Decoupled Networks

Weiyang Liu, Zhen Liu, Zhiding Yu, Bo Dai, Rongmei Lin, Yisen Wang, James M.Rehg, Le Song
Georgia Institute of Technology, NVIDIA, Emory University, Tsinghua University, Ant Financial

CVPR 2018

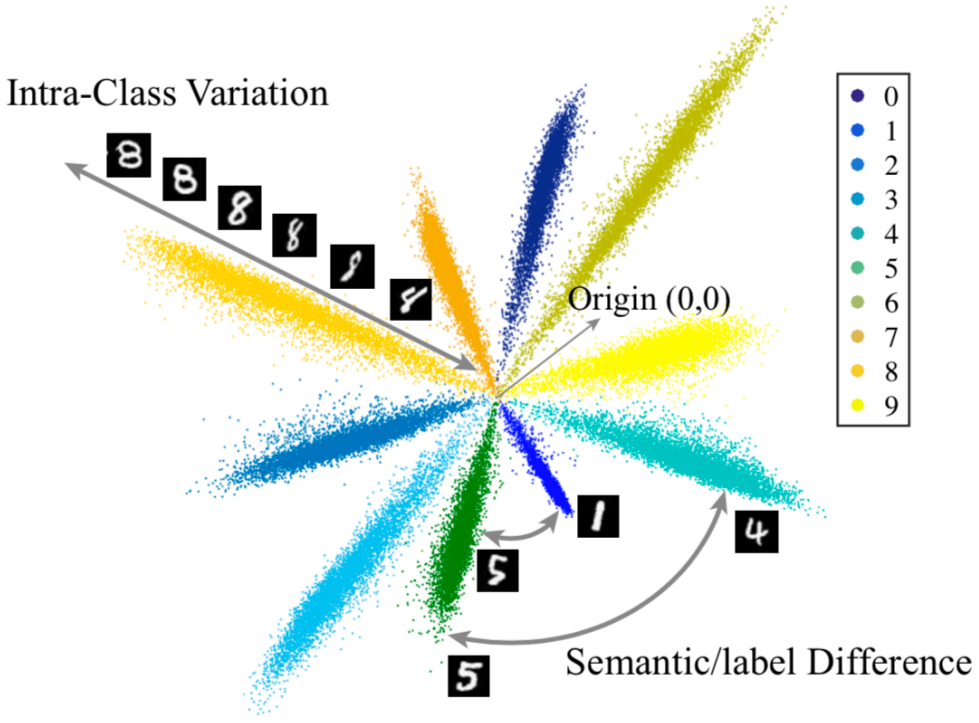
Sungman, Cho.

Introduction

Dive into FR(Face Recognition)

Abstract

- Propose a generic **decoupled learning framework** which **models the intra-class variation and semantic difference independently.**
- Reparametrize the inner product to a decoupled form
- Decoupled reparameterization renders significant performance gain with easier convergence and stronger robustness.



Introduction

- $< w, x >= w^T x$ couples the semantic difference and the intra-class variation in one unified measure.
- When the inner product between two samples is large, **one can not tell whether the two samples have large semantic difference or have large intra-class variation.**
- Propose a novel decoupled network (DCNet) by generalizing traditional Conv. operation.

$$f(\boldsymbol{w}, \boldsymbol{x}) = \langle \boldsymbol{w}, \boldsymbol{x} \rangle = \boldsymbol{w}^\top \boldsymbol{x},$$

$$f(\boldsymbol{w}, \boldsymbol{x}) = \|\boldsymbol{w}\| \, \|\boldsymbol{x}\| \, cos(\theta_{(\boldsymbol{w}, \boldsymbol{x})}),$$

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w}, \boldsymbol{x})}),$$

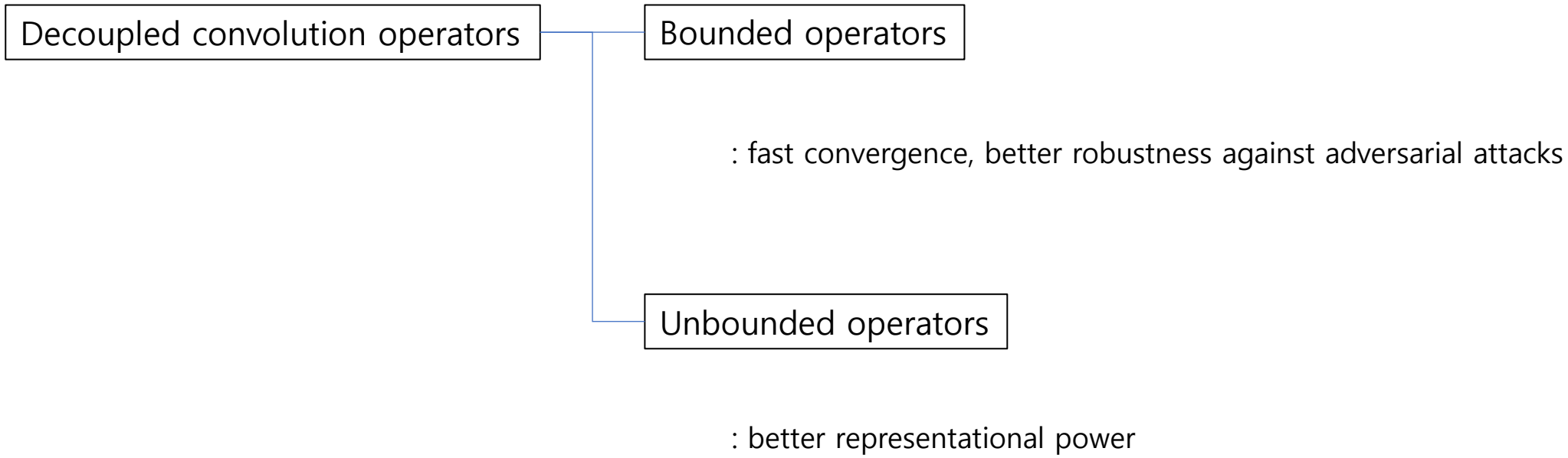
Introduction

magnitude function angular function

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

- magnitude function : models the intra-class variation
- angular function : models the semantic difference

Introduction



Contributions

- Propose an explicitly decoupled framework to study neural networks.
- By decoupling the inner product, we are able to design more effective magnitude and angular functions rather than the original convolution.
- DCNets have easier convergence, better accuracy and stronger robustness.

Related Works

- Large-margin softmax loss for convolutional neural networks, ICML, 2016
- Additive margin softmax for face verification, arXiv:1801.05599, 2018
- Improving face verification and person re-identification accuracy using hyperplane similiarity, ICCV, 2017
- Deep hypersphere embedding for face recognition, CVPR, 2017

1. **normalizes the weights of the last fully connected layer**
2. **specific area. (face recognition)**

Decoupled Networks

Decoupled Networks

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

The design of angular function $g(\cdot)$ is relatively easy but restricted, because it only takes the angle as input.

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

$h(\cdot)$ takes the norm of w and the norm of x as two inputs, therefore it is more complicated to design.

Decoupled Networks

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

$\|w\|$ is the intrinsic property of a kernel itself, corresponding to the importance of the kernel rather than the intra-class variation of the inputs.

Therefore, we tend not to include $\|w\|$ into the magnitude function.

Decoupled Networks

All kernels are assigned with equal importance.

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

Encourages the network to make decision based on as many kernels as possible and therefore may make the network generalize better.

However, *weighted decoupled operators* improve the representational power and may be useful when dealing with a large-scale dataset with numerous categories.

weighted decoupled operators :
combining $\|w\|$ back to $h(\cdot)$

Bounded Decoupled Operators

Bounded decoupled operators : $|f_d(\boldsymbol{w}, \boldsymbol{x})| \leq c$

Hyperspherical Convolution

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

$$h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) = \alpha$$

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

Bounded Decoupled Operators

Bounded decoupled operators : $|f_d(\boldsymbol{w}, \boldsymbol{x})| \leq c$

Hyperspherical Convolution

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

$$h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) = \alpha$$

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

controls the output scale

Bounded Decoupled Operators

Bounded decoupled operators : $|f_d(\boldsymbol{w}, \boldsymbol{x})| \leq c$

Hyperspherical Convolution

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

$$h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) = \alpha$$

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

geodesic distance on the unit hypersphere and typically outputs value from -1 to 1

Bounded Decoupled Operators

Bounded decoupled operators : $|f_d(\boldsymbol{w}, \boldsymbol{x})| \leq c$

Hyperspherical Convolution

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

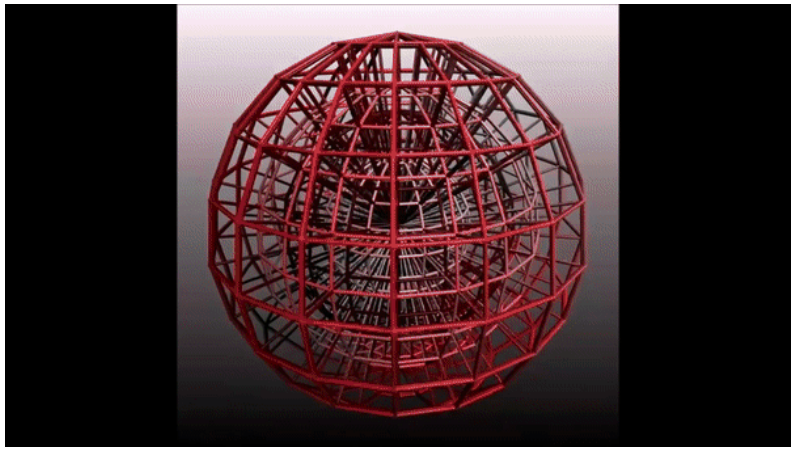
$$h(\|\boldsymbol{w}\|, \|\boldsymbol{x}\|) = \alpha$$

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

output : $[-\alpha, \alpha]$

Bounded Decoupled Operators

Bounded decoupled operators : $|f_d(\boldsymbol{w}, \boldsymbol{x})| \leq c$



HyperSphere

Hyperspherical Convolution

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

Usually, we can use $\alpha = 1$, which reduces to SphereConv in this case.

Projecting w and x to a hypersphere and then performing inner product. (if $g(\theta) = \cos(\theta)$)

Improves the problem conditioning in neural networks, making the network converge better.

Bounded Decoupled Operators

Bounded decoupled operators : $|f_d(\boldsymbol{w}, \boldsymbol{x})| \leq c$

Hyperball Convolution

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot \frac{\min(\|\boldsymbol{x}\|, \rho)}{\rho} \cdot g(\theta_{(\boldsymbol{w}, \boldsymbol{x})}),$$

ρ controls the saturation threshold for the input norm $\|x\|$

Projecting w to a hypersphere, x to a hyperball, then performing the inner product. (if $g(\theta) = \cos(\theta)$)

More robust and flexible than SphereConv

Bounded Decoupled Operators

Is BallConv more robust and flexible than SphereConv ?

SphereConv : $f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$

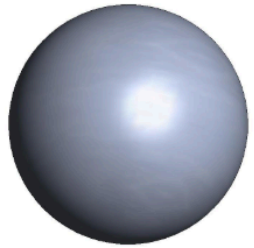
x with small $\|x\|$ and the same direction as w could still produce maximum output

Appendix. Ball vs Sphere

Ball:

A ball is the shape of a circle. It is used in all the common ball games these days such as the sports of football, basketball, cricket and squash. Hockey is also a game in which ball is used and it is totally round in appearance.

Image Courtesy: istockphoto.com



Sphere:

A sphere is a shape almost the same as a ball, but it is a bit flatter than the ball and is most commonly used in the game of rugby these days.

Image Courtesy: istockphoto.com

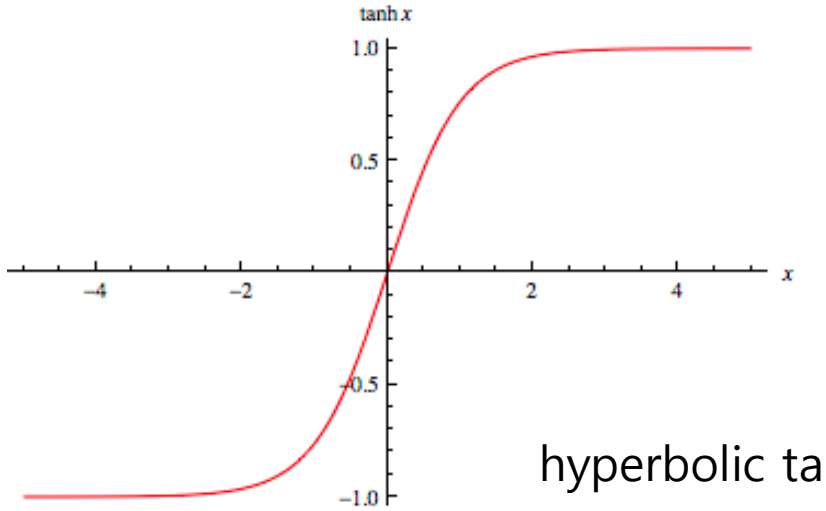
ref: http://www.stepbystep.com/difference-between-ball-and-sphere-99659/

Bounded Decoupled Operators

Bounded decoupled operators : $|f_d(\boldsymbol{w}, \boldsymbol{x})| \leq c$

Hyperbolic Tangent Convolution

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \tanh\left(\frac{\|\boldsymbol{x}\|}{\rho}\right) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$



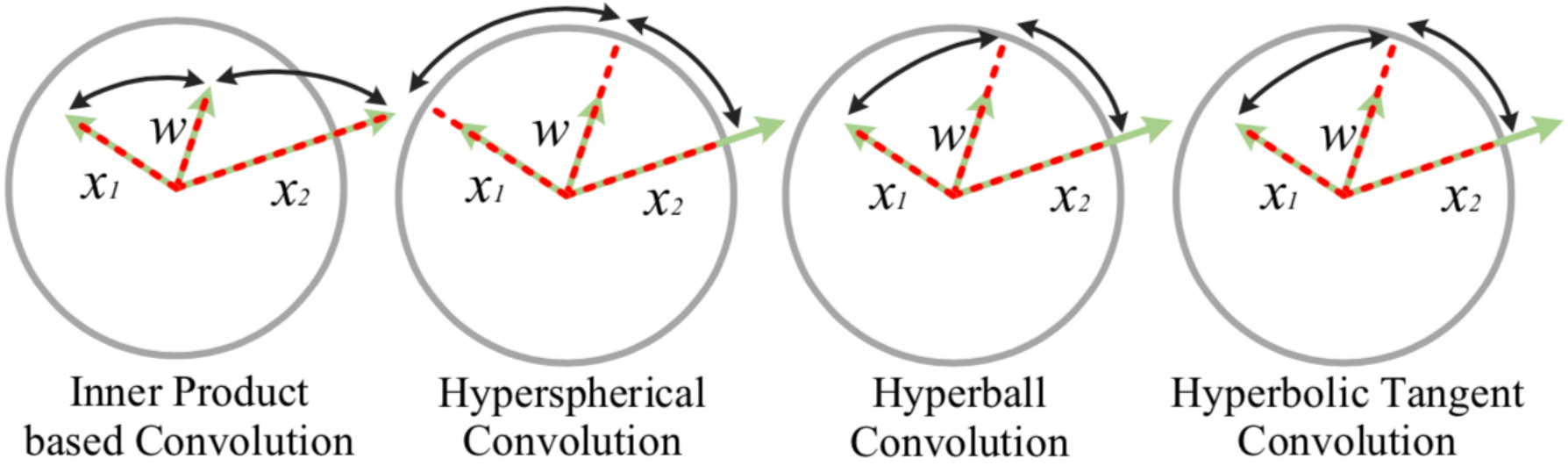
hyperbolic tangent

ρ is parameter controlling the decay curve.

The TanhConv can be viewed as a smooth version of BallConv.
(more convergence gain due to its smoothness)

More convergence gain

Bounded Decoupled Operators



Inner Product based Convolution

Hyperspherical Convolution

Hyperball Convolution

Hyperbolic Tangent Convolution

Unbounded Decoupled Operators

Linear Convolution

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \, \|\boldsymbol{x}\| \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

α controls the output scale.

LinearConv projects the w to a hypersphere and has a parameter to control the slope

Unbounded Decoupled Operators

Segmented Convolution.

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \begin{cases} \alpha \|\boldsymbol{x}\| \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}), & 0 \leq \|\boldsymbol{x}\| \leq \rho \\ (\beta \|\boldsymbol{x}\| + \alpha\rho - \beta\rho) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}), & \rho < \|\boldsymbol{x}\| \end{cases},$$

α controls the slope when $\|x\| \leq \rho$, β controls the slope when $\|x\| \geq \rho$

Both LinearConv and BallConv are special cases of SegConv.

Unbounded Decoupled Operators

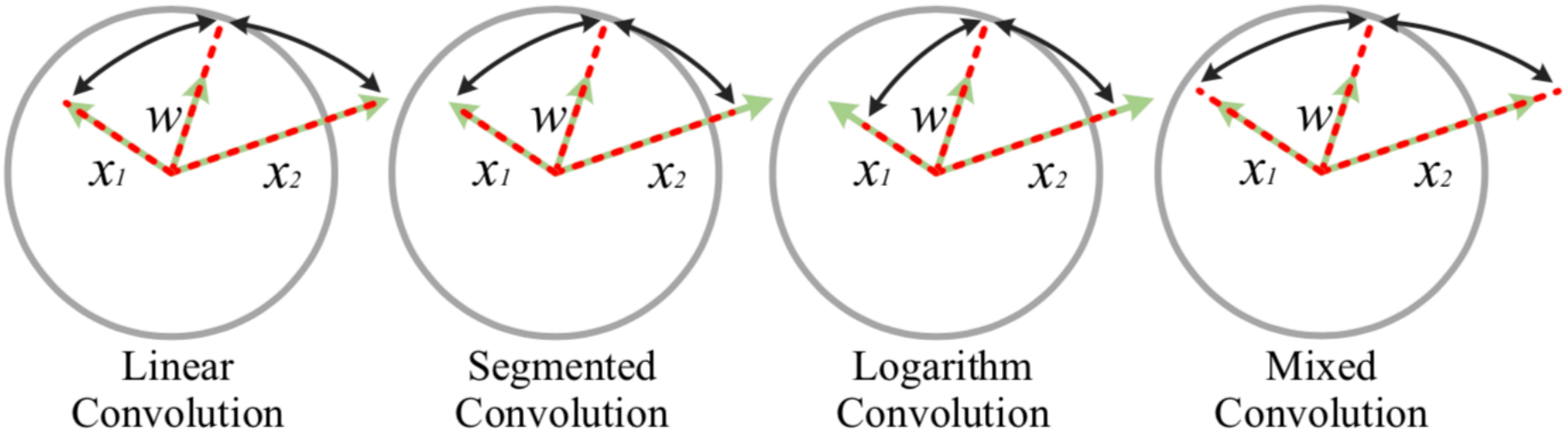
Logarithm Convolution.

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \log(1 + \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

Mixed Convolution.

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \left(\alpha \|\boldsymbol{x}\| + \beta \log(1 + \|\boldsymbol{x}\|)\right) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

Unbounded Decoupled Operators



3 Properties of Decoupled Operators

Operator Radius.

describe the gradient change point of the magnitude function.

BallConv $$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot \frac{\min(\|\boldsymbol{x}\|, \rho)}{\rho} \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

SegConv $$f_d(\boldsymbol{w}, \boldsymbol{x}) = \begin{cases} \alpha \|\boldsymbol{x}\| \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}), & 0 \leq \|\boldsymbol{x}\| \leq \rho \\ (\beta \|\boldsymbol{x}\| + \alpha\rho - \beta\rho) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}), & \rho < \|\boldsymbol{x}\| \end{cases},$$

3 Properties of Decoupled Operators

Boundedness

affect network's convergence speed and robustness.

- Lead to better problem conditioning in training a deep network via stochastic gradient descent.
- Make the variance of the output small and partially address the internal covariate shift problem.
- Constrain the Lipschitz constant of a neural network, making the entire network more smooth.

  (Lipschitz constant of a neural network is shown to be closely related to its robustness against adversarial perturbation)

Design of the Angular Function

Linear angular activation

$$g(\theta_{(\boldsymbol{w},\boldsymbol{x})}) = -\frac{2}{\pi}\theta_{(\boldsymbol{w},\boldsymbol{x})} + 1,$$

Cosine angular activation

$$g(\theta_{(\boldsymbol{w},\boldsymbol{x})}) = \cos(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$$

Sigmoid angular activation

$$g(\theta_{(\boldsymbol{w},\boldsymbol{x})}) = \frac{1+\exp(-\frac{\pi}{2k})}{1-\exp(-\frac{\pi}{2k})} \cdot \frac{1-\exp(\frac{\theta_{(\boldsymbol{w},\boldsymbol{x})}}{k} - \frac{\pi}{2k})}{1+\exp(\frac{\theta_{(\boldsymbol{w},\boldsymbol{x})}}{k} - \frac{\pi}{2k})},$$

Square cosine angular activation

$$g(\theta_{(\boldsymbol{w},\boldsymbol{x})}) = \text{sign}(\cos(\theta)) \cdot \cos^2(\theta),$$

Design of the Angular Function

Linear angular activation $g(\theta_{(\boldsymbol{w},\boldsymbol{x})}) = -\frac{2}{\pi}\theta_{(\boldsymbol{w},\boldsymbol{x})} + 1,$

Sigmoid angular activation $g(\theta_{(\boldsymbol{w},\boldsymbol{x})}) = \frac{1+\exp(-\frac{\pi}{2k})}{1-\exp(-\frac{\pi}{2k})} \cdot \frac{1-\exp(\frac{\theta_{(\boldsymbol{w},\boldsymbol{x})}}{k} - \frac{\pi}{2k})}{1+\exp(\frac{\theta_{(\boldsymbol{w},\boldsymbol{x})}}{k} - \frac{\pi}{2k})},$

Cosine angular activation $g(\theta_{(\boldsymbol{w},\boldsymbol{x})}) = \cos(\theta_{(\boldsymbol{w},\boldsymbol{x})}),$

Square cosine angular activation $g(\theta_{(\boldsymbol{w},\boldsymbol{x})}) = \text{sign}(\cos(\theta)) \cdot \cos^2(\theta),$

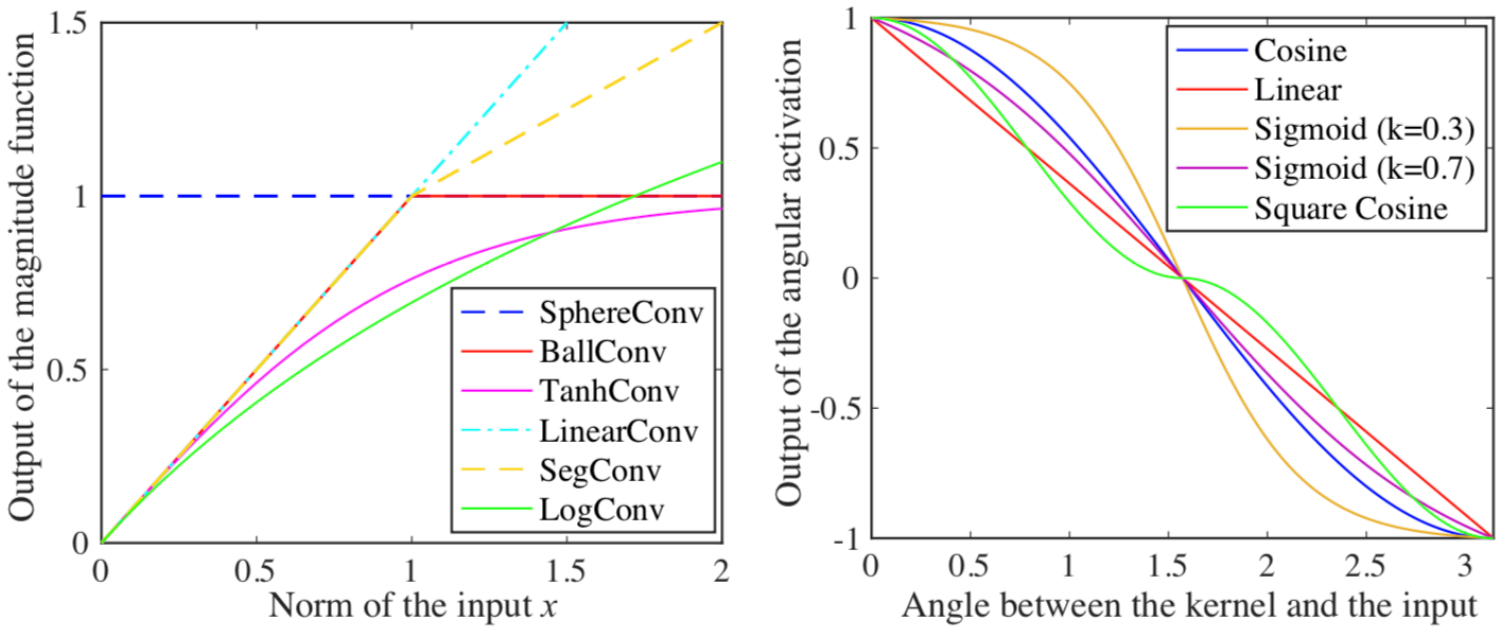


Figure 3: Magnitude function ($\rho = 1$) and angular activation function.

Weighted Decoupled Operators

Although empirically we find that the standard decoupled operators work better than the weighted ones in most cases, we still consider weighted decoupled operators, in order to improve the operator's flexibility

Linearly Weighted Decoupled Operator.

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}), \longrightarrow f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \|\boldsymbol{w}\| \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})})$$

Nonlinearly Weighted Decoupled Operator.

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \tanh\left(\frac{\|\boldsymbol{x}\|}{\rho}\right) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}), \longrightarrow f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \tanh(\frac{1}{\rho} \|\boldsymbol{x}\| \cdot \|\boldsymbol{w}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}).$$

$$f_d(\boldsymbol{w}, \boldsymbol{x}) = \alpha \tanh(\frac{1}{\rho} \|\boldsymbol{w}\|) \cdot \tanh(\frac{1}{\rho} \|\boldsymbol{x}\|) \cdot g(\theta_{(\boldsymbol{w},\boldsymbol{x})}).$$

Learnable Decoupled Operators

By making hyperparameters learnable,

Representational power ← Trade Off → Prohibitive amount of training data

In order to achieve an effective trade-off,

use only one learning parameter : ρ (operator radius)

Improving the Optimization

Weight Projection

Large $\|w\|$ can make the gradients very small.

$$\frac{\partial}{\partial \boldsymbol{w}}\left(\hat{\boldsymbol{w}}^{\top}\hat{\boldsymbol{x}}\right)=\frac{\hat{\boldsymbol{x}}-\hat{\boldsymbol{w}}^{\top}\hat{\boldsymbol{x}}\cdot\hat{\boldsymbol{w}}}{\|\boldsymbol{w}\|}, \quad \text{where } \hat{\boldsymbol{w}}=\boldsymbol{w}/\|\boldsymbol{w}\| \text{ and } \hat{\boldsymbol{x}}=\boldsymbol{x}/\|\boldsymbol{x}\|.$$

Weight projection performs $\boldsymbol{w} \leftarrow s \cdot \hat{\boldsymbol{w}}$
$\boldsymbol{s}$ is positive constant which controls the norm of the gradient (use $\boldsymbol{s} = 1$ in experiments)

Weight Gradients

We could simply multiply $\|w\|$.

$$\Delta \boldsymbol{w} = \|\boldsymbol{w}\| \cdot \frac{\partial}{\partial \boldsymbol{w}} \left(\hat{\boldsymbol{w}}^\top \hat{\boldsymbol{x}} \right) = \hat{\boldsymbol{x}} - \hat{\boldsymbol{w}}^\top \hat{\boldsymbol{x}} \cdot \hat{\boldsymbol{w}},$$

Experiments

CIFAR-10 & CIFAR-100

Weighted Decoupled Operators & Optimization Tricks

Method	Error
Linearly Weighted Decoupled Operator	22.95
Nonlinearly Weighted Decoupled Operator (Eq. (15))	23.03
Nonlinearly Weighted Decoupled Operator (Eq. (16))	23.38
Decoupled Operator (Standard Gradients)	23.09
Decoupled Operator (Weight Projection)	**21.17**
Decoupled Operator (Weighted Gradients)	21.45

Weighted decoupled operators do not show obvious advantages.

CIFAR-10 & CIFAR-100

Weighted Decoupled Operators & Optimization Tricks

Method	Error
Linearly Weighted Decoupled Operator	22.95
Nonlinearly Weighted Decoupled Operator (Eq. (15))	23.03
Nonlinearly Weighted Decoupled Operator (Eq. (16))	23.38
Decoupled Operator (Standard Gradients)	23.09
Decoupled Operator (Weight Projection)	**21.17**
Decoupled Operator (Weighted Gradients)	21.45

Weight projection and weighted gradients work much better than the competing methods.

CIFAR-10 & CIFAR-100

Learning without Batch Normalization

Method	Linear	Cosine	Sq. Cosine
CNN Baseline	-	35.30	-
LinearConv	33.39	31.76	N/C
TanhConv	**32.88**	31.88	**34.26**
SegConv	34.69	**30.34**	N/C

Ours can perform much better than the original convolution even without BN

CIFAR-10 & CIFAR-100

Learning without ReLU.

Method	Cosine w/o ReLU	Sq. Cosine w/o ReLU	Cosine w/ ReLU	Sq. Cosine w/ ReLU
Baseline	58.24	-	26.01	-
SphereConv	33.31	25.90	26.00	26.97
BallConv	**31.81**	25.43	25.18	26.48
TanhConv	32.27	25.27	25.15	26.94
LinearConv	36.49	24.36	**24.81**	25.14
SegConv	33.57	**24.29**	24.96	**25.04**
LogConv	33.62	24.91	25.17	25.85
MixConv	33.46	24.93	25.27	25.77

Table 3: Testing error rate (%) of plain CNN-9 on CIFAR-100. Note that, BN is used in all compared models. Baseline is the original plain CNN-9.

Ours can perform much better than the original convolution even without BN

CIFAR-10 & CIFAR-100

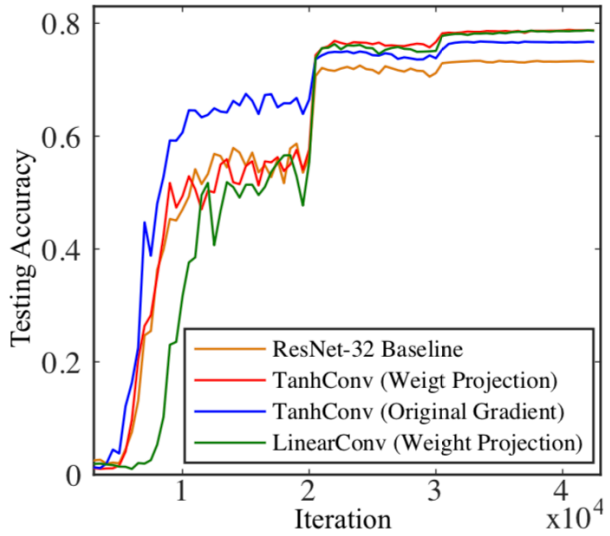
Comparison among Different Decoupled Operators.

Method	Linear	Cosine	Sq. Cosine
ResNet Baseline	-	26.69	-
SphereConv	21.79	21.44	24.40
BallConv	21.44	21.12	24.31
TanhConv	21.6	21.17	24.77
LinearConv	21.09	22.17	21.31
SegConv	**20.86**	**20.91**	**20.88**
LogConv	21.84	21.08	22.86
MixConv	21.02	21.28	21.81

Table 4: Testing error rate (%) of ResNet-32 on CIFAR-100.

CIFAR-10 & CIFAR-100

Convergence



Acc.

Method	CIFAR-10	CIFAR-100
ResNet-110-original [5]	6.61	25.16
ResNet-1001 [6]	4.92	**22.71**
ResNet-1001 (64 mini-batch size) [6]	**4.64**	-
DCNet-32 (TanhConv + Cosine)	**4.75**	21.12
DCNet-32 (LinearConv + Sq. Cos.)	5.34	**20.23**

Table 5: Comparison to the state-of-the-art on CIFAR-10 and CIFAR-100.

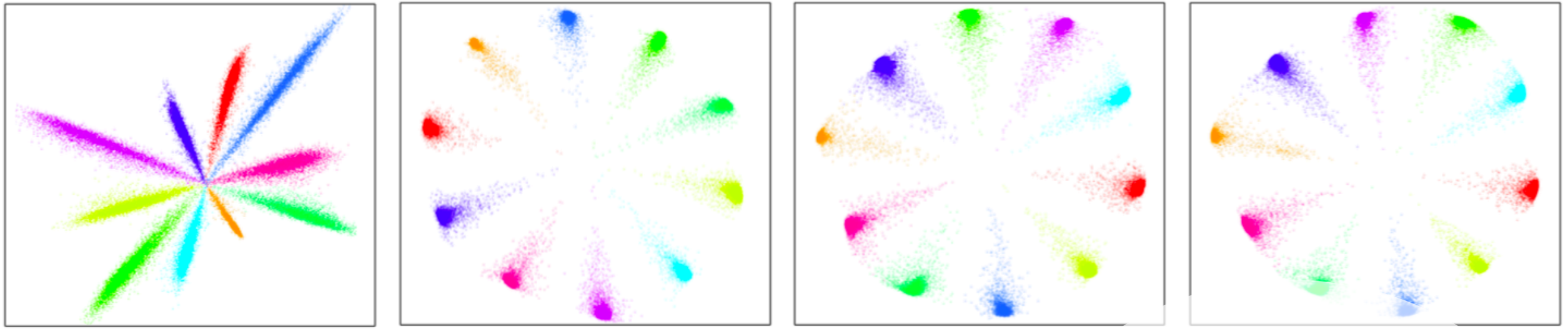
Adversarial attacks

Attack	Target models			
	Baseline	SphereConv	BallConv	TanhConv
None	85.35	88.58	91.13	**91.45**
FGSM	50.90	**56.71**	49.50	50.61
BIM	36.22	**43.10**	27.48	29.06
None	83.70	87.41	87.47	**87.54**
FGSM	77.57	76.29	78.67	**80.38**
BIM	78.55	77.79	80.59	**82.47**

Table 8: Black-box attacks on CIFAR-10. Performance is measured in accuracy (%). The first three rows are results of naturally trained models, and the last three rows are results of adversarially trained models.

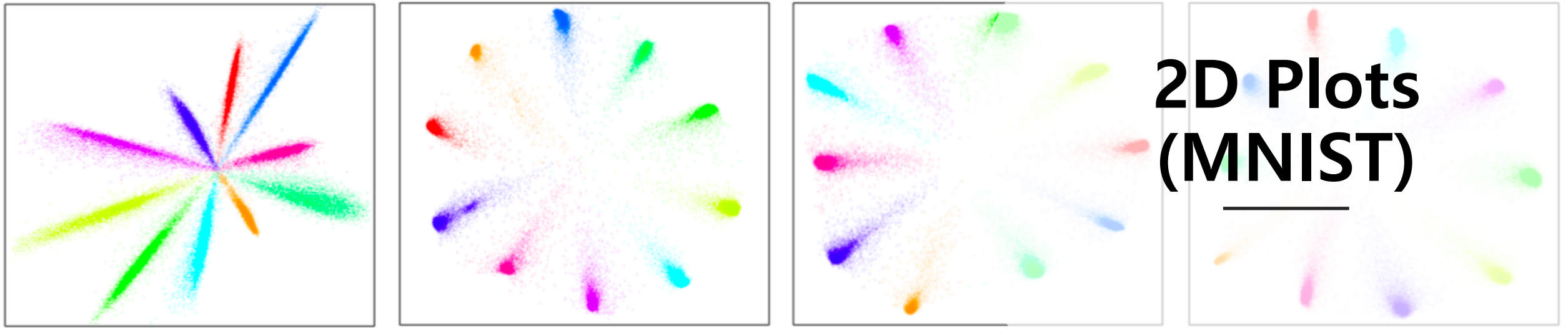
Attack	Target models			
	Baseline	SphereConv	BallConv	TanhConv
None	85.35	88.58	91.13	**91.45**
FGSM	18.82	43.64	50.47	**52.60**
BIM	8.67	8.89	7.74	**10.18**
None	83.70	87.41	87.47	**87.54**
FGSM	78.96	**85.98**	82.20	81.46
BIM	7.96	**35.07**	17.38	19.86

Table 7: White-box attacks on CIFAR-10. Performance is measured in accuracy (%). The first three rows are results of naturally trained models, and the last three rows are results of adversarially trained models.



Baseline CNN (Original Convolution) DCNet (SphereConv) DCNet (BallConv) DCNet (TanhConv)

Figure 6: 2D feature visualization on MNIST dataset with natural training.



Baseline CNN (Original Convolution) DCNet (SphereConv) DCNet (BallConv) DCNet (TanhConv)

Figure 7: 2D feature visualization on MNIST dataset with adversarial training.

2D Plots (MNIST)

Thanks.