WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Densely Connected Neural Networks

CS 839 - Special Topics in AI: Deep Learning

Team:
Diwanshu Jain
Shri Shruthi Shridhar
Sacha Jungerman

September 10, 2020

Overview

1. Motivation for ResNet

2. ResNet Overview

3. DenseNet
    a. Motivation
    b. Key Idea
    c. Architecture Overview and Features
    d. Experiments and Results
    e. Advantages and Discussion
    f. Future Work

1.

Motivation for ResNet

Why stacking layers is a problem?

Is learning better networks as easy as stacking more layers?

- Problem 1: Vanishing/ Exploding gradients
    - Large derivatives → increase exponentially → ‘Explodes’
    - Small derivatives → decrease exponentially → ‘Vanishes’

Why is it bad?

- Exploding Gradients:
  - Unstable
  - Incapable of effective learning
- Vanishing Gradients:
  - Incapable of effective learning

Solution?

➢ Reducing amount of Layers

➢ Weight Initialization
  - Check out this article for different kinds of initialization!

➢ ResNet
  - More on this later.

Is learning better networks as easy as stacking more layers?

- Problem 2: Degradation Problem in deeper networks

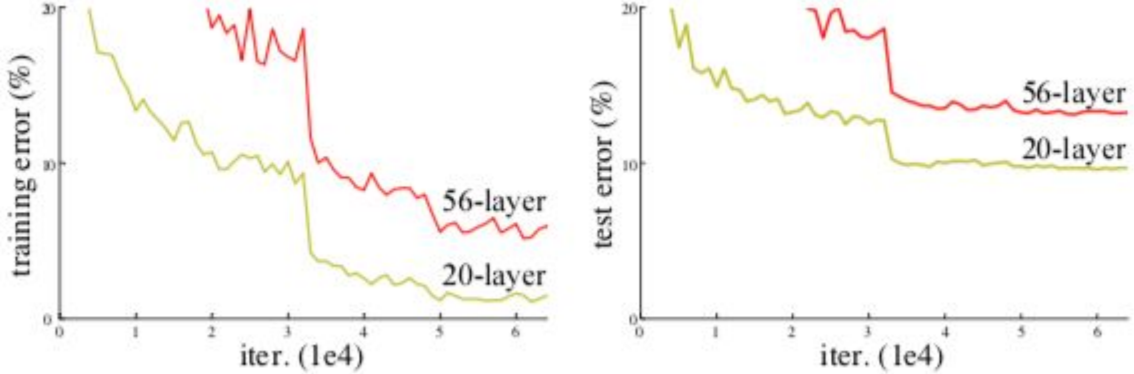


Fig 1. Training error (left) & test error (right) on CIFAR-10 with 20-layers and 56-layer 'plain' networks. [ResNet]

Why is it counterintuitive?

- *Analogy:*
  - Data which can be learned effectively using a linear representation: $h(x) = bx + c$; (b, c - learned parameters)
  - If $h(x) = ax^2 + bx + c$ is used while training
    - Expect $a \rightarrow 0$: This is what is observed in practice.
  - DOESN'T apply to neural networks!

Reason? Solution?

- *Cause:*

  Optimization problems

- *Solution:*

  ResNet

2.

ResNet

Key Idea

- Deep Residual Learning:
  - Fitting a *residual mapping* rather than *direct mapping*

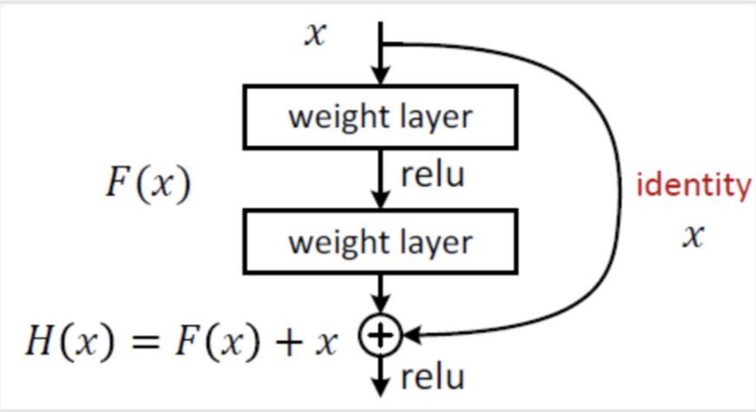


Fig 2. The operation $F + x$ is performed by a shortcut connection and element-wise addition. [ResNet]

Key Idea

➢ *Why should this be helpful?*

- Easier to optimize the residual mapping
- Difficulties in approximating identity mappings by multiple non-linear layers
- E*asier* for the solver to find the perturbations with a reference to an identity mapping

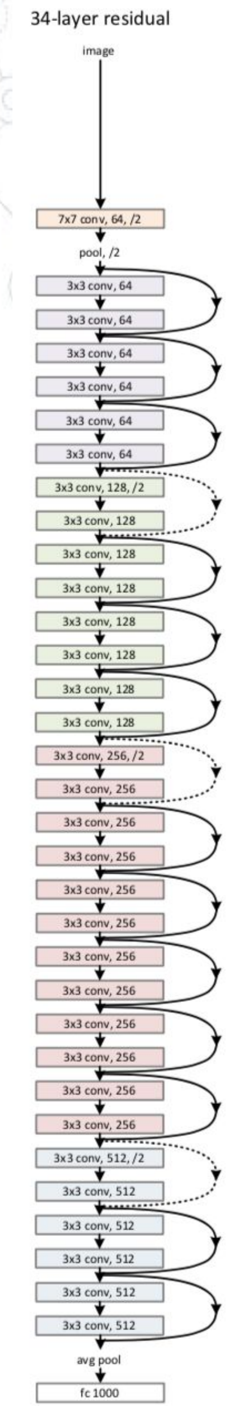
Architecture

- *Shortcut Connections*
  - $y = F(x, \{W_i\}) + x;$

    $F(x, \{W_i\})$: *Residual mapping*
  - Eg. For 2 layers,

    $F = W_2\sigma(W_1x);$ σ: *activation function*
- Dimension mismatch b/w F, x
  - Linear projection W_s

    $y = F(x, \{W_i\}) + W_sx;$



Did it solve the problems?

- *Exploding/Vanishing Gradients:*
    - Shortcut connections path allow gradient to reach those beginning nodes with greater magnitude by skipping some layers in between

Did it solve the problems?

➢ *Degradation Problem:*

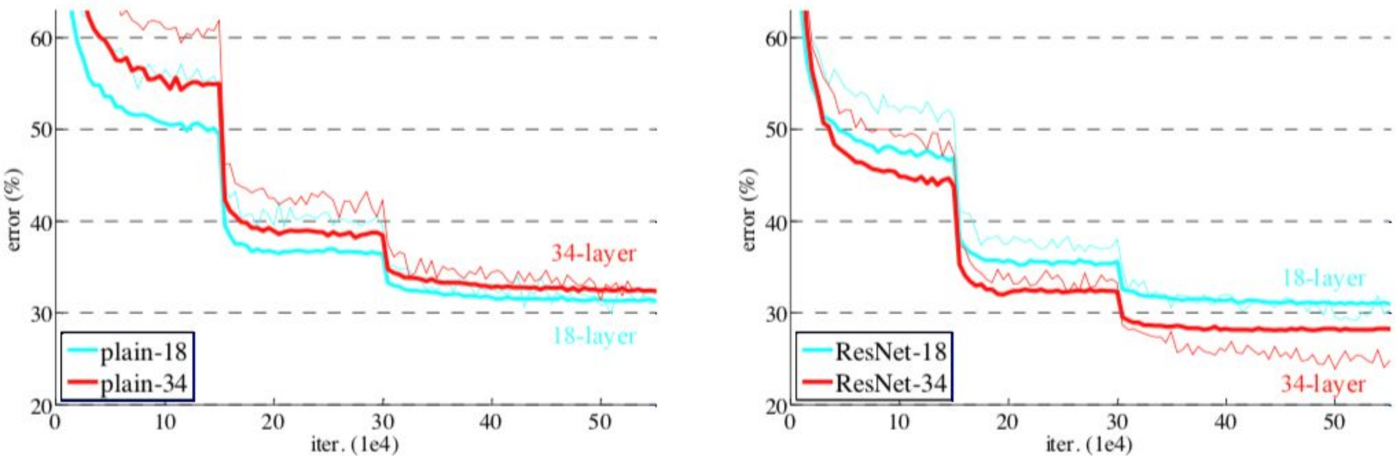


Fig 4. Training on ImageNet. Thin curves denote training error, bold curve denotes validation error. Left: Plain networks; Right: ResNets. The residual network have no extra parameter as compare to the plain counterparts. [ResNet]

Drawbacks

- The identity function and the output are combined by summation, which may impede the information flow in the network
  - Let $H_l(.)$ be a non-linear transformation, l = index of the layer; output of the l^{th} layer be x_l

$$x_l = H_l(x_{l-1}) + x_{l-1}$$

- DenseNet found to achieve more accuracy!

3.

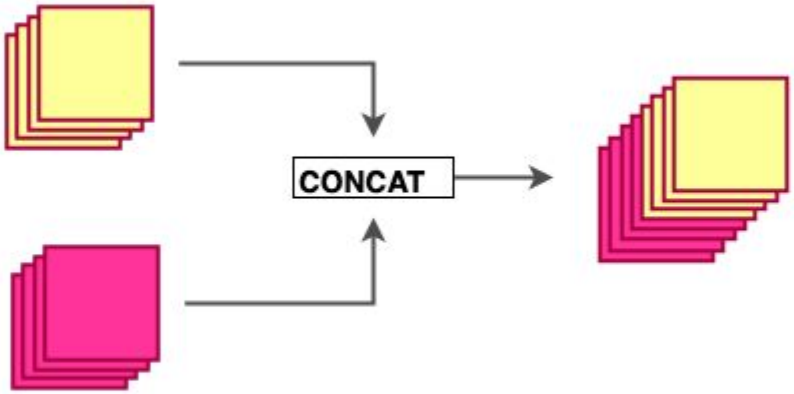
DenseNet

Motivation

- To avoid vanishing gradient:
    - Shortcut connections between layers!
- Instead of summation (+), use concatenation ©
    - Ensure maximum information flow between layers

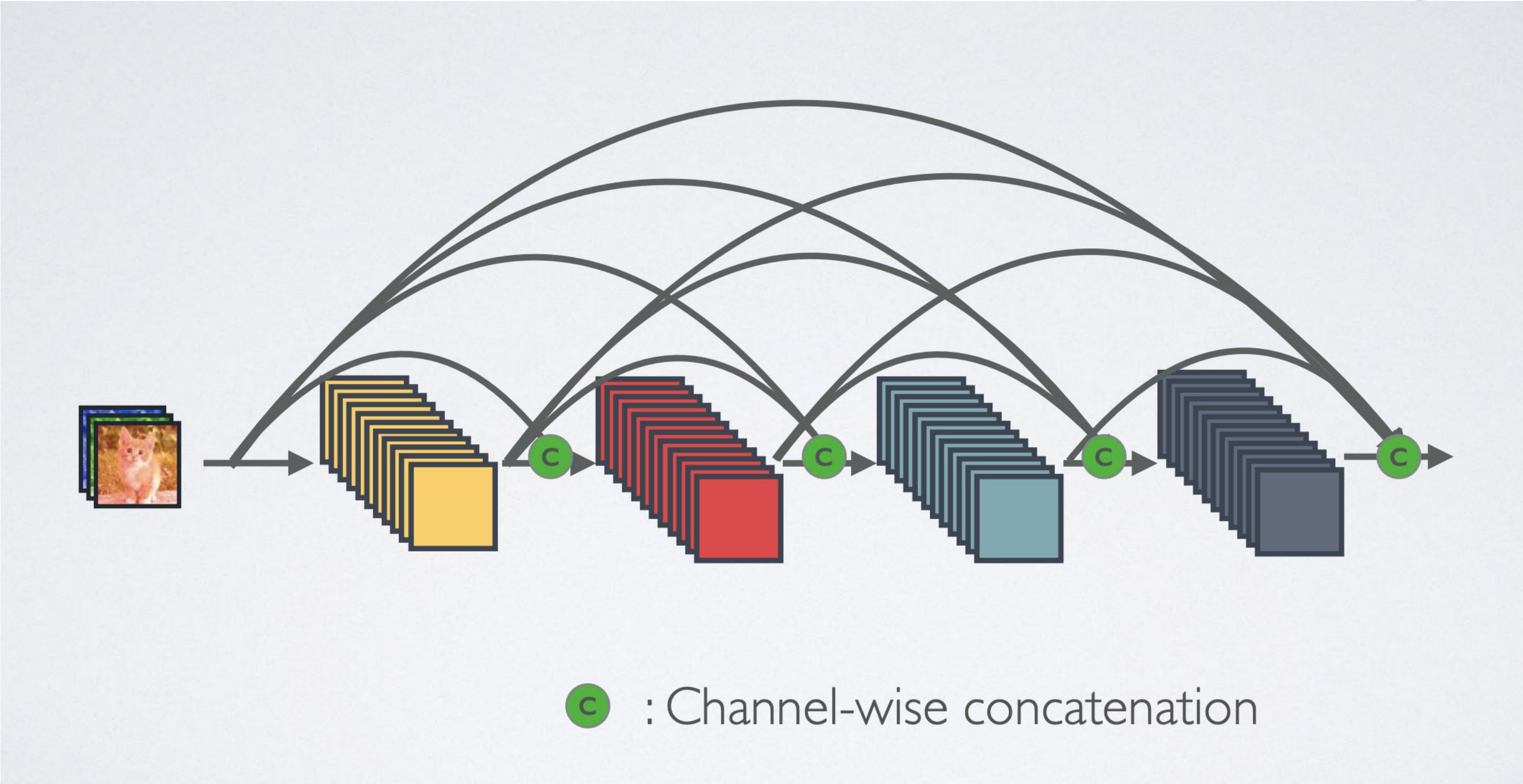
Key Idea

- Connect each layer to every subsequent layer
- Feature maps connected through concatenation



Source: [DenseNet Review Blog]

Architecture Overview



Channel-wise concatenation (Dense Connectivity)[DenseNet CVPR]

Architecture Features

- Dense Connectivity:
  - Aim: To improve information flow
  - Let $H_l(.)$ be a non-linear transformation, l = index of the layer; output of the l^{th} layer be x_l
  - ResNets: $x_l = H_l(x_{l-1}) + x_{l-1}$

    DenseNets: $x_l = H_l([x_0, x_1, x_2, \ldots, x_{l-1}])$

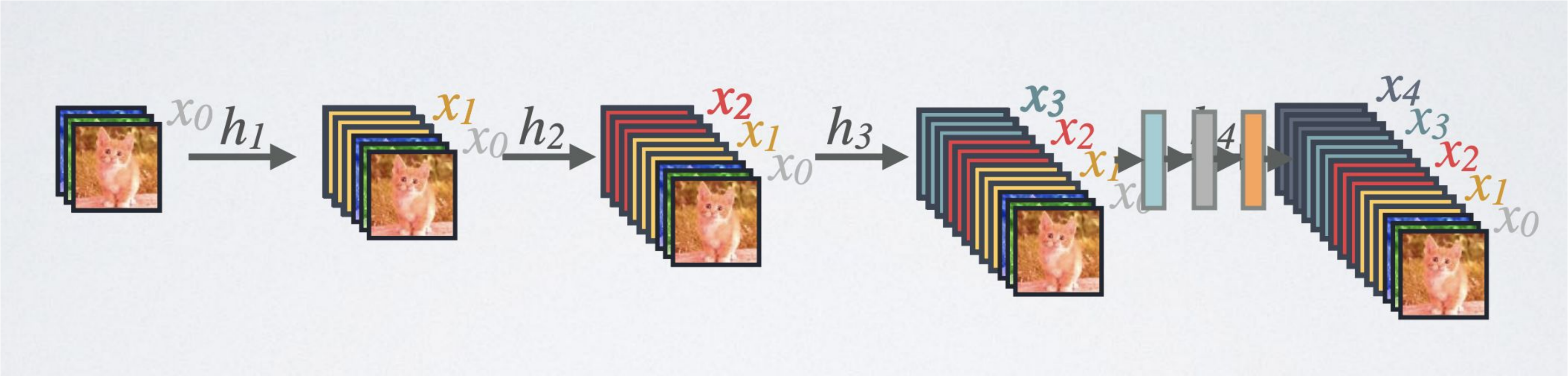
Architecture Features

- Dense Connectivity:

Forward Propagation [DenseNet CVPR]

Architecture Features

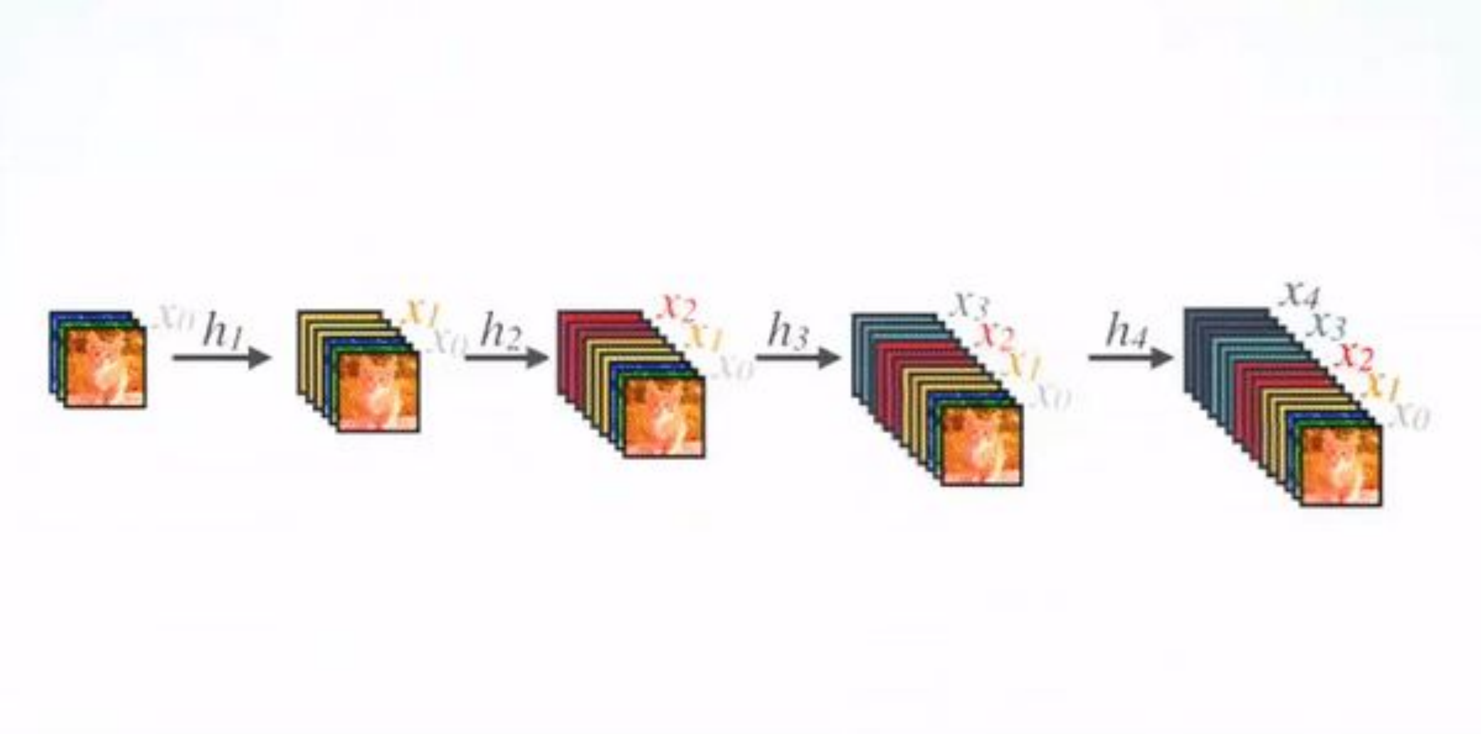
➢ Dense Connectivity:



Forward Propagation [DenseNet CVPR]

Architecture Features

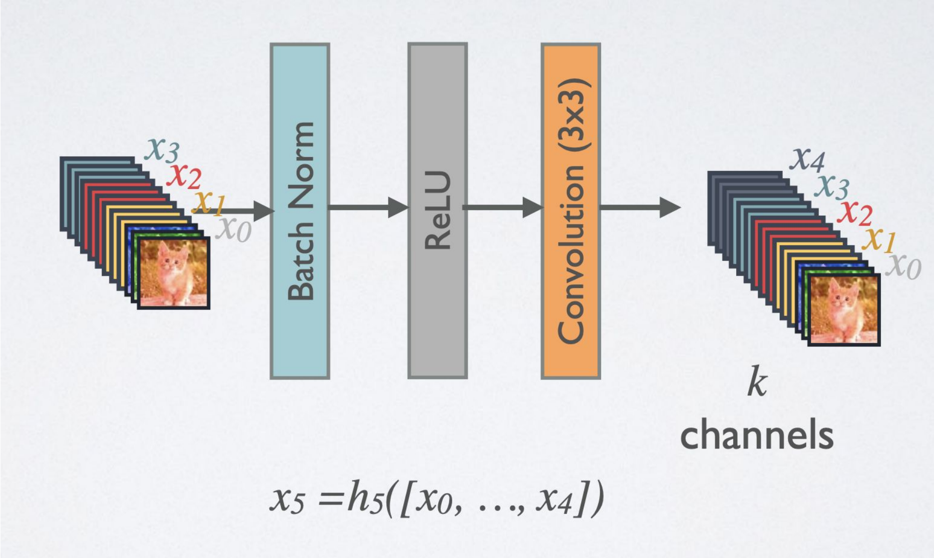
➢ Composite layer: [*pre-activation*]



Composite Layer [DenseNet CVPR]

Architecture Features

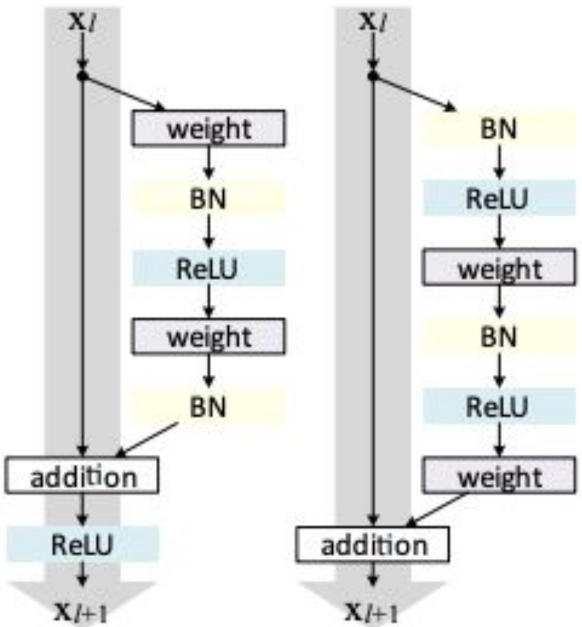
- Composite function: [*pre-activation*]
    - Ease of optimization
    - Reduce overfitting [More on this [here]]

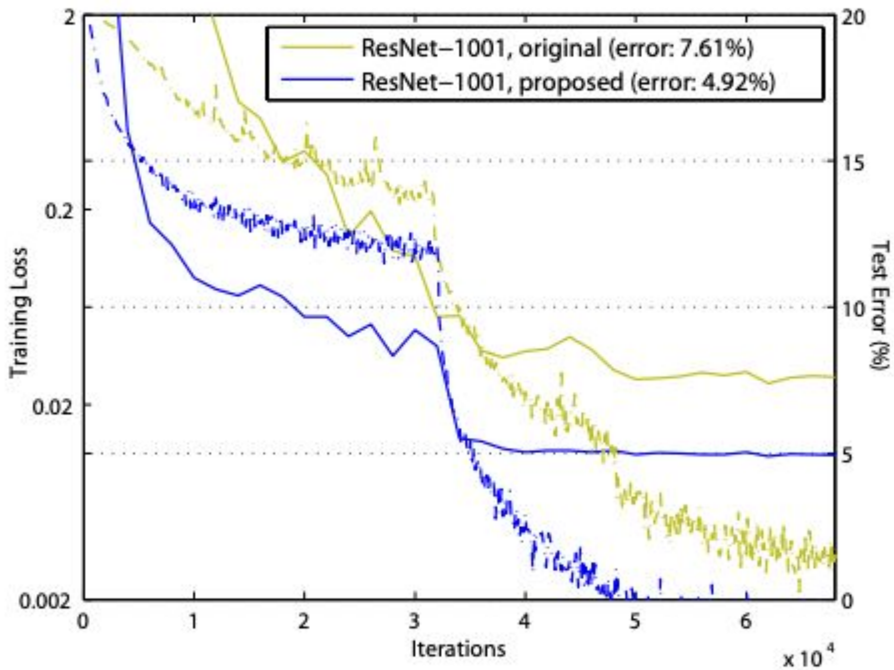


Composite Layer [DenseNet CVPR]

Architecture Features

- Composite function: [*pre-activation*]

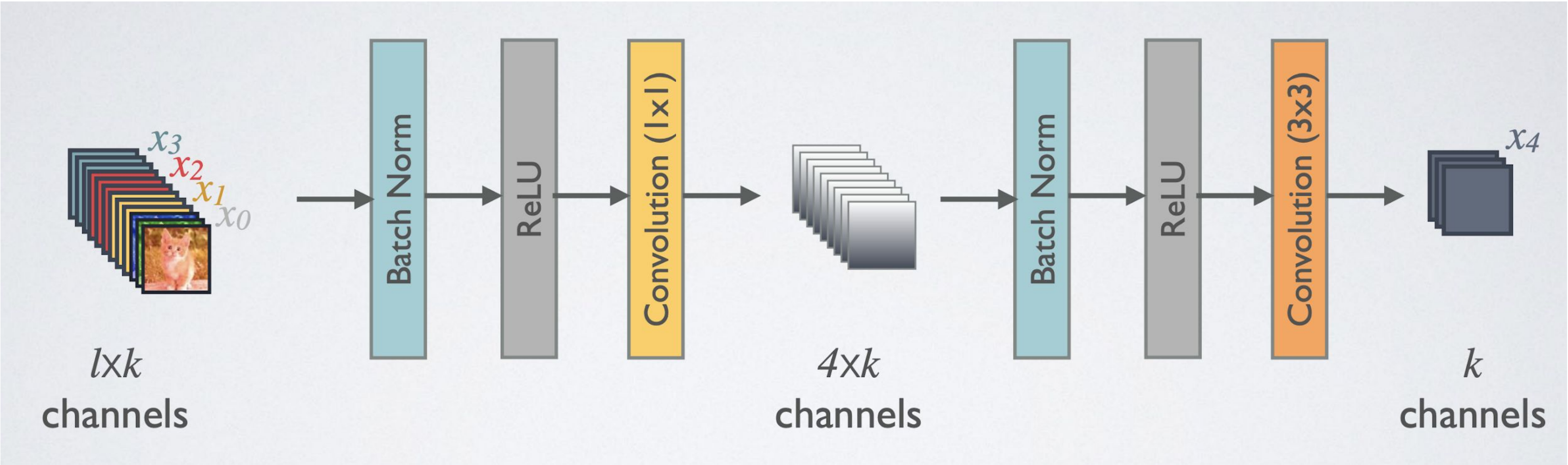




Post-activation Vs Pre-activation [paper]

Architecture Features

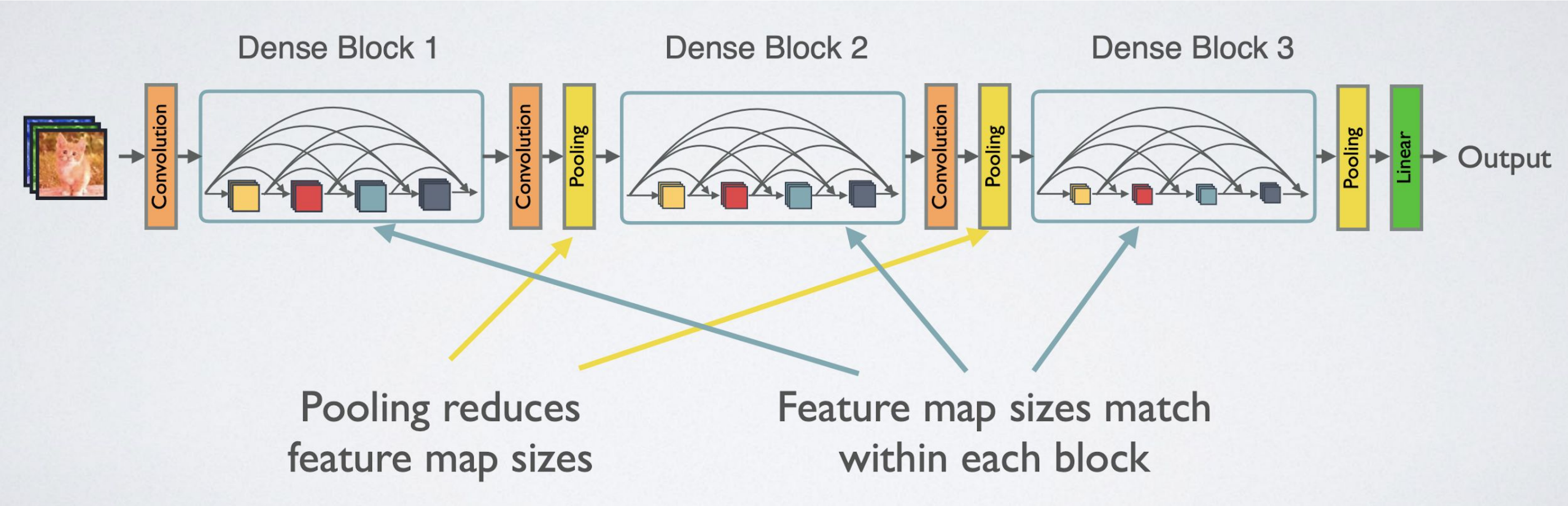
- Bottleneck layers:
  - Aim: To improve computational efficiency



Composite Layer with Bottleneck layer (DenseNet - B)[DenseNet CVPR]

Architecture Features

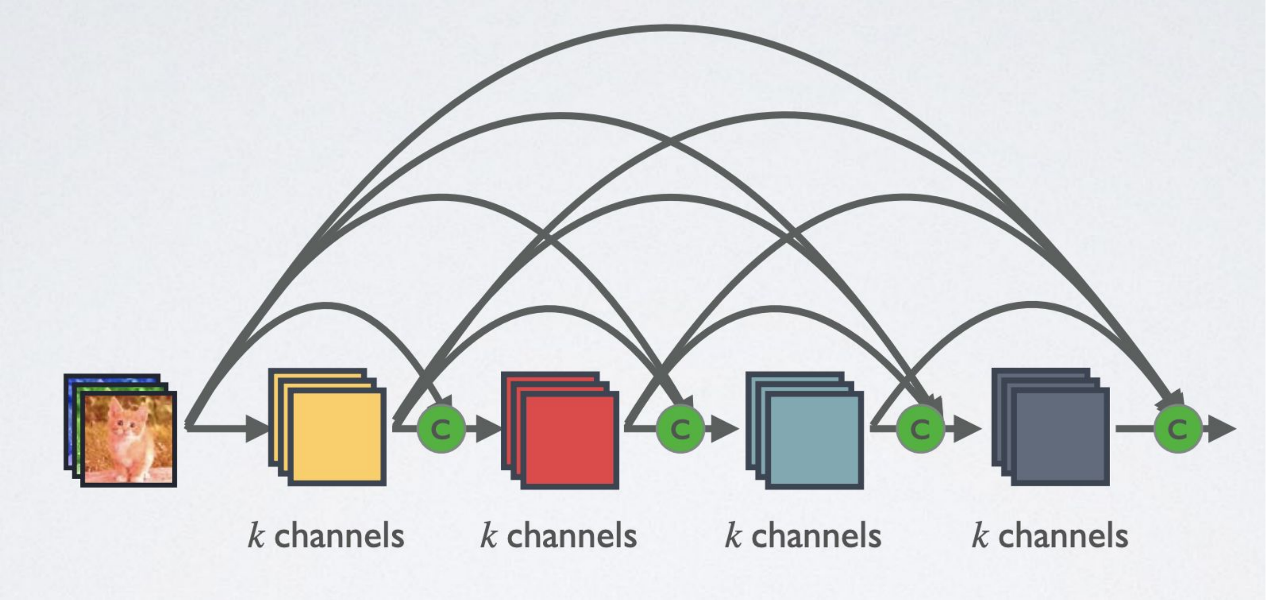
- Pooling layers:
    - Aim: Consistent feature-map sizes



Transition layer: Pooling + Convolution [DenseNet CVPR]

Architecture Features

- Growth rate (k):
  - Each function H_l produces k feature maps.



Dense & Slim**:** "Collective Knowledge"[DenseNet CVPR]

Architecture Features

- Compression:
  - Aim: Compactness → Reduce #feature-maps (*m*) at transition layer.
  - θ: *compression factor* $(0 < \theta \leq 1)$
  - Referred to as *DenseNet - C.*

Architecture Details

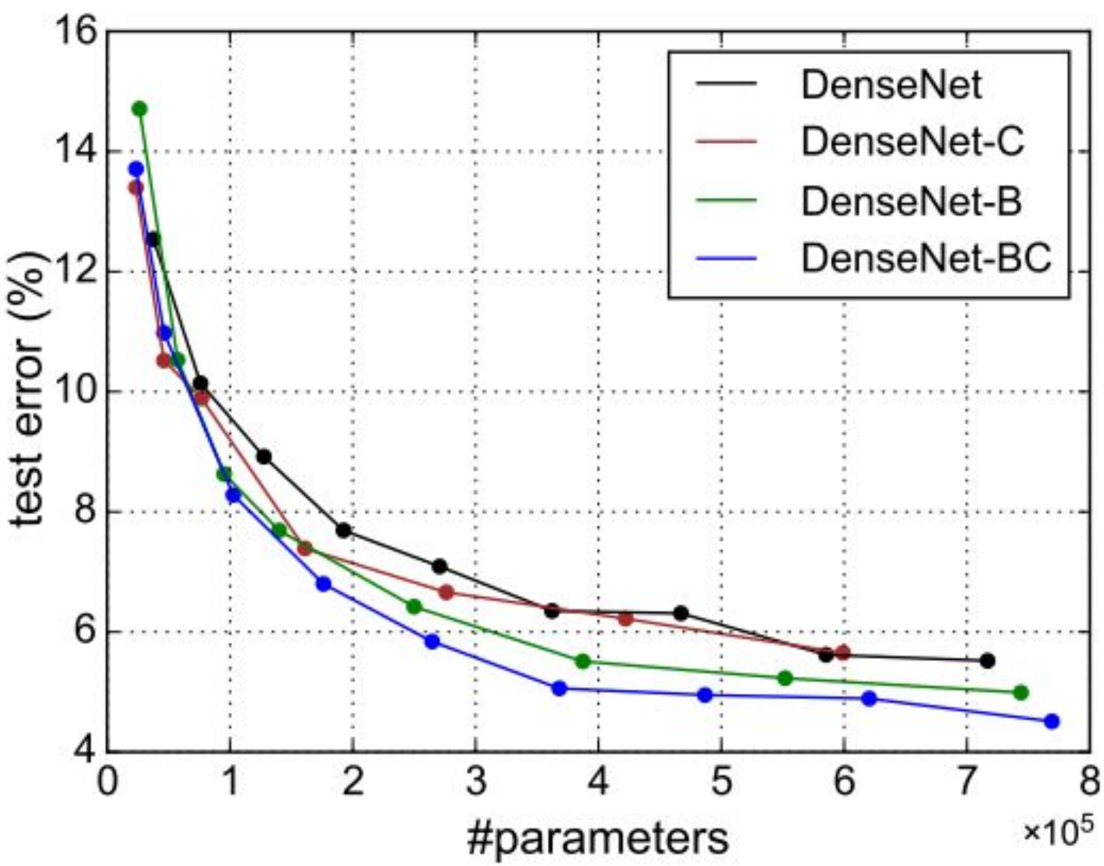
Layers	Output Size	DenseNet-121	DenseNet-169	DenseNet-201	DenseNet-264
Convolution	112×112	7×7 conv, stride 2			
Pooling	56×56	3×3 max pool, stride 2			
Dense Block (1)	56×56	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 6$
Transition Layer (1)	56×56	1×1 conv			
	28×28	2×2 average pool, stride 2			
Dense Block (2)	28×28	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 12$
Transition Layer (2)	28×28	1×1 conv			
	14×14	2×2 average pool, stride 2			
Dense Block (3)	14×14	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 24$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 64$
Transition Layer (3)	14×14	1×1 conv			
	7×7	2×2 average pool, stride 2			
Dense Block (4)	7×7	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 16$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 32$	$\begin{bmatrix} 1 \times 1 \text{ conv} \\ 3 \times 3 \text{ conv} \end{bmatrix} \times 48$
Classification Layer	1×1	7×7 global average pool			
		1000D fully-connected, softmax			

DenseNet architecture for ImageNet. Growth rate ($k = 32$). Each “conv” layer shown corresponds to BN-ReLU-Conv [DenseNet Paper]

Experiments

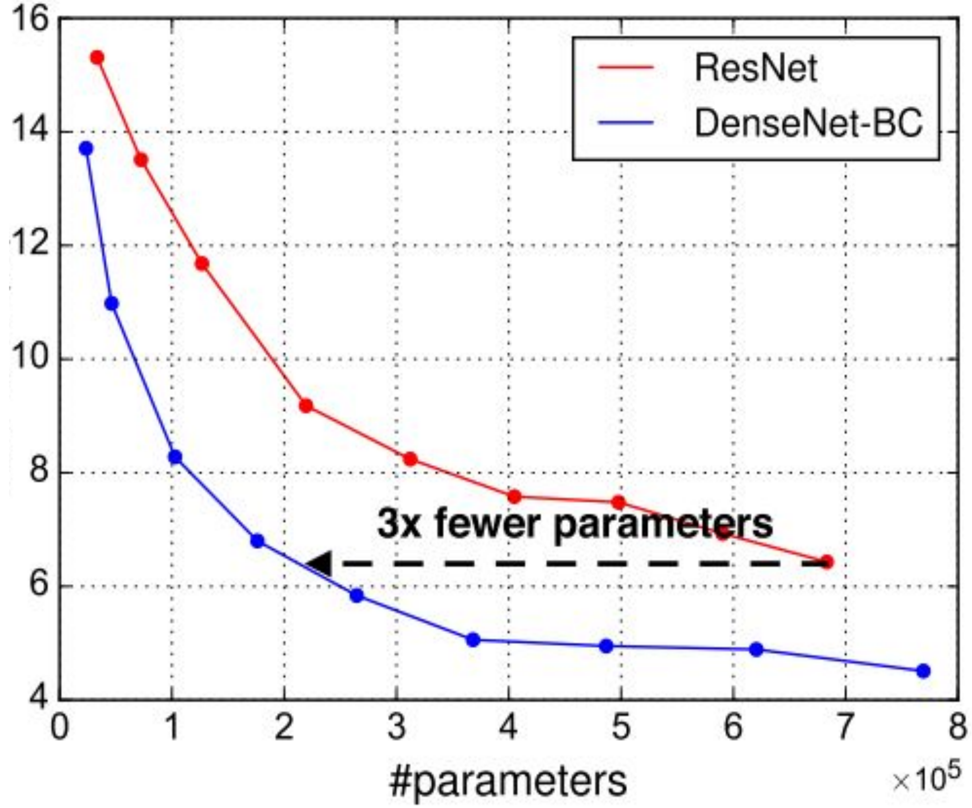
- Datasets:
  - CIFAR
  - SVHN (Street View House Numbers)
  - ImageNet
- Trained using SGD (Stochastic Gradient Descent)
- Three-step learning rate decay by 10%
- Weight decay of 0.0001 and momentum of 0.9

Results:



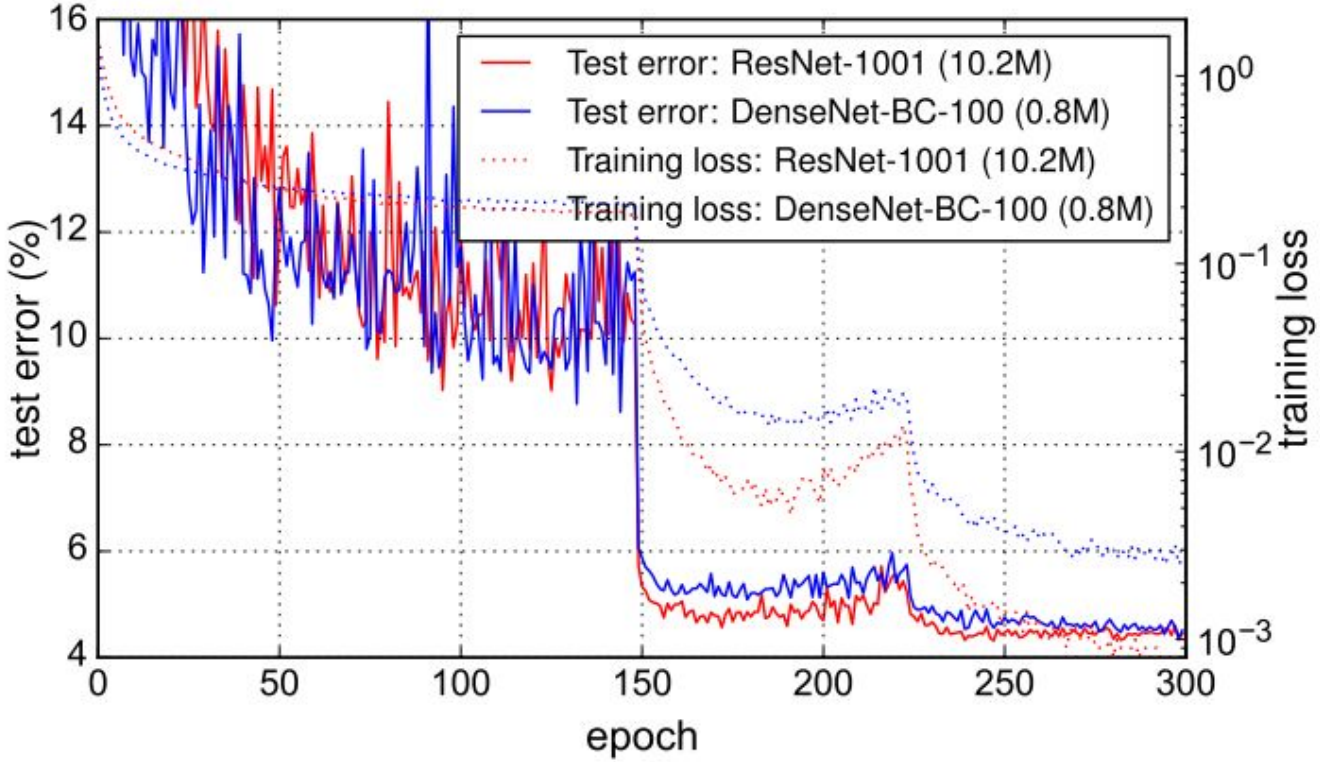
Comparison of the parameter efficiency on C10+ between DenseNet variations [DenseNet paper]

Results:



Comparison of the parameter efficiency between
DenseNet-BC and (pre-activation) ResNets [DenseNet paper]

Results:



Training and testing curves of ResNet and DenseNet [DenseNet paper]

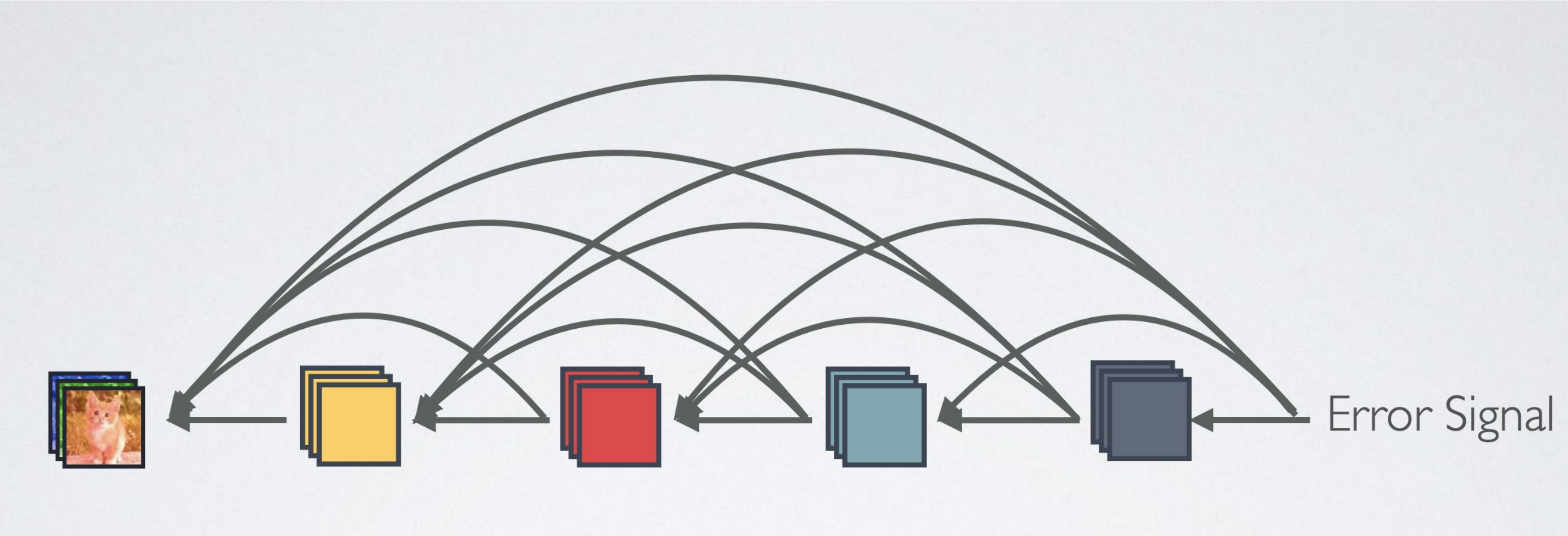
Results:

Method	Depth	Params	C10	C10+	C100	C100+	SVHN
Network in Network [22]	-	-	10.41	8.81	35.68	-	2.35
All-CNN [32]	-	-	9.08	7.25	-	33.71	-
Deeply Supervised Net [20]	-	-	9.69	7.97	-	34.57	1.92
Highway Network [34]	-	-	-	7.72	-	32.39	-
FractalNet [17]	21	38.6M	10.18	5.22	35.34	23.30	2.01
with Dropout/Drop-path	21	38.6M	7.33	4.60	28.20	23.73	1.87
ResNet [11]	110	1.7M	-	6.61	-	-	-
ResNet (reported by [13])	110	1.7M	13.63	6.41	44.74	27.22	2.01
ResNet with Stochastic Depth [13]	110	1.7M	11.66	5.23	37.80	24.58	1.75
	1202	10.2M	-	4.91	-	-	-
Wide ResNet [42]	16	11.0M	-	4.81	-	22.07	-
	28	36.5M	-	4.17	-	20.50	-
with Dropout	16	2.7M	-	-	-	-	1.64
ResNet (pre-activation) [12]	164	1.7M	11.26*	5.46	35.58*	24.33	-
	1001	10.2M	10.56*	4.62	33.47*	22.71	-
DenseNet ($k = 12$)	40	1.0M	**7.00**	5.24	**27.55**	24.42	1.79
DenseNet ($k = 12$)	100	7.0M	**5.77**	**4.10**	**23.79**	**20.20**	1.67
DenseNet ($k = 24$)	100	27.2M	**5.83**	**3.74**	**23.42**	**19.25**	**1.59**
DenseNet-BC ($k = 12$)	100	0.8M	**5.92**	4.51	**24.15**	22.27	1.76
DenseNet-BC ($k = 24$)	250	15.3M	**5.19**	**3.62**	**19.64**	**17.60**	1.74
DenseNet-BC ($k = 40$)	190	25.6M	-	**3.46**	-	**17.18**	-

Error rates of different models on CIFAR and SVHN datasets with other details [DenseNet paper]

Advantages

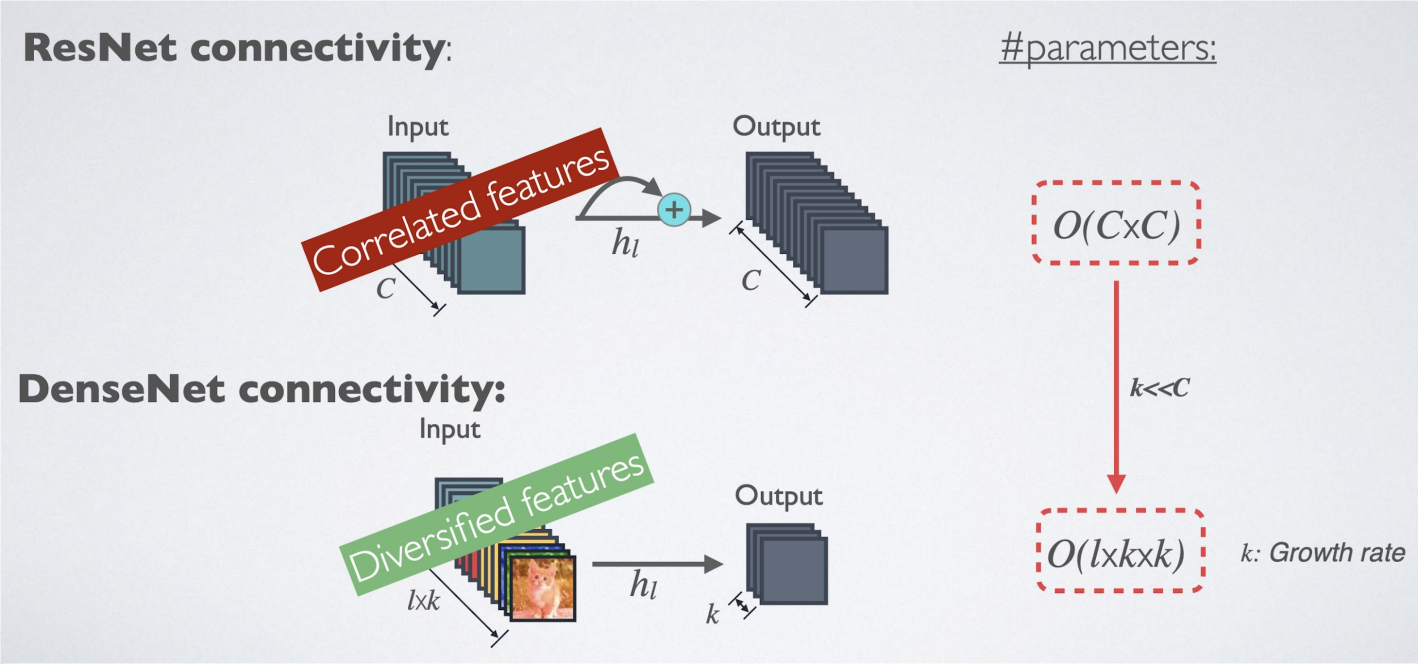
➢ Strong Gradient Flow



Implicit “Deep Supervision” [DenseNet CVPR]

Advantages

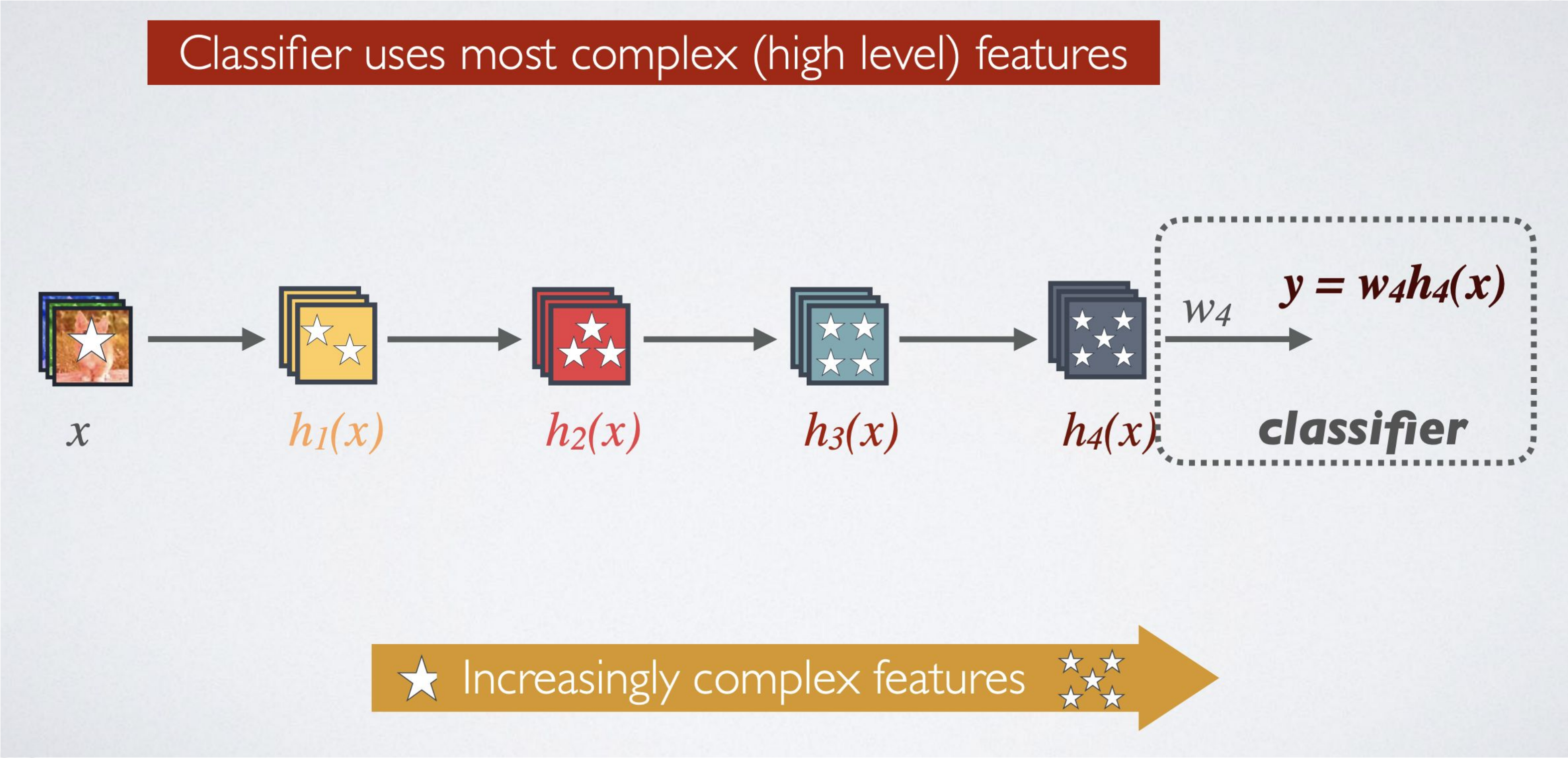
➢ Parameter and computational efficiency; diversified features



Parameters comparison in ResNet Vs DenseNet [DenseNet CVPR]

Advantages

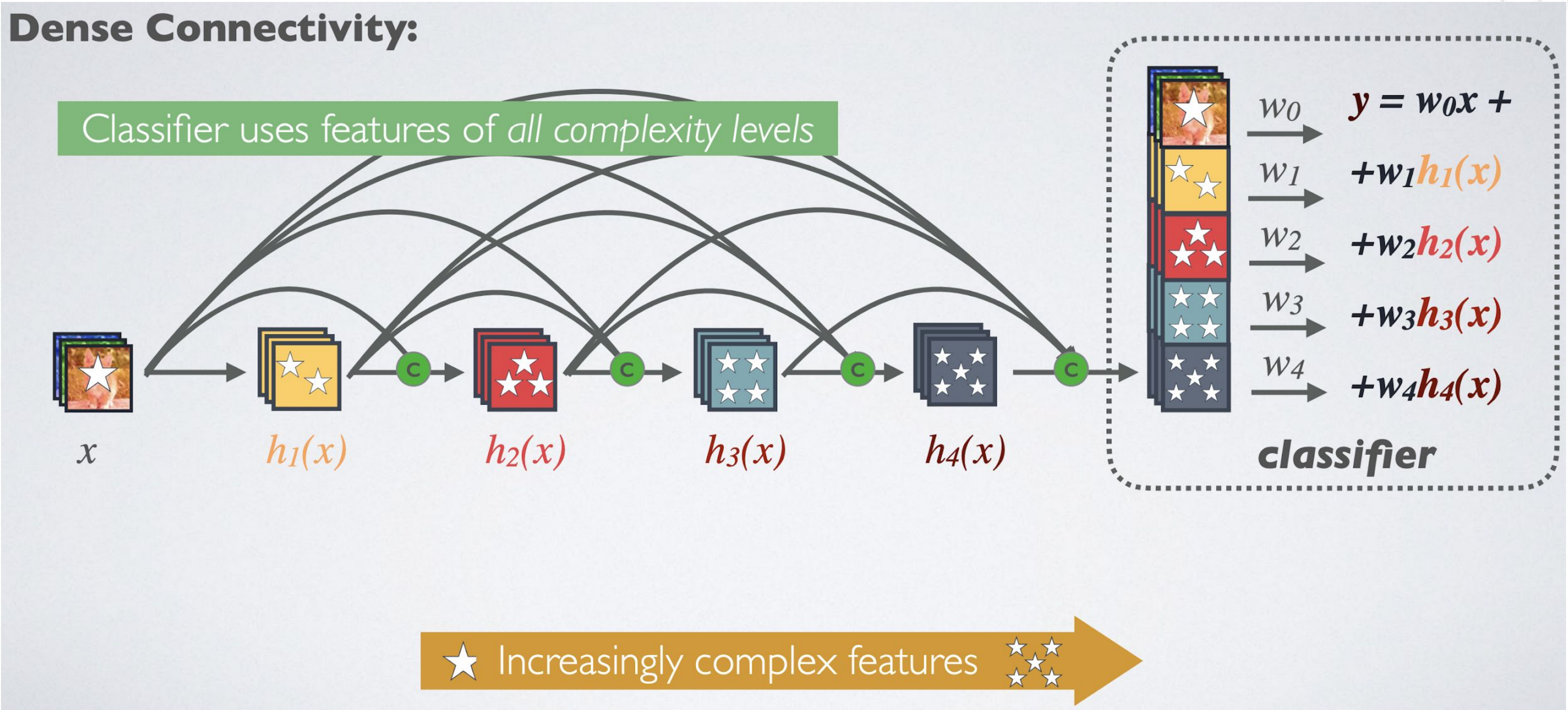
➢ Maintains low complexity features



Standard Connectivity [DenseNet CVPR]

Advantages

➢ Maintains low complexity features

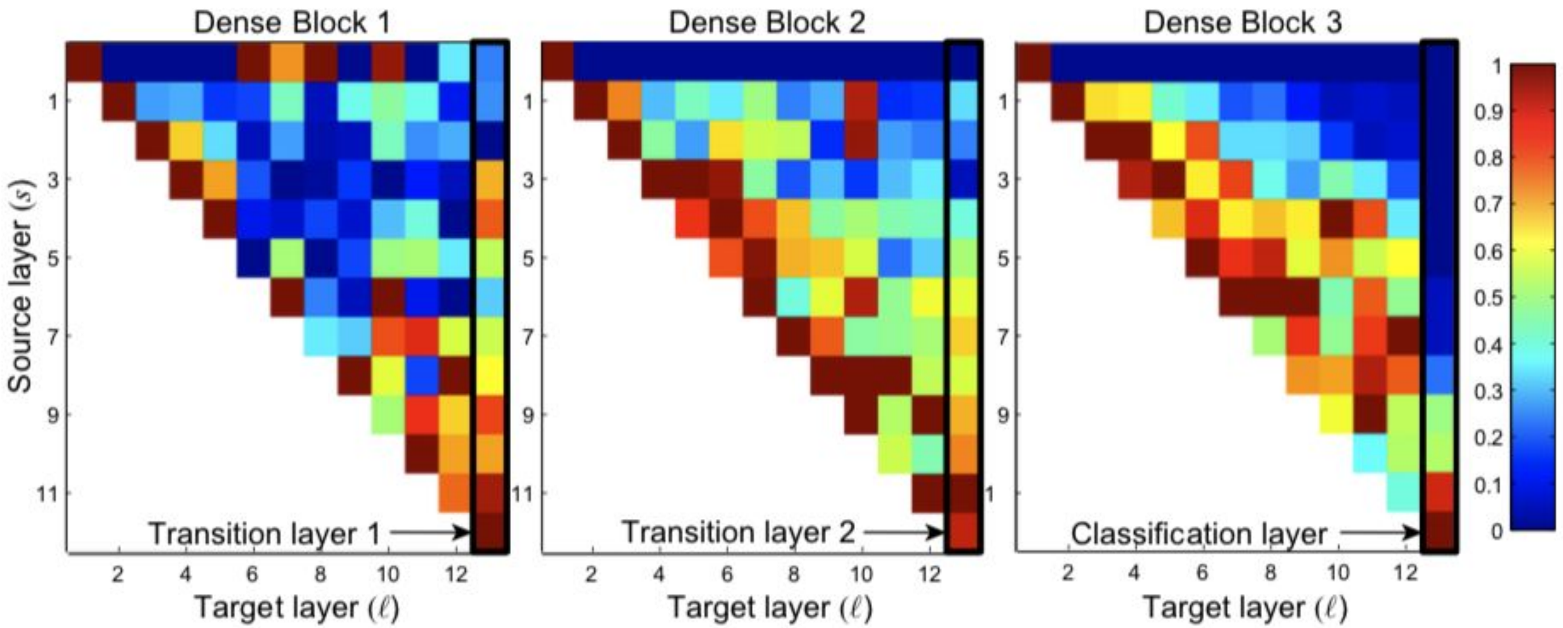


Dense Connectivity [DenseNet CVPR]

Discussion

- Model Compactness
- Implicit Deep Supervision
- Stochastic Vs Deterministic Connection
- Feature Reuse

Analysis on Feature Reuse



Heat map on the average absolute weights of how Target layer (l) reuses the source layer (s) [DenseNet paper]

Breakout Group Discussion!

Any limitations that you can think of?

Future Work

- Computational and parameter-efficiency can be improved

- SparseNet: A Sparse DenseNet for Image Classification paper addresses this and proposes sparsity as a solution to address this

References

- Huang, G., Liu, Z., van der Maaten, L. & Weinberger, K. Q. (2016). Densely Connected Convolutional Networks (cite arxiv:1608.06993Comment: CVPR 2017)
- He, K., Zhang, X., Ren, S. & Sun, J. (2015). Deep Residual Learning for Image Recognition (cite arxiv:1512.03385Comment: Tech report)
- https://towardsdatascience.com/review-densenet-image-classification-b6631a8ef803
- https://towardsdatascience.com/paper-review-densenet-densely-connected-convolutional-networks-acf9065dfefb

Any questions?

Thank you!