100	**5.25**	0.55	-	-
		Square	17.88	**18.02**	-	-
		AutoAttack	**3.33**	0.34	-	-
		CW	**7.68**	3.21	-	-
		MIM	**5.23**	0.51	-	-
	50	None (Clean)	39.89	**46.80**	-	-
		PGD100	**9.77**	0.59	-	-
		Square	28.39	**32.40**	-	-
		AutoAttack	**7.03**	0.47	-	-
		CW	**14.14**	6.31	-	-
		MIM	**9.84**	0.64	-	-

(a)

IPC	Attack	Methods					
		DC	DC†	SRe^{2}L	SRe^{2}L†	CDA	CDA†
1	None (Clean)	29.96	**30.95**	17.13	**22.88**	14.98	**23.18**
	PGD100	24.59	**46.88**	13.56	**19.21**	12.69	**18.70**
	Square	24.72	**48.56**	13.75	**19.55**	12.84	**19.17**
	AutoAttack	**24.33**	14.99	13.43	**18.91**	12.63	**18.42**
	CW	24.58	**15.19**	13.52	**18.95**	12.62	**18.52**
	MIM	**24.62**	15.27	13.57	**19.22**	12.69	**18.71**
10	None (Clean)	45.38	**46.83**	26.58	**30.76**	20.55	**30.65**
	PGD100	31.84	**32.36**	18.24	**22.31**	14.60	**24.33**
	Square	**33.71**	33.54	19.99	**24.16**	15.93	**25.66**
	AutoAttack	31.05	**31.84**	18.11	**21.58**	14.47	**24.04**
	CW	31.95	**32.35**	18.73	**21.98**	14.83	**24.51**
	MIM	31.89	**32.37**	18.25	**22.35**	14.62	**24.34**
50	None (Clean)	-	-	43.96	**44.05**	36.32	**43.05**
	PGD100	-	-	24.74	**33.12**	21.58	**33.02**
	Square	-	-	29.68	**35.22**	25.76	**35.19**
	AutoAttack	-	-	24.45	**32.24**	21.46	**31.96**
	CW	-	-	26.09	**32.67**	22.54	**32.56**
	MIM	-	-	24.81	**33.12**	21.61	**33.03**

(b)

Table 2: Evaluation results of our GUARD method. (a) Clean accuracy and robustness under various adversarial attacks compared with previous methods; (b) comparison of different DD methods and their GUARD regularized version (marked by †). For more results please see our paper.

• Subfigure (a)(b): GUARD significantly improves robust accuracy across most settings, while maintaining or even improving clean accuracy
• Subfigure (b): GUARD remains effective when applied to other DD methods

Discussion

• **Robustness Guarantee**: Our theory guarantees the robustness on distilled dataset effectively transfer to the real dataset

• **Computational Overhead**: Only requires an extra forward pass to compute the loss $\ell(\mathbf{x}+h\mathbf{z})$ within each iteration
◦ More than 10x faster than incorporating adversarial training into DD

• **Transferability**: GUARD can theoretically be applied to a large number of dataset distillation methods, beyond the 3 methods in our experiments