BMVC 2022

LAYER FOLDING: NEURAL NETWORK DEPTH REDUCTION USING ACTIVATION LINEARIZATION

AMIR BEN DROR, NIV ZEHNGUT, AVRAHAM RAVIV, EVGENY ARTYOMOV, RAN VITEK, ROY JEVNISEK

SAMSUNG ISRAEL RESEARCH CENTER

24.03.07
Juyun Wee
EffL@POSTECH

NETWORK SIMPLIFICATION: WHY IT MATTERS

Despite the increasing prevalence of deep neural networks, their applicability in resource-constrained devices is limited due to their computational load. Real-time latency largely depends on the network's depth

DILEMMA OF DEPTH

DEEPER

Real-time latency problem

SHALLOWER

width of networks must grow exponentially

MINIMUM DEPTH

certain depth required to preserve performance on a given task,

however, many architectures are typically deeper than that! WHY?

BACKGOUND

ROLE OF THE ADDED LAYERS

1. CONVERGENCE ACCELERATION

- Act as **preconditioners** to speed up optimal solution finding
- Enhance feature representation refinement over iterations, improving the efficiency of gradient descent and other optimization algorithms.

2. ACCURACY ENHANCEMENT

- Incremental improvement in prediction accuracy through iterative refinement
- Improved capability in capturing data intricacies, boosting task performance

Hence, some layers can be regarded as crucial for deepening complex feature while others for refining optimization efficiency

MOTIVATION

EDNL

Effective Degree of Non-Linearity (EDNL)

Identifies minimal depth for optimal functionality, based on non-linear layers' count.
Reduction up to this level may exhibit no impact and yet considerably improve network’s efficiency.

For Network efficiency, many architectures have layers exceeding EDNL.

Advantages of Shallower Networks

Hardware Compatibility: Shallower networks enhance performance on devices by reducing inter-layer computational overhead.

⟶ **Activation Layer Optimization!**

MOTIVATION

OPTIMIZING PRE-TRAINED NETWORK

Optimization Perspective

1. Efficient Fine-tuning:

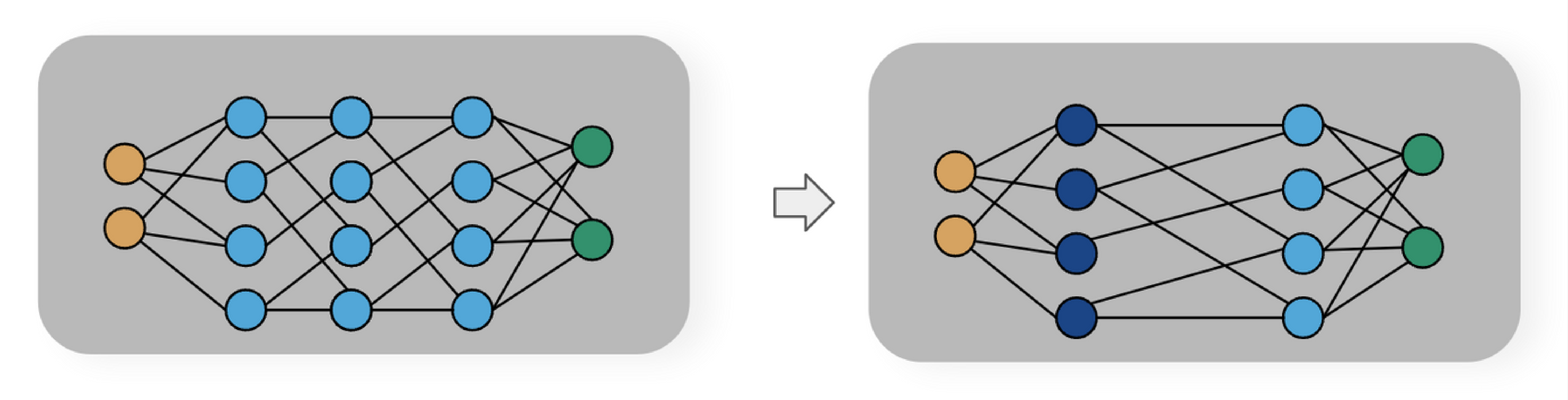
Optimizing pre-trained models consumes significantly less computational resources than training a new model.

2. Leveraging Pre-trained Depths:

Fine-tuning shallower networks from deeper counterparts utilizes learned representations and local minima, enhancing efficiency

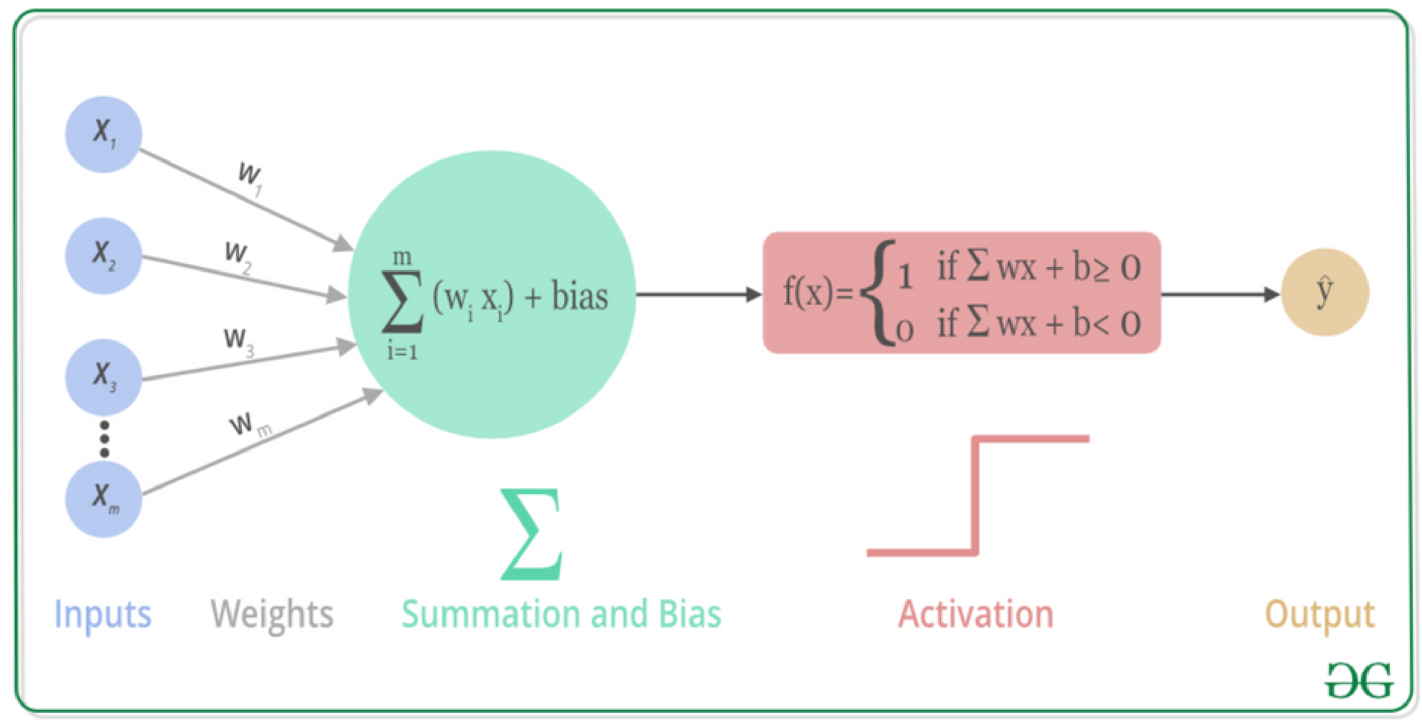
OVERVIEW

LAYER FOLDING



OBJECTIVE

reduce the network's depth (number of layers)



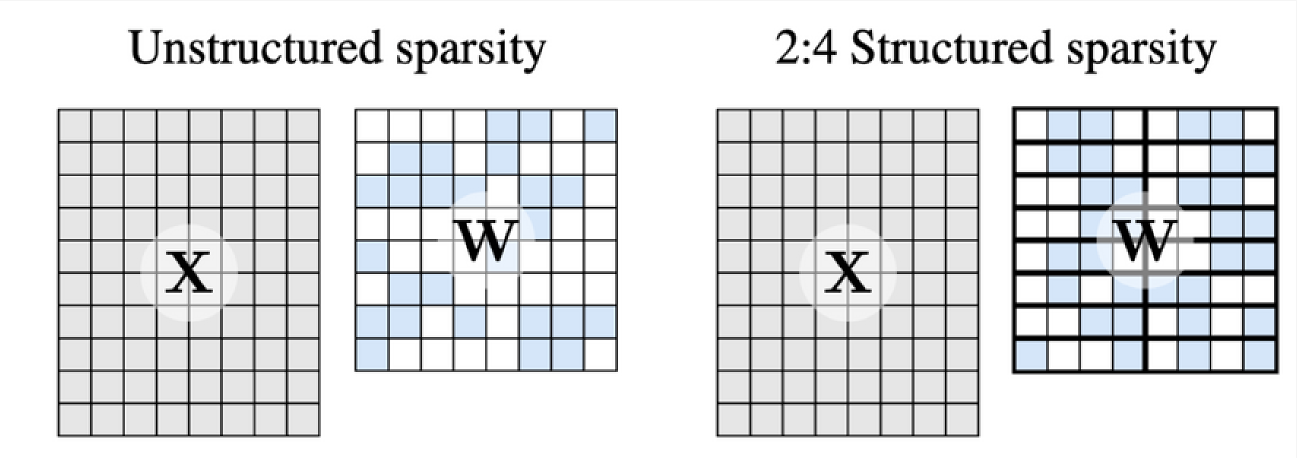
This paper propose to learn <u>which activations can be removed</u> without incurring a significant accuracy degradation. This allows us to merge adjacent linear layers, and in turn, transform deep networks into shallow ones

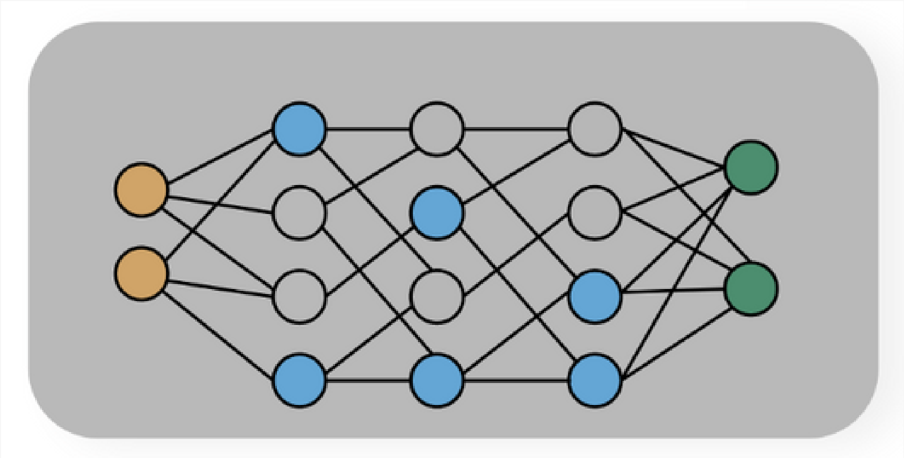
OVERVIEW

PRUNING VS LAYER FOLDING

They both optimize network by removing layer or reducing width

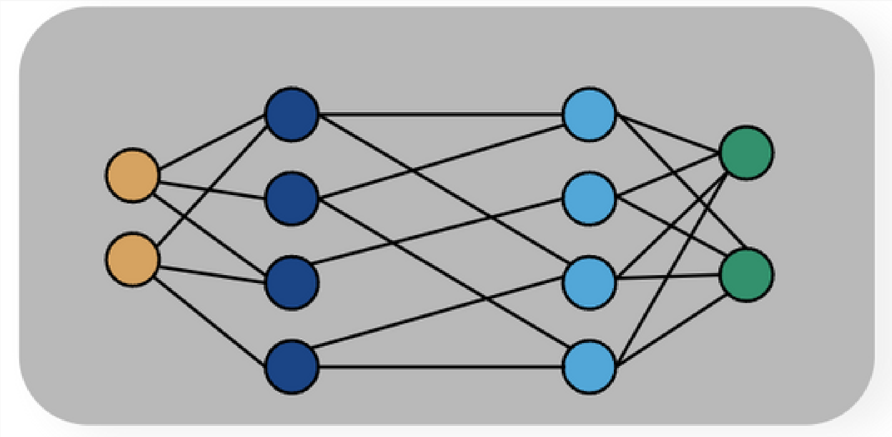
PRUNING





Reduce layer size by adopting some layers while allowing compensation during fine-tuning, which force the network to adopt a new intermediate representation

LAYER FOLDING



Maintain representation while using foundational-preserving transformation

CONTRIBUTIONS

1. Innovative Depth Reduction

Introduced Layer Folding to merge consecutive linear layers by removing non-linear activations, optimizing while preserving the network's learned features.

2. Establishing EDNL

Defined the Effective Degree of Non-Linearity (EDNL) to determine the minimal functional depth of networks, highlighting its dependence on task complexity over original depth.

3. Enhanced Mobile Network Performance

Applied Layer Folding to mobile networks for the ImageNet task, achieving reduced latency with minimal accuracy trade-offs.

METHOD

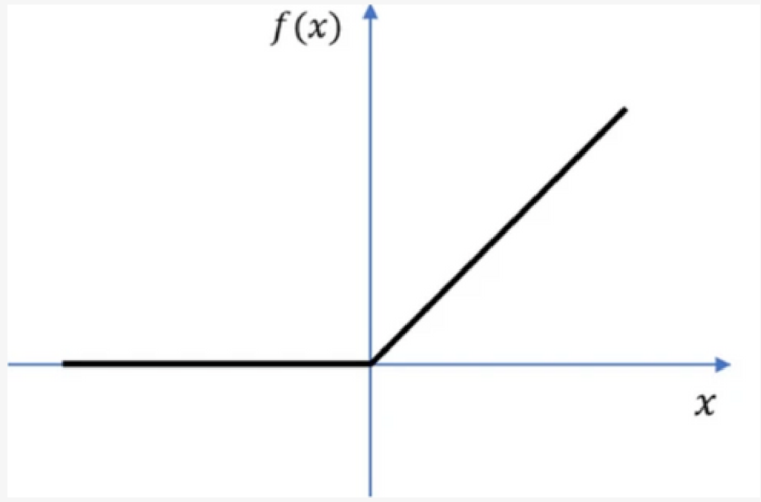
LAYER FOLDING

Simplifying Neural Networks by Reducing Non-Linear Activations

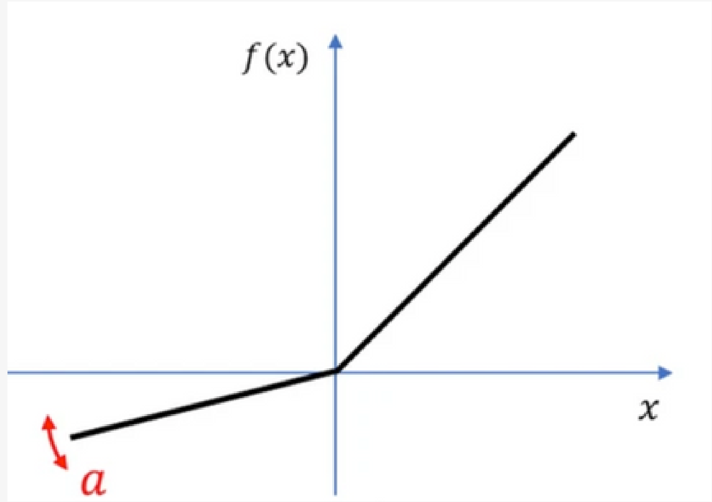
Method Overview: Introduces a technique to decrease the count of non-linear activations, by consolidating the neighboring linear layers into a singular layer.

STEP 1 ACTIVATION FUNCTION REPLACEMENT

$$\sigma_\alpha(x) = \alpha x + (1-\alpha)\sigma(x), \quad 0 \leq \alpha \leq 1$$



ReLU



PReLU

α : trainable parameter

(provides an interpolation between σ and the identity function)

F_α : network by transforming the activations

initializing with α=0, F_α = F

METHOD

LAYER FOLDING

$$\sigma_\alpha(x) = \alpha x + (1-\alpha)\sigma(x), \quad 0 \leq \alpha \leq 1$$

STEP 2 LOSS FUNCTION

to make some activations linear

$$\mathcal{L} = \mathcal{L}_t + \lambda_c \mathcal{L}_c$$

L_t : original task loss

L_c : auxiliary loss (penalizes smaller α values, encouraging them to become 1)

L : Achieves the main goal (maximizing classification accuracy) while simultaneously enabling additional goals (network simplification)

λc : hyperparameter that controls the number of layers to be folded

$$\mathcal{L}_c = \sum_{l \in L} c_l \, h(\alpha_l)$$

h(α) : monotonically decreasing function for 0 ≤ α ≤ 1

c_l : {cl}l∈L weigh the contribution of each layer to Lc

METHOD

LAYER FOLDING

$$\sigma_\alpha(x) = \alpha x + (1-\alpha)\sigma(x), \quad 0 \le \alpha \le 1$$

$$\mathcal{L} = \mathcal{L}_t + \lambda_c \mathcal{L}_c$$

STEP 2 LOSS FUNCTION

$$\mathcal{L}_c = \sum_{l \in L} c_l\, h(\alpha_l) \quad = \sum_{l \in L} c_l\, (1 - \alpha_l^p)$$

Choosing Auxiliary Loss Form

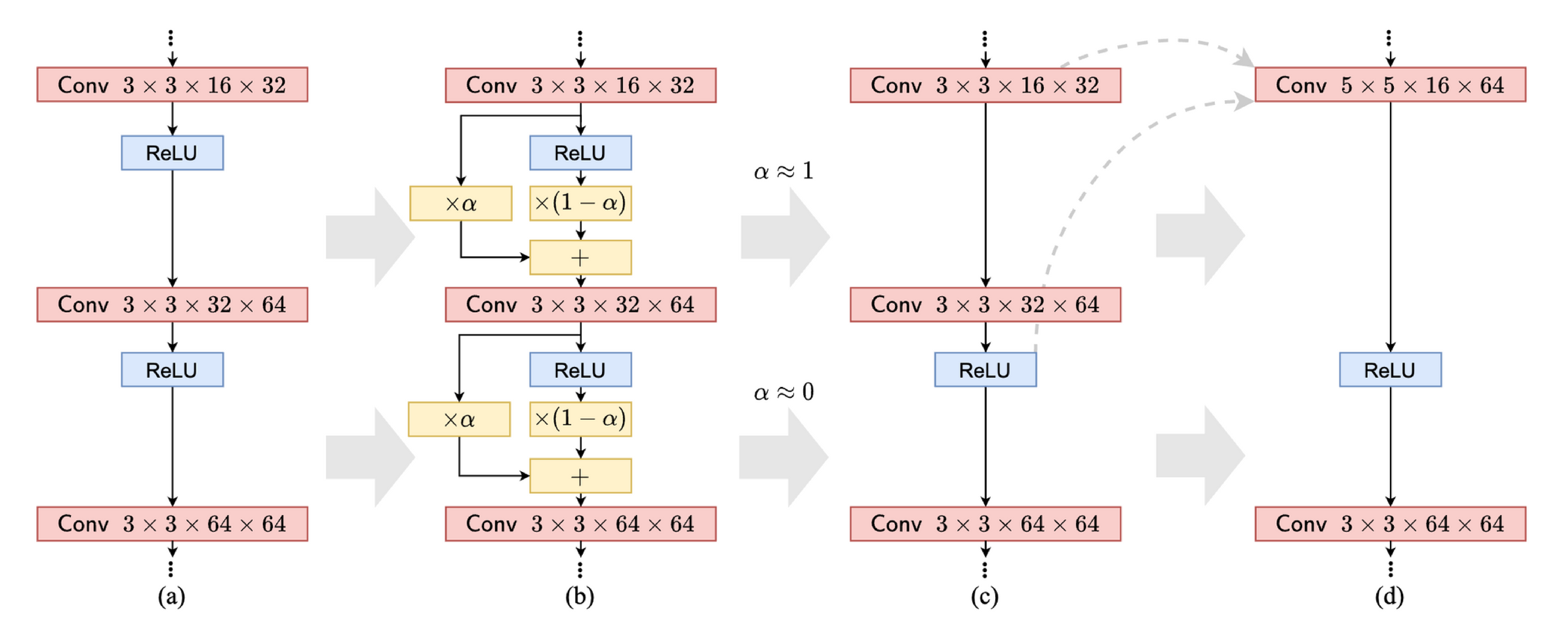
- **Sensitivity Near α=1**: Encourages significant loss reduction as α approaches 1. This effectively reduces network depth.

- **Indifference Near α=0**: Design ensures minimal loss change for α close to 0, avoiding penalizing layers where merging isn't considered, preserving original non-linear functions like ReLU

- **Regulating Loss Surface and Strong Push:** The hyperparameter p>1 adjusts the flatness of the loss surface around α=0 and strongly pushes larger α values towards 1.

OVERVIEW

LAYER FOLDING



METHOD

LAYER FOLDING

PHASE 1 PRE-FOLDING

Fine−tune F_α with the loss defined. When training converges, remove activations whose αs exceed a threshold τ and fold the corresponding adjacent layers, resulting in a shallower network.

PHASE 2 POST-FOLDING

fine−tune Ff old once more because the folded network may yet deviate from F_α due to various layers' attributes such as padding, resulting in a small accuracy decrease

EXPERIMENTS

SETTING

MNIST

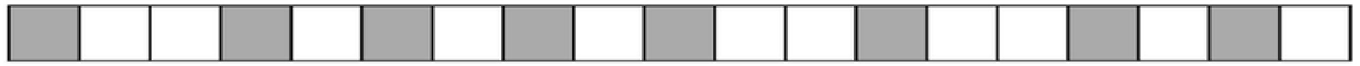
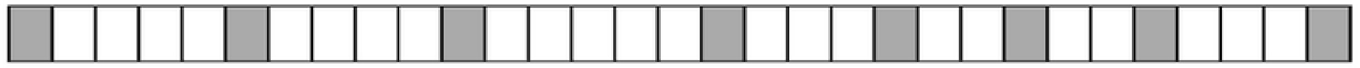
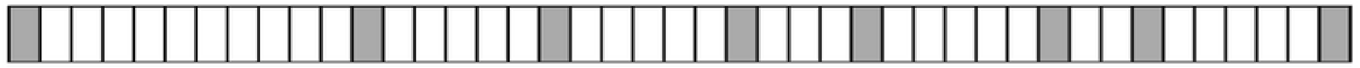
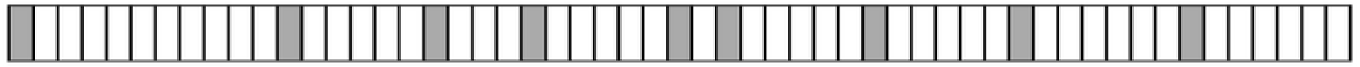
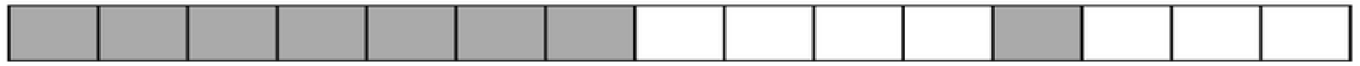
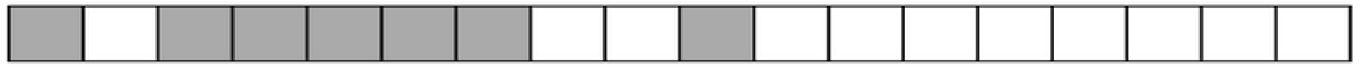
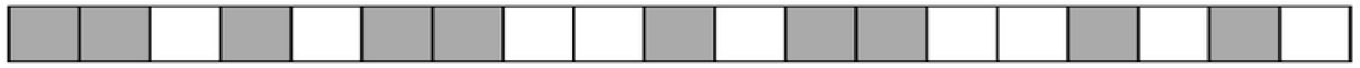
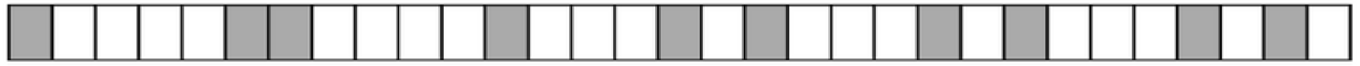
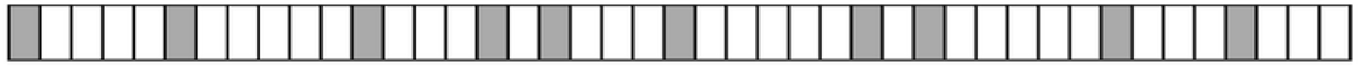
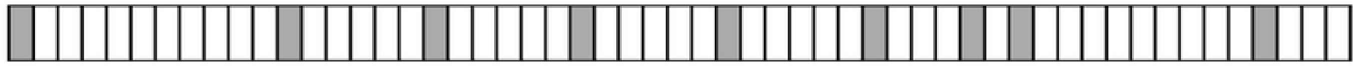
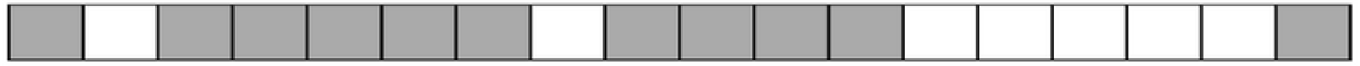
- Fully−connected network
- depth $L \in [2:10]$
- ReLU activation
- width $d = 256$

CIFAR-10 & CIFAR-100

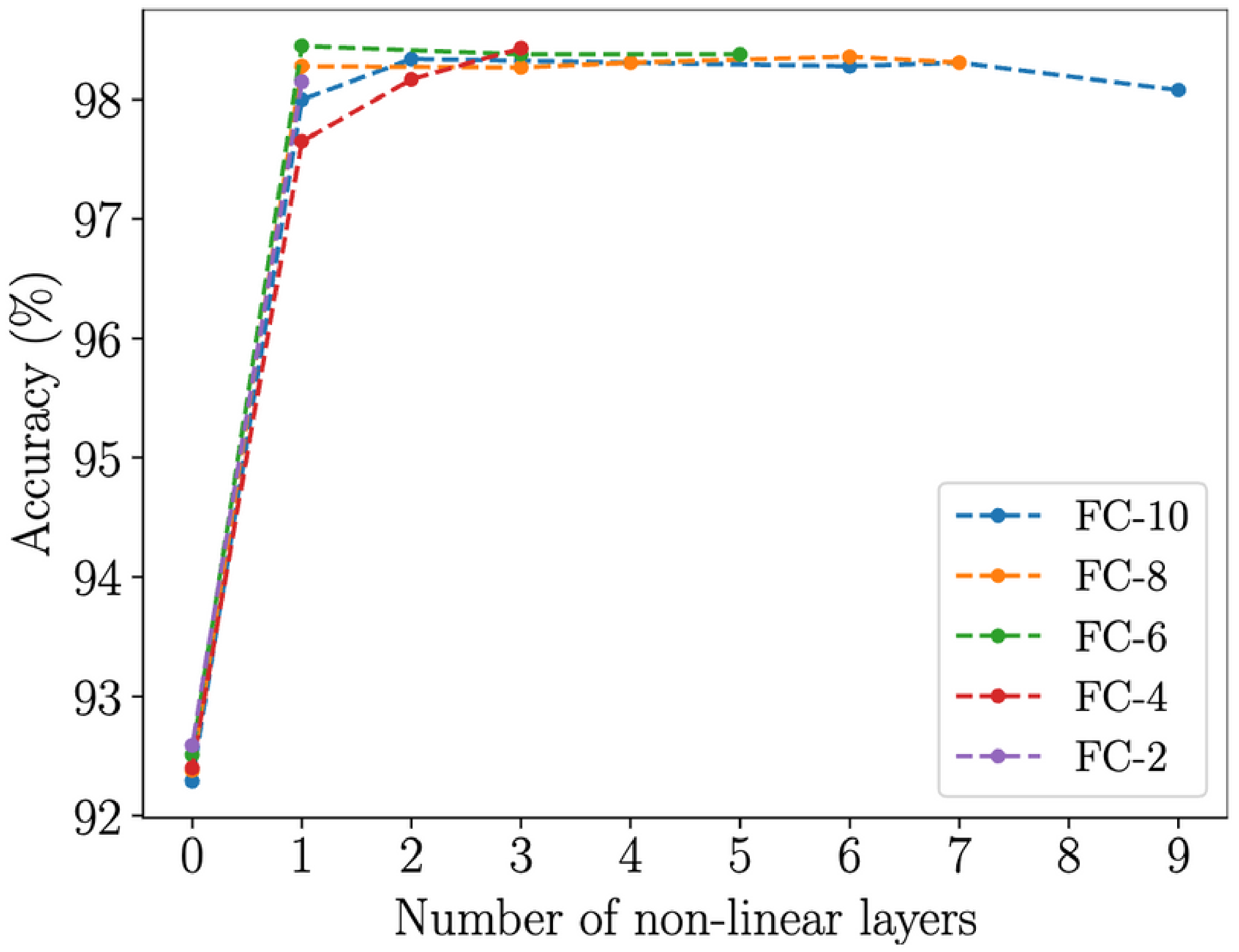
- ResNet models
  - depth $L \in \{20, 32, 44, 56\}$
- VGG models
  - depth $L \in \{16, 19\}$

apply Layer Folding with $c_l = 1$, $l = 1:L$, $p = 2$, $\tau = 0.9$ while varying λ_c to obtain shallower networks of varying depth.

EXPERIMENTS

Dataset	Model	Removed (white) and remaining (gray) activations	Depth	Acc. (%)
CIFAR-10	ResNet-20		9	89.82
	ResNet-32		9	90.02
	ResNet-44		9	89.88
	ResNet-56		10	90.29
	VGG16		9	93.89
	VGG19		8	93.23
CIFAR-100	ResNet-20		11	67.88
	ResNet-32		11	68.20
	ResNet-44		11	67.96
	ResNet-56		10	67.04
	VGG16		12	72.82
	VGG19		12	73.18

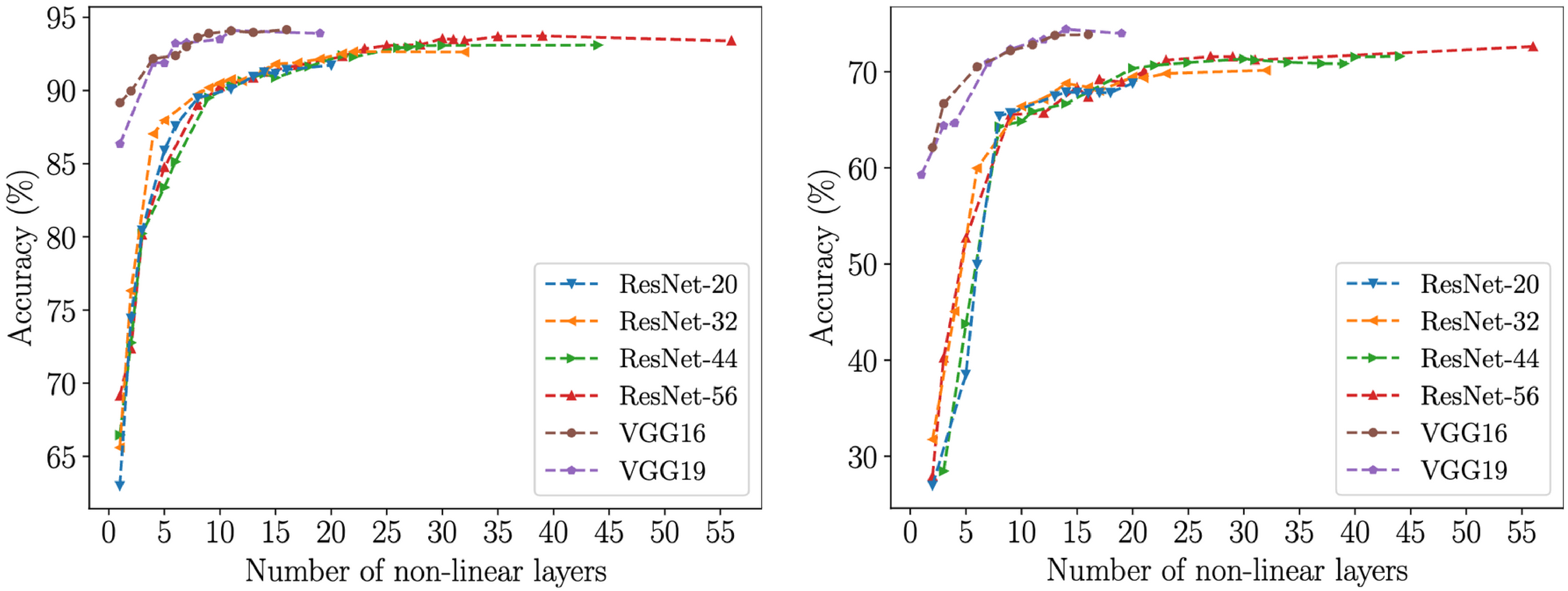
EXPERIMENTS



accuracy is roughly maintained down to a certain depth and drops below it

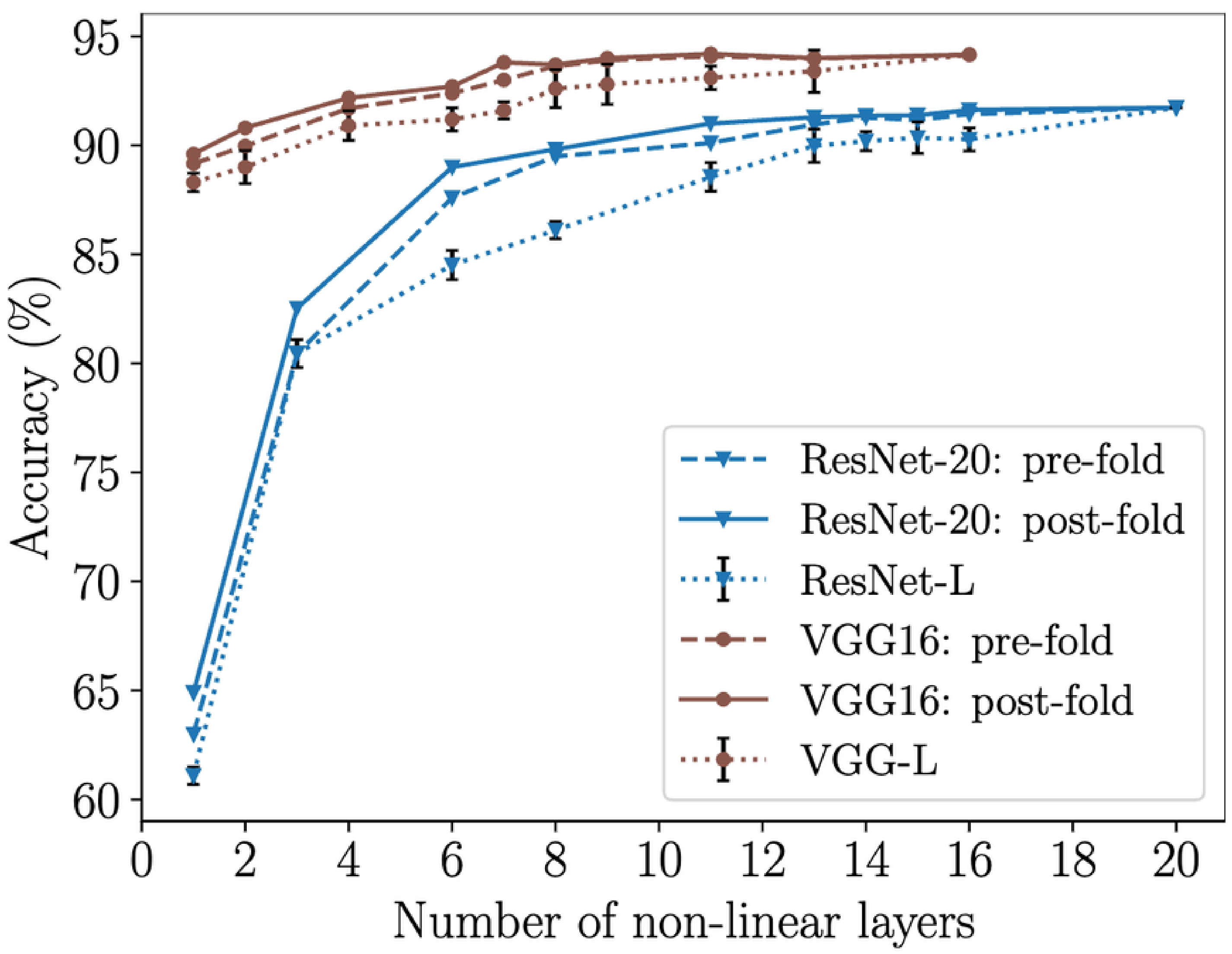
-> network possesses an EDNL

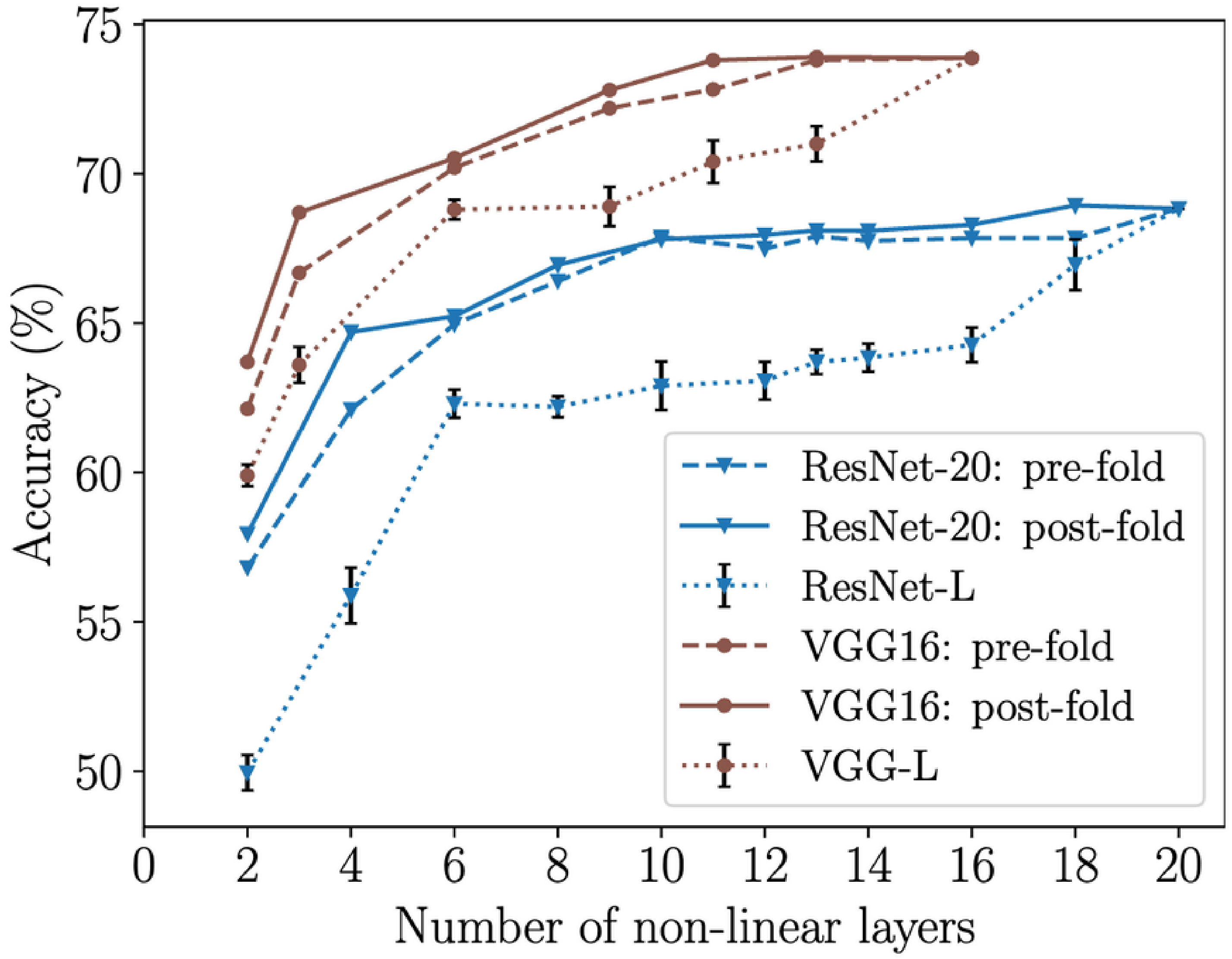
EXPERIMENTS



1. classification task with the added classes exhibits a slightly larger EDNL
2. such depth knee-point is shared for different networks over a particular task

EXPERIMENTS





EXPERIMENTS

Table 2: Latency and FLOPs reduction obtained by applying Layer Folding on MobileNetV2 (MNV2) and EfficientNet (EffNet) on ImageNet.

Model	Acc. (%) / Acc. Drop (%)	Latency Reduction	FLOPs Reduction
MNV2-0.75	68.1 / 1.7	21%	4%
MNV2-1.0	71.0 / 0.8	25%	7%
MNV2-1.4	75.5 / 0.5	19%	3%
EffNet-lite0	74.6 / 0.5	15%	3%
EffNet-lite1	75.8 / 1.0	13%	0%

CONCLUSION

This paper proposes a novel method for removing non-linear activations

- EDNL : minimal number of non-linear layers to which networks can be reduced while retaining accuracy
- scope of this work is EDNL evaluation of CNNs with ReLU activations
    - future extension to other architectures for future work
- showed reducing depth can aid latency reduction on hardware divices

QUESTIONS?