CARTL: Cooperative Adversarially-Robust Transfer Learning

Dian Chen[1] Hongxin Hu[2] Qian Wang[1] Yinli Li[1]
Cong Wang[3] Chao Shen[4] Qi Li[5]

[1]Wuhan University [2]University at Buffalo [3]City University of Hong Kong
[4]Xi’an Jiaotong University [5]Tsinghua University

Overview

- We reveal that there is a trade-off between accuracy and robustness in transfer learning.
- We propose a new transfer learning strategy, CARTL, for improving the accuracy-robustness trade-off of the target model.
- We demonstrate that selectively freezing the Batch Norm layers can further boost the robustness transfer.

Training Deep Neural Networks Is Tough

Network capacity
- Billions of parameters

Training data
- Extra training images except for ImageNet

Computational cost
- Thousands of core-hour

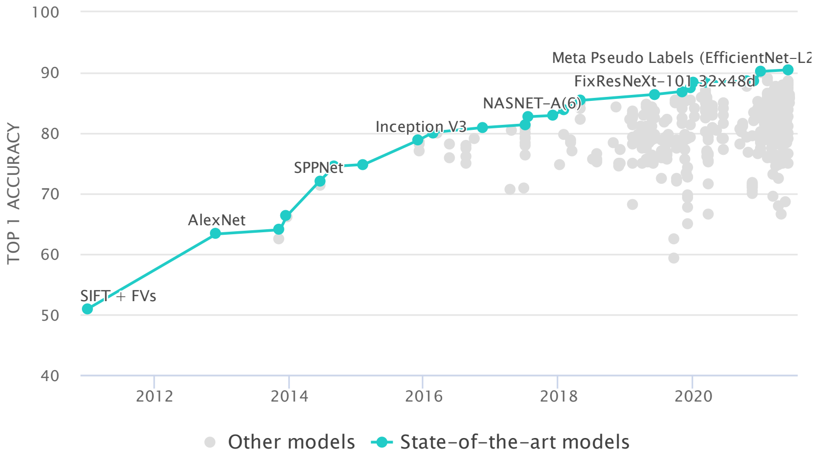


Figure from paperswithcode.com (2021.5)

- Zhai et al., Scaling Vision Transformers, arXiv, 2021.

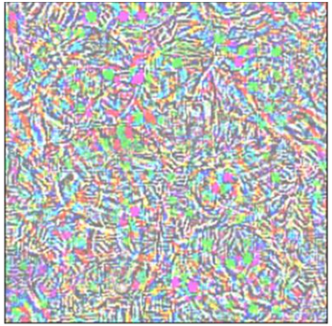
Adversarial Examples

Adversarial examples are perturbated inputs that deceive DNNs into answering incorrect results.

$$\arg\max_i f_{\boldsymbol{\theta}}(\boldsymbol{x})_i \neq \arg\max_j f_{\boldsymbol{\theta}}(\boldsymbol{x}')_j$$

Golf ball (33.74%)

$\boldsymbol{x}$



$\boldsymbol{\delta}$

$s.t. \parallel \boldsymbol{\delta} \parallel_p \leq \epsilon, p = 0, 2, \infty$

Sleeping bag (98.12%)

$\boldsymbol{x}'$

- Szegedy et al., Intriguing Properties of Neural Networks, ICLR 2014

Adversarial Training

Adversarial training is similar to natural model training, but it takes (only) adversarial examples as the training data.

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}} \left[\max_{\boldsymbol{\delta}\in\mathbb{S}} \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}), y) \right]$$

$AT(\boldsymbol{\theta}, \mathcal{D}, \epsilon, \alpha)$:
repeat
  $(\boldsymbol{x}, y) \leftarrow \mathcal{D}$ *// mini-batch*
  $\boldsymbol{\delta} \leftarrow rand\_init$
  for i **in** $1, \ldots, N$
    $loss = \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}), y)$
    $\boldsymbol{\delta} = \Pi_{\epsilon}(\boldsymbol{\delta} + \alpha \cdot \nabla_{\boldsymbol{\delta}} loss)$
  end for
  $\boldsymbol{\theta} = \boldsymbol{\theta} + \eta \cdot \nabla_{\boldsymbol{\theta}} \mathcal{L}(f_{\boldsymbol{\theta}}(\boldsymbol{x}+\boldsymbol{\delta}), y)$
until *convergence*

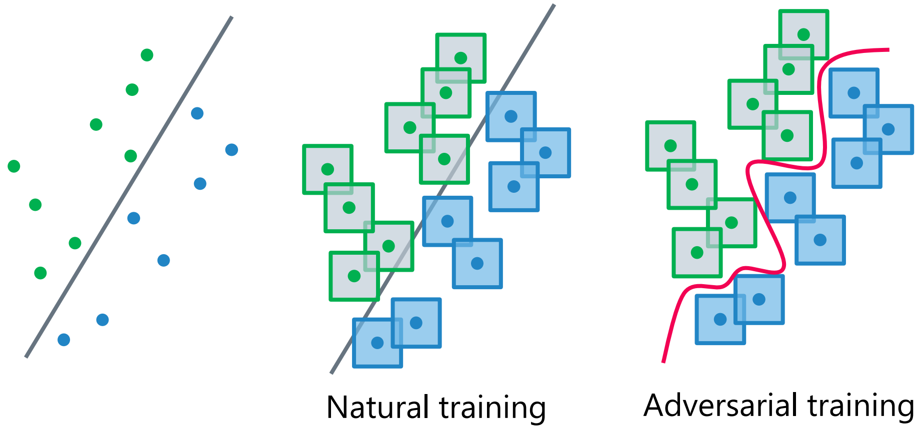


Figure from (Madry et al., 2018)

- Madry et al., Towards Deep Learning Models Resistant to Adversarial Attacks, ICLR 2018.

Adversarial Training

Network capacity

- Adversarial robustness exhibits a strong demand on the network's capability, i.e., its depth and width.

Training data

- Adversarial training requires more training data than natural training.

Computational cost

- The training cost is N-times higher than natural training.

"Adversarial training increases the burden of model training."

- Xie et al., Intriguing Properties of Adversarial Training at Scale, ICLR 2020.
- Schmidt et al., Adversarially Robust Generalization Requires More Data, NeurIPS 2018.
- Shafahi et al., Adversarial Training for Free!, NeurIPS 2019.

Transfer Learning

Utilizing the knowledge obtained from the source domain to solve target domain tasks.

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathcal{L}\left(f_{\overline{\boldsymbol{\theta}}}^{(L-k+1..L)}\left(f^{(L-k)}(\boldsymbol{x})\right), y\right)\right]$$

$TL(\boldsymbol{\theta}_*, \mathcal{D}, k)$:

$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta}_*$ *// copy pre-trained weights*

$\overline{\boldsymbol{\theta}} \coloneqq \{\boldsymbol{\theta}_{L-k+1}, \dots, \boldsymbol{\theta}_L\}$ *// fine-tune last k layers*

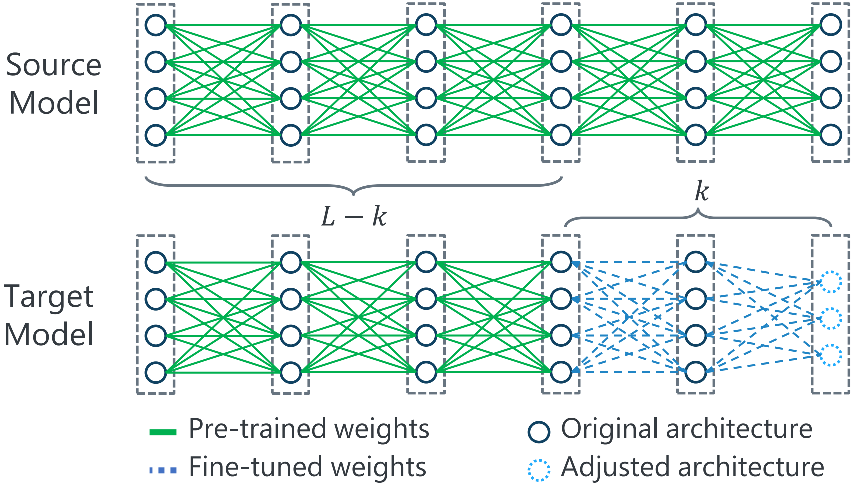
repeat

$(\boldsymbol{x}, y) \leftarrow \mathcal{D}$ *// mini-batch*

$loss = \mathcal{L}\left(f_{\overline{\boldsymbol{\theta}}}\left(f^{(L-k)}(x)\right), y\right)$

$\overline{\boldsymbol{\theta}} = \overline{\boldsymbol{\theta}} + \eta \cdot \nabla_{\overline{\boldsymbol{\theta}}} loss$

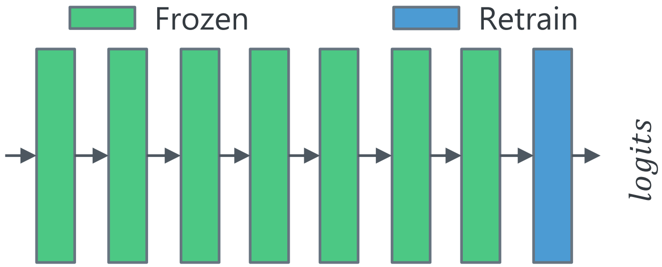
until *convergence*



- Yosinski et al., How Transferable Are Features in Deep Neural Networks?, NeurIPS 2014.

Shafahi's Work

- The robustness of a hardened model is mainly due to its robust deep feature.
- Robustness does transfer when merely retraining the **last** fully-connected layer of a robust model.
- The accuracy of the target model **is poor**.

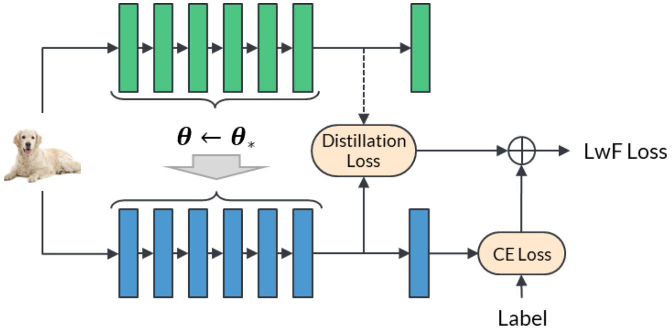


- Shafahi et al., Adversarially Robust Transfer Learning, ICLR 2020.

Shafahi's Work

- **End-to-end** transfer learning with **a distillation term**, called LwF

$$\min_{\boldsymbol{\theta}} \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}\left[\mathcal{L}(f(\boldsymbol{x};\boldsymbol{\theta}),y) + \lambda_d \cdot \| f^{(L-1)}(\boldsymbol{x};\boldsymbol{\theta}) - f^{(L-1)}(\boldsymbol{x};\boldsymbol{\theta}_*) \|_2\right]$$

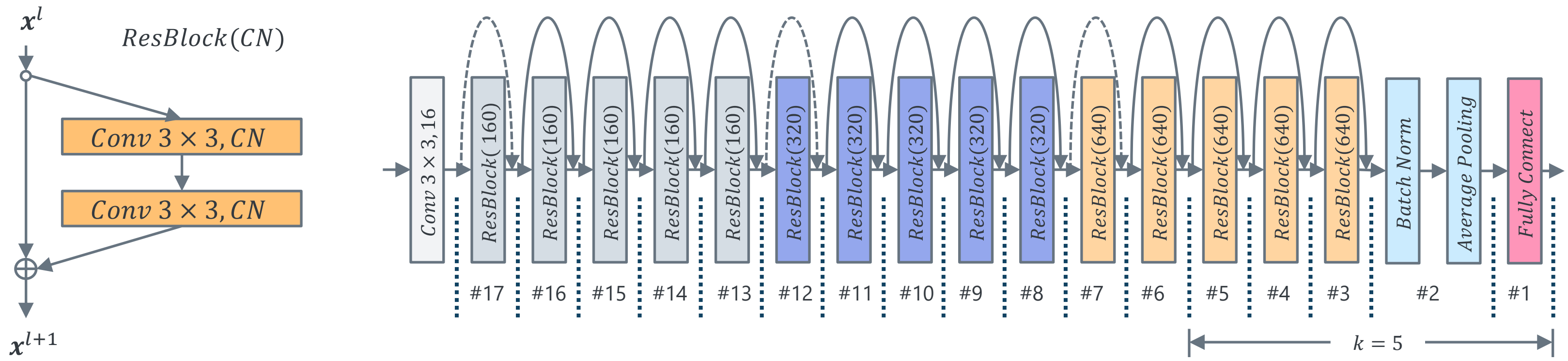


- A trade-off between the accuracy and robustness of the target model:
  - Reduce λ_d for improving generalization, i.e., accuracy on the target domain.
  - Increase λ_d for obtaining better robustness transfer.

- Shafahi et al., Adversarially Robust Transfer Learning, ICLR 2020.

Problem Exploration

- How the number of fine-tuned layers affects the target model's robustness and accuracy.
- Widely-adopted architecture: wide residual network (WRN)
- Fine-tune the pre-trained robust model in the unit of the block on the target domain.

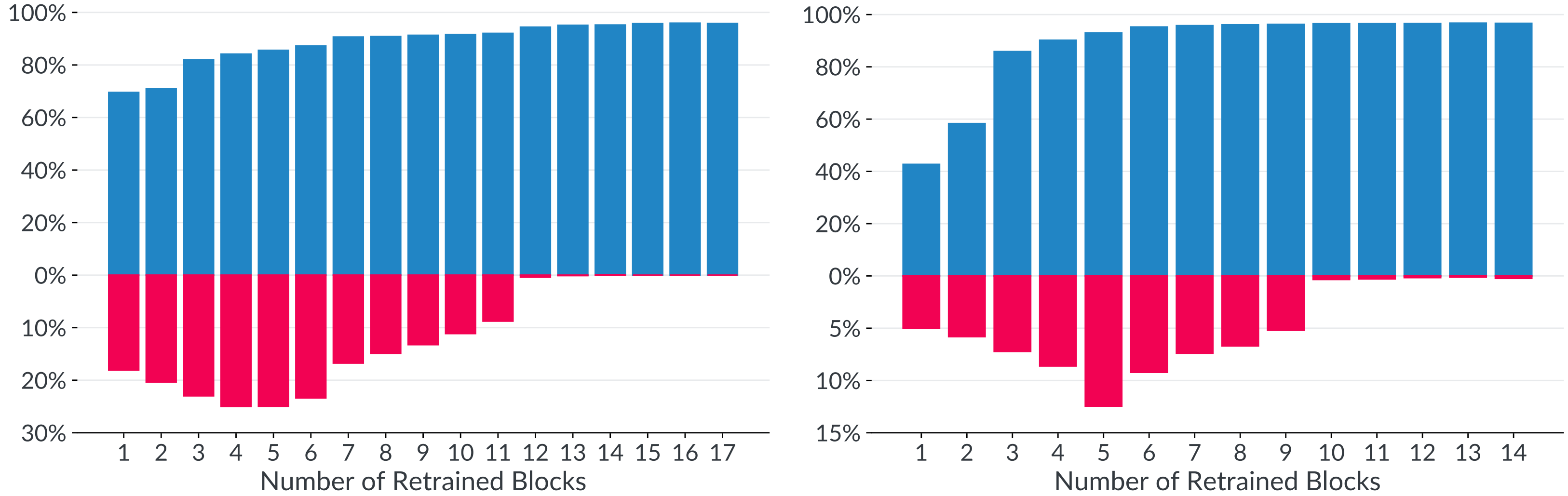


WRN 34-10

Figure from (Shafahi et al., 2020)

- Zagoruyko et al., Wide Residual Networks, arXiv, 2016.

Problem Exploration



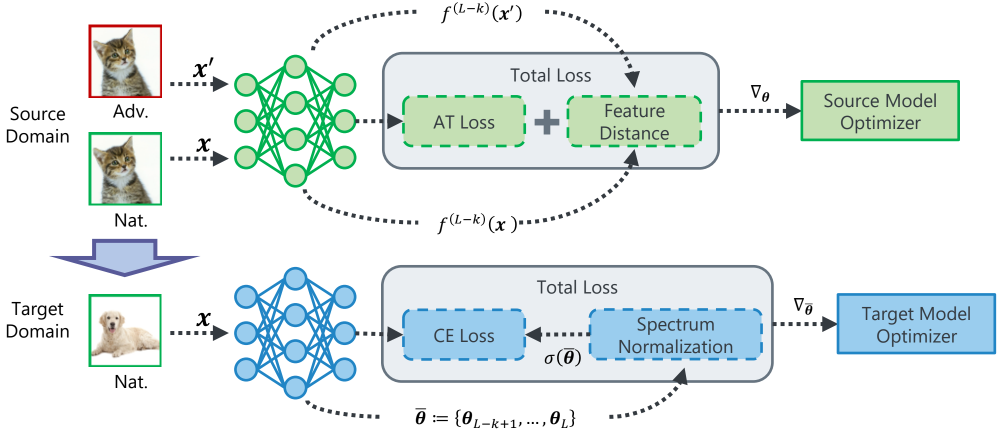
- Merely fine-tuning the last layer may not be sufficient.
- Insufficient accuracy leads to lower robustness.
- Accuracy increases together with the number of the fine-tuned layers.
- There is **a trade-off** between robustness and accuracy.

“

Can the target model obtain high accuracy while inheriting more robustness from the source model?

CARTL

- A cooperative approach
  - Feature distance minimization (FDM): adjusted adversarial training for the source model
  - Non-expansive fine-tuning (NEFT): constrained fine-tuning for model transfer
- Fine-tune the last k layers and freeze the first $L-k$ layers

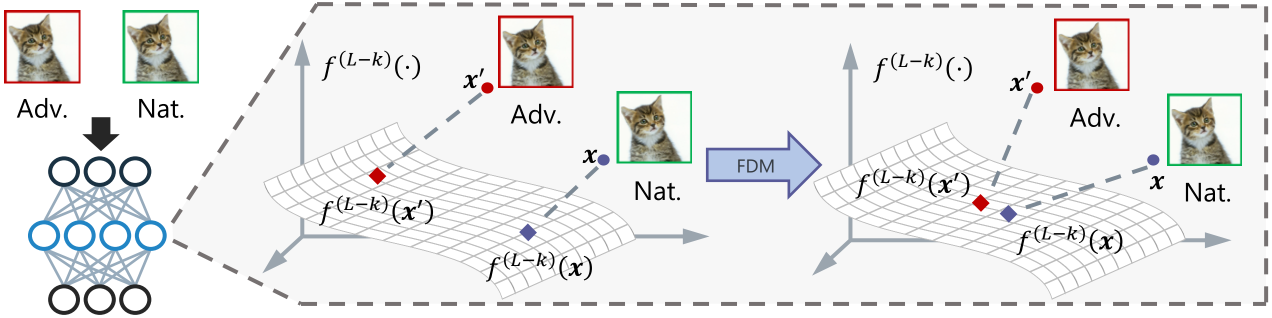


Feature Distance Minimization

The first $L - k$ layers frozen during transfer learning are taken as a feature extractor.

- Two inputs extracted similar features tend to be classified into an identical label.
- Reduce the distance between the features of adv. ($\boldsymbol{x}'$) and nat. ($\boldsymbol{x}$) examples.

$$\mathcal{L}_{AT} + \frac{\lambda}{\sqrt{d}} \cdot \| f^{(L-k)}(\boldsymbol{x}) - f^{(L-k)}(\boldsymbol{x}') \|_2$$



- Wang et al., With Great Training Comes Great Vulnerability: Practical Attacks against Transfer Learning, USENIX Security 2018

Non-expansive Fine-tuning

- We call a function f Lipchitz continuous if

$$\| f(x) - f(x') \|_2 \leq \Lambda \cdot \| x - x' \|_2$$

- Lipchitz constant for DNN $f(\cdot;\boldsymbol{\theta}) \coloneqq f^{L}_{\boldsymbol{\theta}_L} \circ f^{L-1}_{\boldsymbol{\theta}_{L-1}} \circ \cdots \circ f^{1}_{\boldsymbol{\theta}_1}(\cdot)$

$$\| f(\boldsymbol{x}) - f(\boldsymbol{x}') \|_2 \leq \Lambda_L \cdot \Lambda_{L-1} \cdots \Lambda_1 \| \boldsymbol{x} - \boldsymbol{x}' \|_2$$

- A general form of the deep neural layer f^l:

$$f^l = \boldsymbol{W}^l \cdot \boldsymbol{x} + \boldsymbol{b}^l$$

Non-expansive Fine-tuning

- The remaining dissimilarity of features may still result in model misclassification.

- Non-expansive fine-tuning: mitigate the error caused by the dissimilarity of features.

$$\boldsymbol{W}_*^l := \beta \cdot \frac{\boldsymbol{W}^l}{\sigma(\boldsymbol{W}^l)}$$

- β is a hyper-parameter for further scaling down the Lipchitz constant

- Miyato et al., Spectral Normalization for Generative Adversarial Networks, ICLR 2018.

Rethinking Fine-tuning Batch Norm Layer

- An essential component for DNN: internal covariate shift, model training acceleration

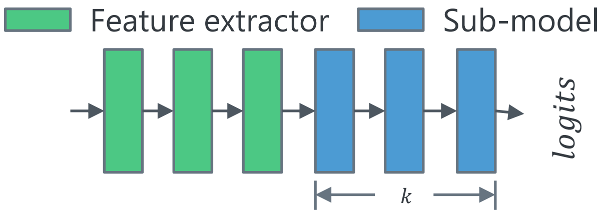
$$BN(\boldsymbol{x}) \coloneqq \boldsymbol{W} \cdot \frac{\boldsymbol{x} - mean(\boldsymbol{x})}{\sqrt{var(\boldsymbol{x}) + \varepsilon}} + \boldsymbol{b}$$

$$\boldsymbol{\mu} \coloneqq m \cdot \boldsymbol{\mu} + (1 - m) \cdot mean(\boldsymbol{x})$$

$$\boldsymbol{\sigma} \coloneqq m \cdot \boldsymbol{\sigma} + (1 - m) \cdot \sqrt{var(\boldsymbol{x})}$$

- Parameters:
  - Statistic parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$: updated with a momentum (m).
  - Affine parameters $\boldsymbol{W}$ and $\boldsymbol{b}$: updated through back propagation.

- Four cases:
  - Update/reuse $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ in the feature extractor
  - Fine-tune/freeze $\boldsymbol{W}$ and $\boldsymbol{b}$ in the sub-model



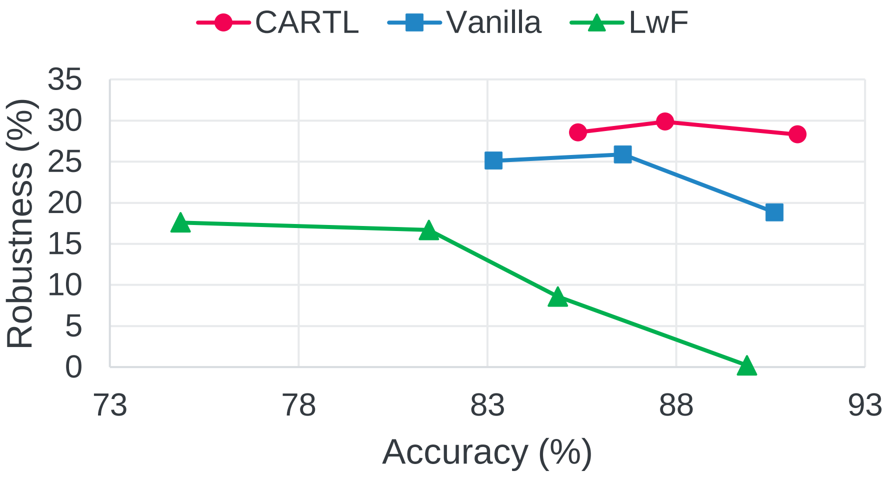
Rethinking Fine-tuning Batch Norm Layer

		W, b		μ, σ, W, b	
		Acc (%)	**Rob (%)**	**Acc (%)**	**Rob (%)**
CIFAR-100 → CIFAR-10 (8)	-	91.17	14.36	90.86	14.89
	W, b	90.70	17.41	90.84	18.54
CIFAR-10 → GTSRB (6)	-	93.02	30.22	89.29	32.22
	W, b	92.13	32.22	88.94	34.53
CIFAR-10 → SVHN (6)	-	95.29	3.88	95.24	9.22
	W, b	95.16	4.90	94.86	11.52
CIFAR-10 → SVHN (5)	-	93.47	4.71	92.92	12.45
	W, b	93.41	5.64	92.10	14.16

- Transferred robustness can be boosted if freezing affine parameters of the sub-model.
- Freezing statistics of the feature extractor plays a crucial role in robustness transfer.
- Reuse source domain statistics may cast negative impacts on the accuracy

Evaluations

- LwF improves the robustness but aggressively harms the accuracy and vice versa.
- Vanilla and CARTL maintain higher robustness in the case of an equivalent level of accuracy.
- CARTL further improves the accuracy-robustness trade-off compared with Vanilla.



Evaluations

- CARTL exhibits similar trends to Vanilla, achieving higher robustness at Case-6.
- A smaller λ helps robustness transfer but slightly results in lower accuracy.
- Reducing Lipschitz constants significantly improves the target models' robustness

		NEFT $\beta = 1.0$		NEFT $\beta = 0.6$		NEFT $\beta = 0.4$	
		Acc (%)	**Rob (%)**	**Acc (%)**	**Rob (%)**	**Acc (%)**	**Rob (%)**
Case-4	$\lambda = 0.01$	86.09	25.73	86.08	27.17	85.64	28.40
	$\lambda = 0.005$	85.41	25.75	85.47	27.14	85.51	28.47
Case-6	$\lambda = 0.01$	87.78	25.58	87.92	27.27	87.96	29.60
	$\lambda = 0.005$	87.66	25.97	88.07	27.64	87.79	30.94
Case-8	$\lambda = 0.01$	91.85	16.36	91.63	19.22	91.55	27.47
	$\lambda = 0.005$	91.71	17.62	91.10	21.60	91.30	29.34

CIFAR-100 → CIFAR-10

Evaluations

- Fine-tuning the target model with NEFT significantly increases its robustness.
- FDM further improves the robustness except for Case-8.
- By using FDM, the target model's accuracy slightly rises in all cases.

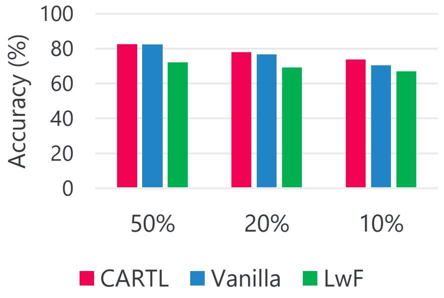
Method		**Case-4**		**Case-6**		**Case-8**	
Source	**Transfer**	**Acc (%)**	**Rob (%)**	**Acc (%)**	**Rob (%)**	**Acc (%)**	**Rob (%)**
AT	TL	83.22	25.23	86.92	25.38	90.82	18.54
AT	NEFT	83.72	26.29	86.87	27.95	90.92	29.97
AT + FDM	NEFT	85.51	28.47	87.79	30.94	91.30	29.34

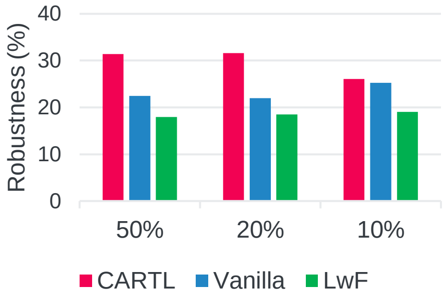
CIFAR-100 → CIFAR-10

Evaluations

- CARTL outperforms both LwF and Vanilla when the data size is small.

- Target models fine-tuned with CARTL inherit superior robustness from the source model in all cases.





Source	Target	Arch.	LwF		Vanilla		CARTL	
			Acc (%)	Rob (%)	Acc (%)	Rob (%)	Acc (%)	Rob (%)
CIFAR-100	SVHN	WRN 34-10	85.90	6.67	92.83	17.64	93.96	22.21
CIFAR-100	GTSRB	WRN 34-10	70.34	15.85	80.40	30.25	83.07	47.34
CIFAR-10	SVHN	WRN 28-4	94.32	4.68	94.86	11.52	94.76	21.65
GTSRB	SVHN	WRN 28-4	81.80	1.08	93.91	6.08	94.07	15.26

Conclusion

- We conduct detailed experiments and reveal that there is a trade-off between accuracy and robustness during transfer learning.

- We propose CARTL, consisted of FDM and NEFT, for improving the accuracy-robustness trade-off of the target model.

- We demonstrate that freezing affine parameters of Batch Norm layers can further boost the robustness transfer, and Batch Norm layers' statistics play a crucial role in robustness transfer.

Thanks!

Any questions?

qianwang@whu.edu.cn

https://github.com/NISP-official/CARTL