Necessary Ingredients of a Model

박우준

Vision & Display Systems Lab.

Sogang University

Outline

- Necessary Ingredients of a Model
  - Inductive bias
  - Without convolution weights
    - Training BatchNorm and Only BatchNorm:
      On the Expressive Power of Random Features in CNNs (ICLR 2021)
  - Without pretraining
    - Generic Perceptual Loss for Modeling Structured Output Dependencies (CVPR 2021)
    - Rethinking and Improving the Robustness of Image Style Transfer (CVPR 2021)
  - Conclusion

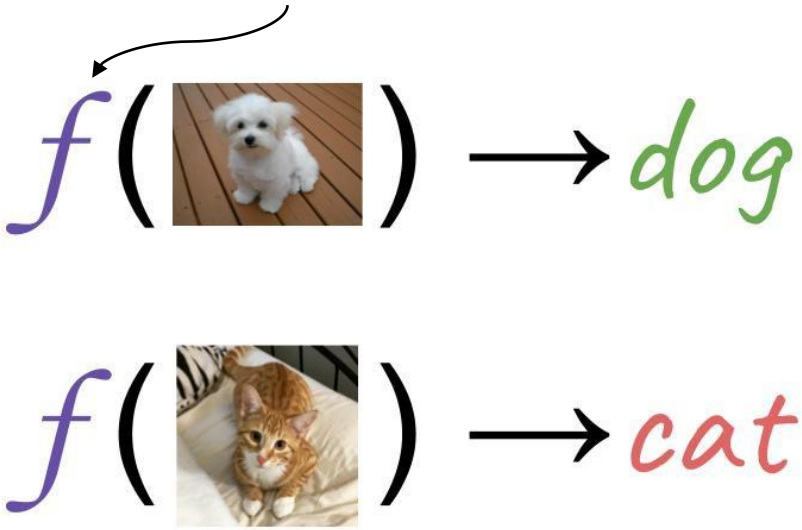
Inductive bias

Inductive bias

: Definition

- The inductive bias
  - A set of assumptions that the learner uses to predict outputs of given inputs that it has not encountered [1]
    - Assumptions about the nature of the **target function**

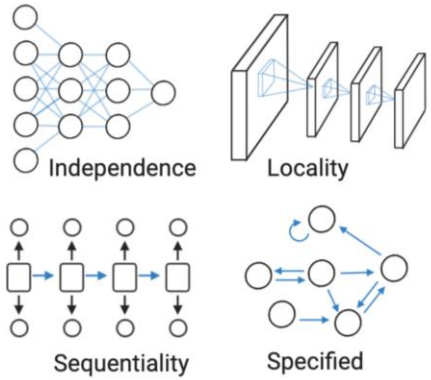
What we expect from this function

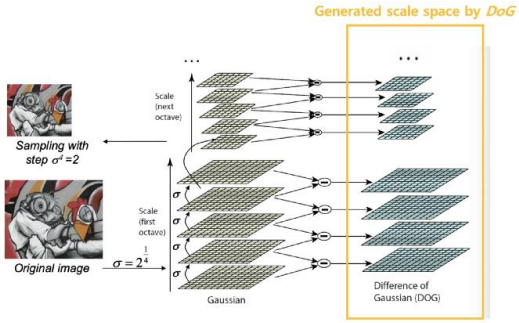


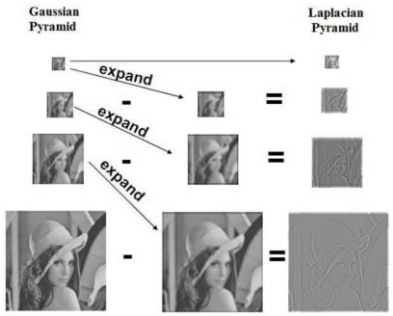
Inductive bias

: Definition

- Examples of inductive biases
  - Relational bias
    - Image classification using locality
      - Combined information from **neighboring pixels**
        → concatenate and abstract
        → higher level information
        → determine the class
  - Additional features
    - Edge map, multi-scale, frequency separation, base-detail separation, residual leaning, non-local information …







Inductive bias

: Necessary Ingredients of a Model

- Trained weights
  - Most of the parameters are **convolution weights**
- Pretraining
  - Large visual representation datasets, pretext tasks
    - e.g. ImageNet

Inductive bias

: Necessary Ingredients of a Model

- ~~Trained weights~~
  - Most of the parameters are **convolution weights**
- ~~Pretraining~~
  - Large visual representation datasets, pretext tasks
    - e.g. ImageNet

- Are these necessary?

Without convolution weights

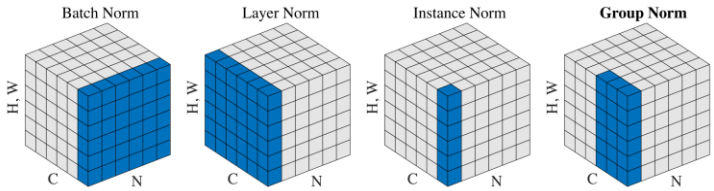
Without convolution weights

: Training BatchNorm and Only BatchNorm (ICLR 2021) [2]

- Problem definition
  - Aims to understand the **role** and **expressive power** of batch normalization parameters
    - Scale parameter (=coefficient) γ, shift parameter (=bias) β
- Findings
  - **Training only γ and β** can reach surprisingly high accuracy
    - Tasks: CIFAR-10, ImageNet classification
    - Model: Various settings of ResNet (depth, width)
    - Conditions: Sufficiently deep networks
    - *This expressive power results from their particular position as a per-feature coefficient and bias*
  - **The role of γ** differs
    - When training only BatchNorm parameters
      - Learns to disable channels (per-feature sparsity)
    - When training all parameters
      - Moderating activations

Without convolution weights

: Batch normalization



- Algorithm
  - Conv → BN (BatchNorm) → Activation

Algorithm 1 Batch normalization at train-time.

1: Let $x^{(1)}, ..., x^{(n)}$ be the pre-activations for a particular unit in a neural network for inputs 1 through n in a mini-batch.
2: Let $\mu = \frac{1}{n}\sum_{i=1}^{n} x^{(i)}$
3: Let $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x^{(i)} - \mu)^2$
4: The batch-normalized pre-activation $\hat{x}^{(i)} = \gamma\frac{x^{(i)}-\mu}{\sqrt{\sigma^2}} + \beta$ where γ and β are trainable parameters.
5: The activations are $f(\hat{x}^{(i)})$ where f is the activation function.

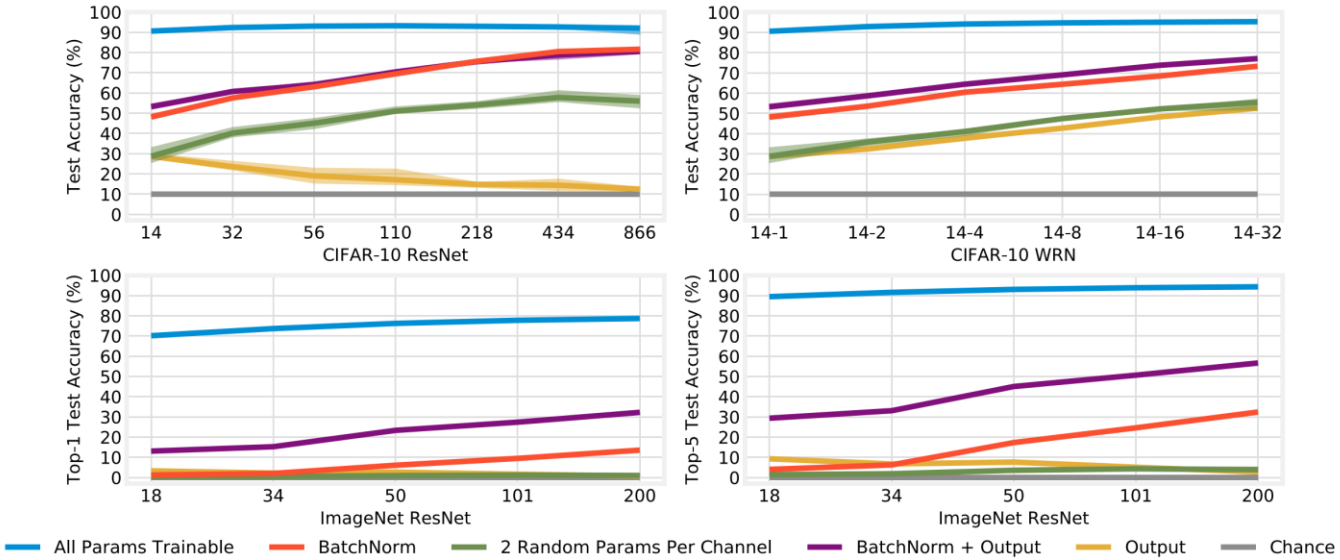
- Parameter count

Family	ResNet for CIFAR-10							Wide ResNet (WRN) for CIFAR-10						ResNet for ImageNet				
Depth	14	32	56	110	218	434	866	14	14	14	14	14	14	18	34	50	101	200
Width Scale	1	1	1	1	1	1	1	1	2	4	8	16	32	1	1	1	1	1
Total	175K	467K	856K	1.73M	3.48M	6.98M	14.0M	175K	696K	2.78M	11.1M	44.3M	177M	11.7M	21.8M	25.6M	44.6M	64.7M
BatchNorm	1.12K	2.46K	4.26K	8.29K	16.4K	32.5K	64.7K	1.12K	2.24K	4.48K	8.96K	17.9K	35.8K	9.6K	17.0K	53.1K	105K	176K
Output	650	650	650	650	650	650	650	650	1.29K	2.57K	5.13K	10.3K	20.5K	513K	513K	2.05M	2.05M	2.05M
Shortcut	2.56K	2.56K	2.56K	2.56K	2.56K	2.56K	2.56K	2.56K	10.2K	41.0K	164K	655K	2.62M	172K	172K	2.77M	2.77M	2.77M
BatchNorm	0.64%	0.53%	0.50%	0.48%	0.47%	0.47%	0.46%	0.64%	0.32%	0.16%	0.08%	0.04%	0.02%	0.08%	0.08%	0.21%	0.24%	0.27%
Output	0.37%	0.14%	0.08%	0.04%	0.02%	0.01%	0.01%	0.37%	0.19%	0.09%	0.05%	0.02%	0.01%	4.39%	2.35%	8.02%	4.60%	3.17%
Shortcut	1.46%	0.55%	0.30%	0.15%	0.07%	0.04%	0.02%	1.46%	1.47%	1.47%	1.48%	1.48%	1.48%	1.47%	0.79%	10.83%	6.22%	4.28%

$< 0.7\%$

Without convolution weights

: The expressive power of the affine parameters γ and β

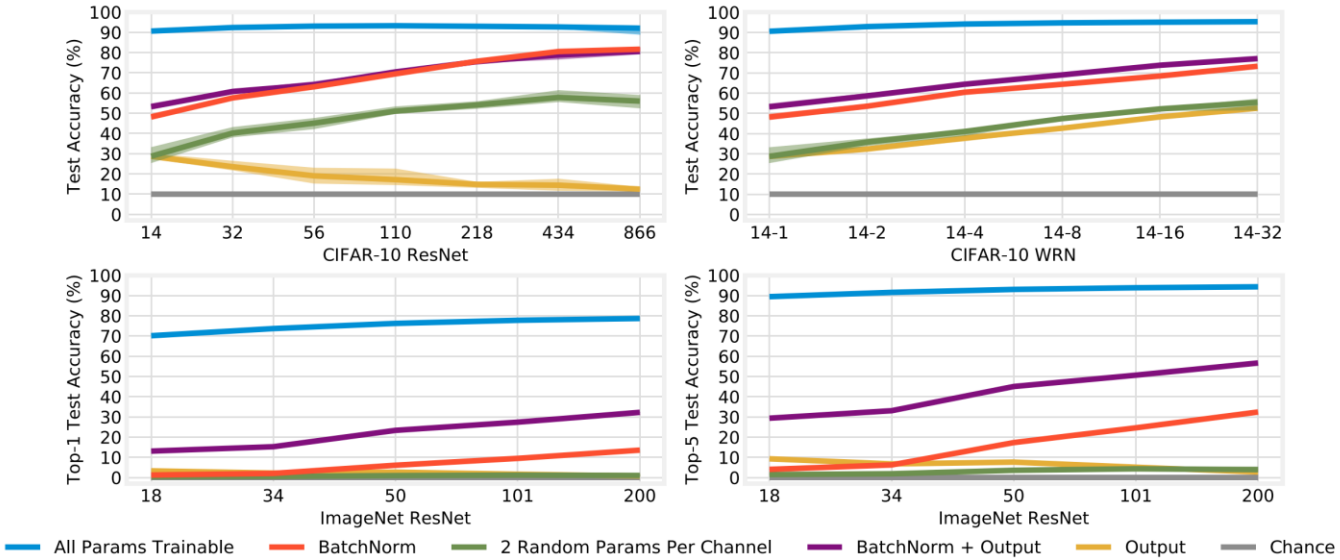


- Blue: training all parameters
- Red: training **only BN parameters**
- Green: training 2 random parameters per channel

  = same number of parameters trained w/ BN only training
- Purple: training BN parameters + linear output layer
- Yellow: training only linear output layer

Without convolution weights

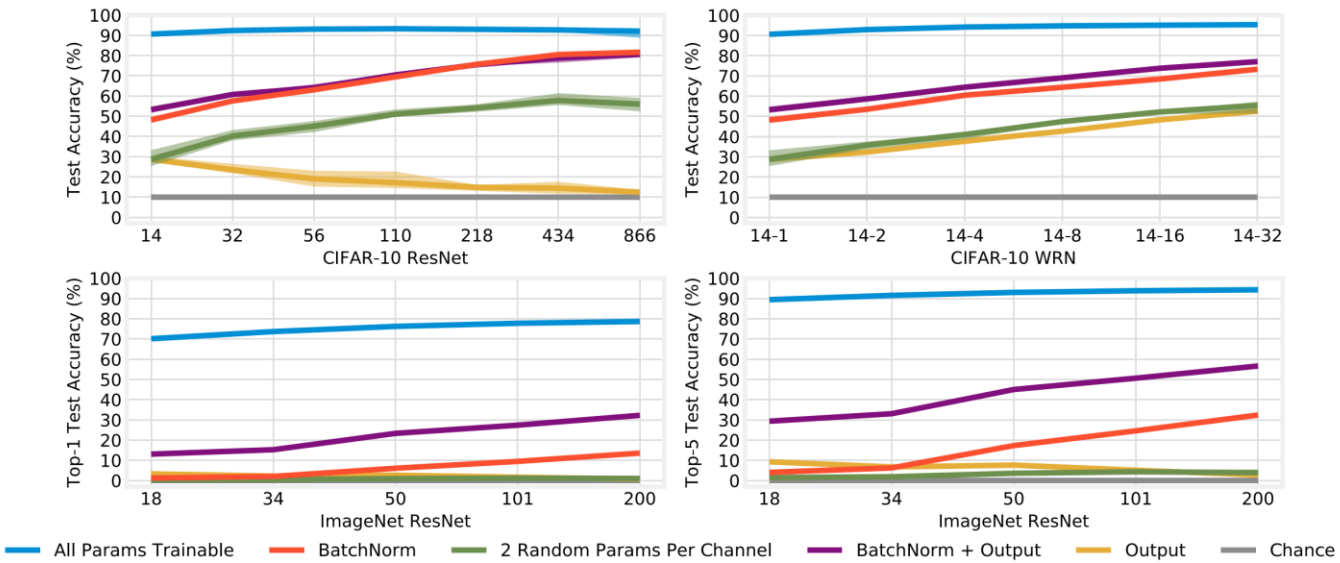
: The expressive power of the affine parameters γ and β



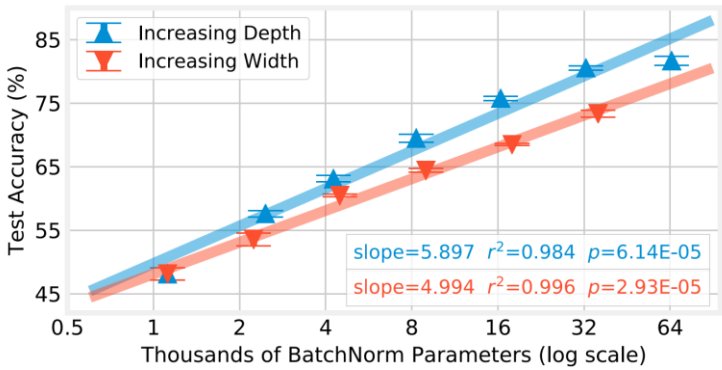
- The expressive power of neural networks composed of **random features**
  - Even though training just < 0.7% parameters the network achieves surprisingly high accuracy (Red)
    - Rest of parameters are just **random initialized weights**
    - Trained BN parameters can only **shift** and **rescale** random features
  - Training BN + the **output layer** improves ImageNet top-5 accuracy by about 25% (Purple)

Without convolution weights

: The expressive power of the affine parameters γ and β

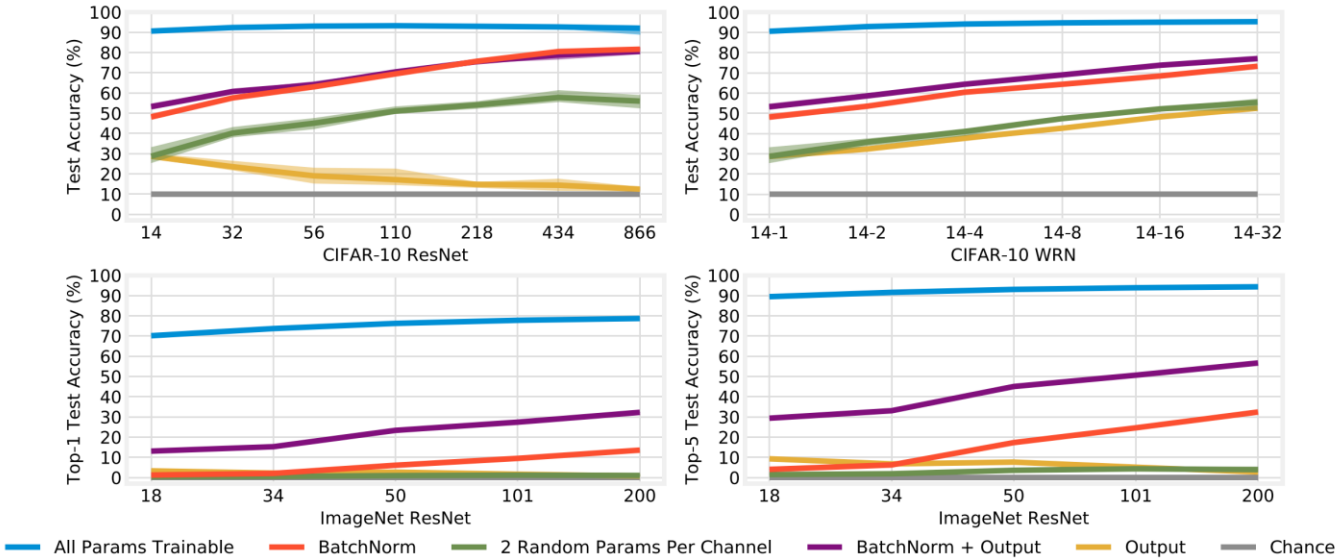


- Does increasing **depth** and **width** gives better results?
  - *The **expressivity** of the network increased by increasing the number of features available for the BN parameters to combine*
- Is it always better to increase **depth** rather than width?



Without convolution weights

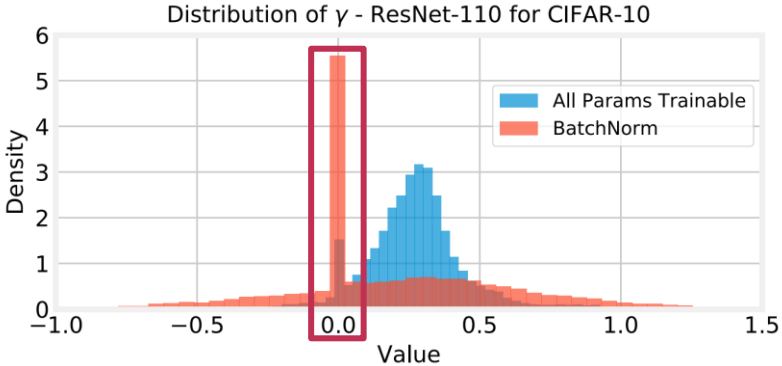
: The expressive power of the affine parameters γ and β

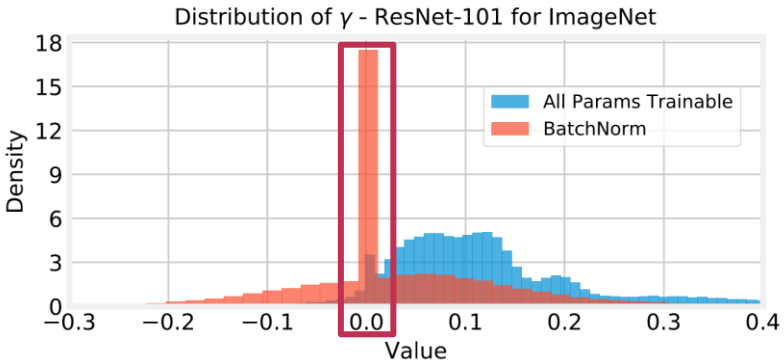


- Are BN parameters special? (Green)
  - γ and β have a greater impact on accuracy than other parameters
    - Coarse-grained control over entire random features

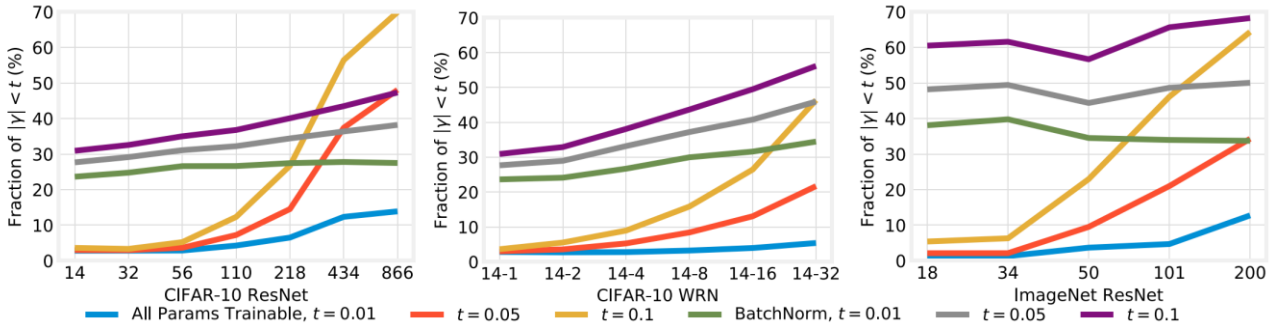
Without convolution weights

: Training only BN sparsifies activations



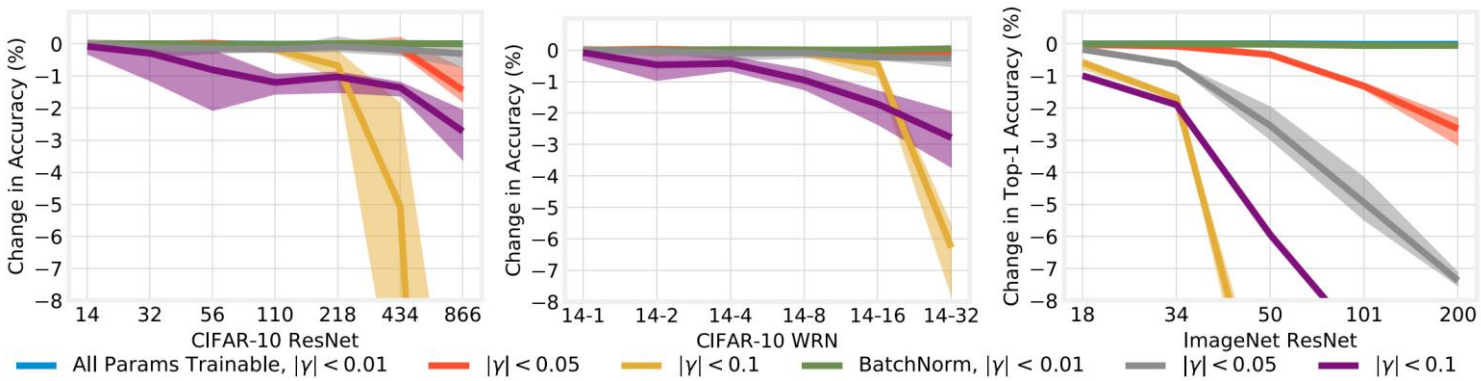


- When training only BN parameters
  - $|\gamma| < 0.01$ are 20% ~ 30%
    - Network seemingly learns to **disable** 20% ~ 40% of entire activations (green)



Without convolution weights

: Small γ disable features



- Both training schemes
  - Clamping all values of $|\gamma| < 0.01$ to 0 does not affect accuracy (Green)
    - A naturally induced feature-level **sparsity**

Without convolution weights

: Discussions

- Random features
  - A novel way of training networks constructed out of random features
  - Reduce the cost of storing networks at inference-time
    - Only need to store:
      - 1) The random seed necessary to generate the network's weights
      - 2) The trained BN parameters

- Utilize convolution filters that are eliminated by γ
  - BN only train → re-initialize $|\gamma| < 0.01$ filters
    → BN + N% conv. param. train → re-initialize $|\gamma| < 0.01$ filters →
    → BN + (sN)% conv. param. train → re-initialize $|\gamma| < 0.01$ filters →
    → BN + (s^2N)% conv. param. train → re-initialize $|\gamma| < 0.01$ filters →
    … → All param. train → Done!

Without pretraining

Without pretraining

: Generic Perceptual Loss (CVPR 2021) [3]

- Previous assumptions
  - Success of the perceptual loss lies in the high-level perceptual feature representations extracted from CNNs **pretrained with a large set of images** (e.g. ImageNet)
- Findings
  - **Network structure** > trained weights
    - *Without any learning, the structure of a deep network is sufficient to capture the dependencies between multiple levels of variable statistics using multiple layers of CNNs*
  - **Dense per-pixel prediction** can be improved by adding the randomized perceptual loss
    - Super resolution. semantic segmentation, instance segmentation and depth estimation

Without pretraining

: Super resolution results

(a) Ground Truth **(b)** Bicubic **(c)** Pixel-wise Loss alone [16] **(d)** w. Pretrained VGG [16] **(e)** w. Random VGG

- Baseline: SRResNet [4] (x4)

Without pretraining

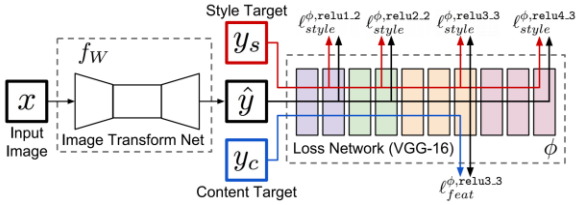
: Perceptual loss [5]

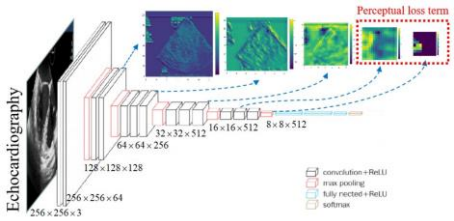
- Definition

$$\ell_{feat}^{\phi,j}(\hat{y}, y) = \frac{1}{C_j H_j W_j} \|\phi_j(\hat{y}) - \phi_j(y)\|_2^2$$

  - $y, \hat{y}$: target images, synthesized image
  - ϕ_j: the perceptual function which outputs the activation of the jth layer in the perceptual loss network

- In practice
  - Different networks
    - Original perceptual loss: VGG-16
    - SRResNet: VGG-19
  - Different layers
  - Different error metric
    - L1, L2
    - Gram matrix
      - matrix multiplication of the each of the activations and the activation matrix's transpose





Without pretraining

: Initialization scheme

- Scheme
  - A zero-mean Gaussian with a standard deviation of $\sqrt{2/n_l}$
    - l, k : layer index, kernel size
    - c_l: input channel
    - $n_l = k^2 c_l$

Init. scheme	mIoU (%)
$\mathcal{N}(0,1)$	–
$\mathcal{N}(0,0.1)$	45.6
$\mathcal{N}(0,0.01)$	68.5
$\mathcal{U}[-1,1]$	–
$\mathcal{U}[-0.1,0.1]$	51.7
$\mathcal{U}[-0.01,0.01]$	69.2
Xavier-normal	69.8
Ours	**71.3**

- Pro
  - *It is important to guarantee that each layer is bounded by a **Lipschitz constant** close to 1, so that the gradient generated by the random network will not **explode** or **vanish**.*
  - *Here we investigate the **gradient scales** in the random network and derive a **robust initialization** method.*
  - *Then the **covariance** becomes close to the variance of the two embeddings and this bound tends to zero.*

- … But isn't that what the **batch normalization** is supposed to do?

Without pretraining

: Searching for the Goldilocks network structure

- Different network families
  - Training with random weights and the pre-trained weights show **almost no difference**
    - Difference around 0.02% to 0.06%
  - **Network structures** lead to a larger performance gap
    - Varies from 68.90% to 71.25%
  - The **VGG families** perform better than other structures

P Net	R: mIoU (%)	T: mIoU (%)
Non-VGG families		
GoogleNet [25]	68.91	68.90
AlexNet [15]	69.80	69.87
MobileNetV2 [22]	69.98	70.01
ResNet18 [11]	70.16	70.14
VGG families		
VGG16 [23]	70.68	70.71
VGG19 [23]	**71.25**	71.19

- Baseline PSPNet [6] w/ ResNet18 backbone w/o perceptual loss: 69.60% of mIoU
- R: weights of the perceptual loss network are randomly initialized
- T: ImageNet pre-trained weights

Percep. network	Structure	R: mIoU (%)	T: mIoU (%)
N/A	1, 1, 1, 1, 1	70.88 ± 0.03	N/A
VGG11	1, 1, 2, 2, 2	70.18 ± 0.11	70.21 ± 0.10
VGG13	2, 2, 2, 2, 2	70.64 ± 0.14	70.62 ± 0.12
VGG16	2, 2, 3, 3, 3	70.68 ± 0.03	70.71 ± 0.02
VGG19	2, 2, 4, 4, 4	71.25 ± 0.04	71.19 ± 0.07
N/A	3, 3, 4, 4, 4	70.89 ± 0.23	N/A

- Variants of VGG
- Kernel size
  - With a larger receptive field, the perceptual loss works slightly better

Kernel size	mIoU (%)
1	70.71 ± 0.020
3	70.88 ± 0.025
5	70.93 ± 0.191
7	71.16 ± 0.122

Without pretraining

: Dense prediction results

- Semantic segmentation
  - The perceptual loss consistently improves the baseline
    - Across different datasets with different network structures

Network	Param.	Percep.	mIoU
Cityscapes			
PSPRes18 [36]	22.9M		69.6%
PSPRes18 [36]	22.9M	✓	71.2% (↑1.6%)
HRNetw18s [32]	3.76M		73.61%
HRNetw18s [32]	3.76M	✓	74.17% (↑0.56%)
DeepLabV3+ [3]	39.3M		80.09 %
DeepLabV3+ [3]	39.3M	✓	80.70% (↑0.61%)
ADE20K			
PSPRes18 [36]	23.0M		33.8%
PSPRes18 [36]	23.0M	✓	34.2% (↑0.4%)
HRNetw18s [32]	3.79M		31.38%
HRNetw18s [32]	3.79M	✓	32.26% (↑0.88%)
DeepLabV3+ [3]	39.4M		42.72%
DeepLabV3+ [3]	39.4M	✓	42.95% (↑0.23%)
PascalVOC			
PSPRes18 [36]	22.9M		49.1%
PSPRes18 [36]	22.9M	✓	50.31% (↑1.21%)
HRNetw18s [32]	3.76M		65.20%
HRNetw18s [32]	3.76M	✓	65.41% (↑0.21%)
DeepLabV3+ [3]	39.3M		75.93%
DeepLabV3+ [3]	39.3M	✓	76.79% (↑0.86%)

- Depth estimation

Methods	WCE [1]	VNL [34]	Percep.	Rel. (%)
a	✓			14.5
b	✓		✓	14.0
c	✓	✓		13.6
d	✓	✓	✓	**13.2**

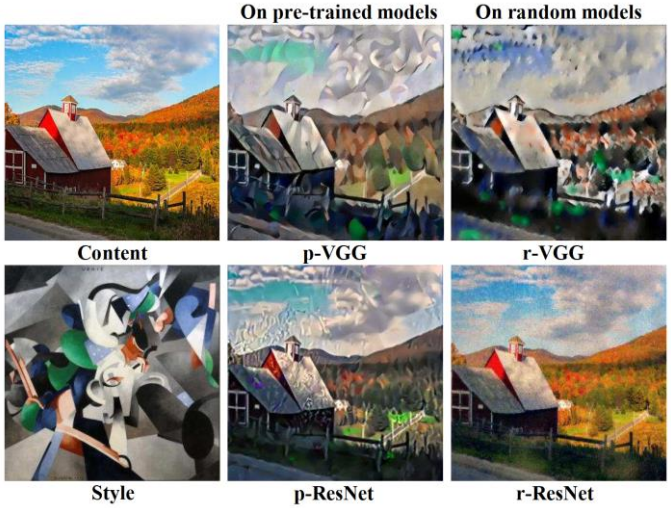
- Instance segmentation
  - Both the detection results (Box AP (%)) and the segmentation results (Mask AP (%)) are improved

Method	Percep.	Box			Mask		
		AP	AP_{50}	AP_{75}	AP	AP_{50}	AP_{75}
CondInst [26]		36.91	55.29	39.94	33.42	53.00	35.56
CondInst [26]	✓	**37.44**	**55.69**	**40.51**	**33.69**	**53.33**	**35.82**

Without pretraining

: Rethinking and Improving the Robustness of Image Style Transfer (CVPR 2021) [7]

- Previous assumptions
  - Stylization quality degrades when using ResNets as a perceptual loss network
- Findings
  - **Residual connections** produce feature maps of **small entropy** not suitable for style transfer
  - Propose a solution to improve the robustness of the ResNet architecture-based style transfer
    - Based on a softmax transformation of the feature activations that enhances their entropy
  - *Experimental results demonstrate that this small magic can greatly improve the quality of stylization results, even for networks with **random weights**.*
    - p-, r-: pre-trained, random initialization
  - *This suggests that **the architecture** used for feature extraction **is more important than** the use of **learned weights** for the task of style transfer.*



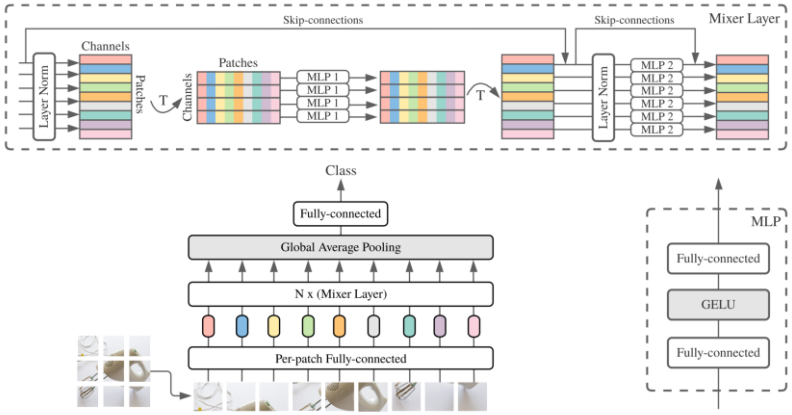
Conclusion

Conclusion

: Necessary Ingredients of a Model

- Without convolution weights
  - The expressive power of the affine parameters γ and β (and random features)
- Without pretraining
  - The structure of a deep network itself maybe sufficient
- Without convolution & attentions
  - **Convolutions, attentions** …
    - MLP-Mixer: An all-MLP Architecture for Vision (Google Brain 2021) [8]

*While **convolutions** and **attention** are both sufficient for good performance, **neither of them are necessary**.*



Reference

- [1] Mitchell, T. M. (1980), The need for biases in learning generalizations
- [2] Frankle, J., Schwab, D., & Morcos, A.S. (2021). Training BatchNorm and Only BatchNorm: On the Expressive Power of Random Features in CNNs. 2021 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
- [3] Liu, Y., Chen, H., Chen, Y., Yin, W., & Shen, C. (2021). Generic Perceptual Loss for Modeling Structured Output Dependencies. 2021 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
- [4] Ledig, C., Theis, L., Huszár, F., Caballero, J., Aitken, A.P., Tejani, A., Totz, J., Wang, Z., & Shi, W. (2017). Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 105-114.
- [5] Johnson, J., Alahi, A., & Fei-Fei, L. (2016). Perceptual Losses for Real-Time Style Transfer and Super-Resolution. ECCV 2016
- [6] Zhao, H., Shi, J., Qi, X., Wang, X., & Jia, J. (2017). Pyramid Scene Parsing Network. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
- [7] Wang, P., Li, Y., & Vasconcelos, N. (2021). Rethinking and Improving the Robustness of Image Style Transfer. 2021 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)
- [8] Tolstikhin, I., Houlsby, N., Kolesnikov, A., Beyer, L., Zhai, X., Unterthiner, T., Yung, J., Keysers, D., Uszkoreit, J., Lucic, M., & Dosovitskiy, A. (2021). MLP-Mixer: An all-MLP Architecture for Vision. ArXiv, abs/2105.01601.