# An Efficient Alternative to Network Pruning through Ensemble Learning

**Martin Pöllot**, Rui Zhang, and André Kaup
{martin.poellot}, {rui.zhang}, {andre.kaup}@fau.de

ICASSP 2020, Barcelona, Spain

Multimedia Communications and Signal Processing

FRIEDRICH-ALEXANDER
UNIVERSITÄT
ERLANGEN-NÜRNBERG
TECHNISCHE FAKULTÄT

# Motivation

- Since the advent of AlexNet in 2010 [1], deep convolutional neural networks (CNNs) are dominating image classification leaderboards, like the Image Net LSVR Challenge[2]

- High recognition capabilities require two factors to be utilized:
  - Efficient GPU implementations allow training and inference in an acceptable amount of time (several hours/few days)
  - Deeper structures and thus more parameters yield better performance [3, 4]

Problem: Resource-critical situations, like embedded systems, lack storage and computational power to make use of aforementioned structures

[1] all references are equivalent to the numbers provided in the paper

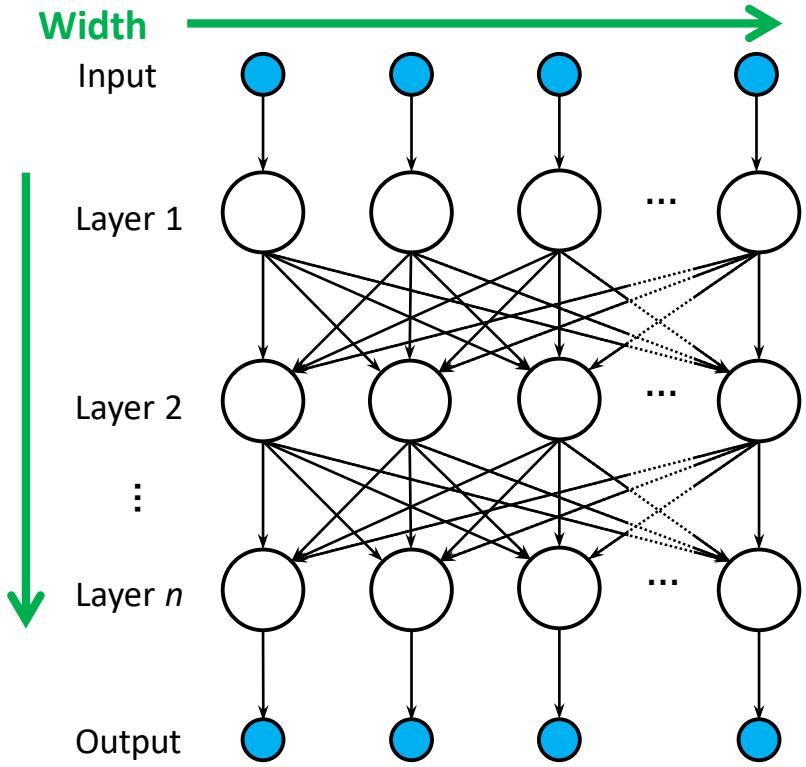
# Motivation

- In systems with high parameter count, a significant amount of redundancy is present [5]
- Reducing the parameter count, network pruning methods [6 - 10] can be applied to:
  - Reduce the number of parameters and
  - Achieve better generalization of networks with fewer parameters

- This paper presents an alternative to network pruning by combining multiple narrow networks to overcome resource critical situations and maintain classification capabilities

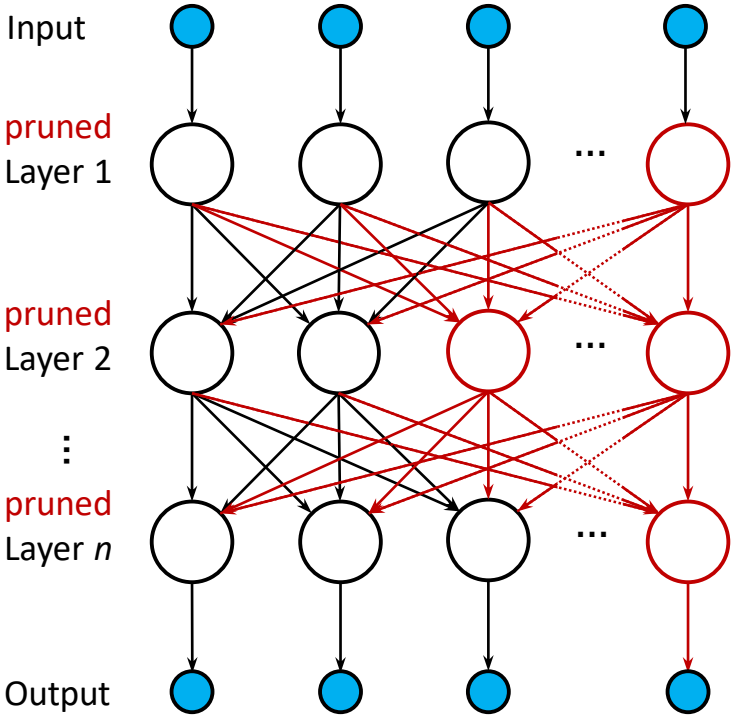
# State of the Art – Basics on Neural Networks

- Neural Networks are composed of layers
- Depth is more important than width in expressive power [12]
- Narrow networks with sufficient depth can still solve difficult tasks [13]
- However, wide networks like WRN [14] can achieve similar accuracy compared to narrower but deeper networks
- Pruning methods generally try to adjust the number of layers or their width to reduce the number of parameters [6, 9, 10, 15]



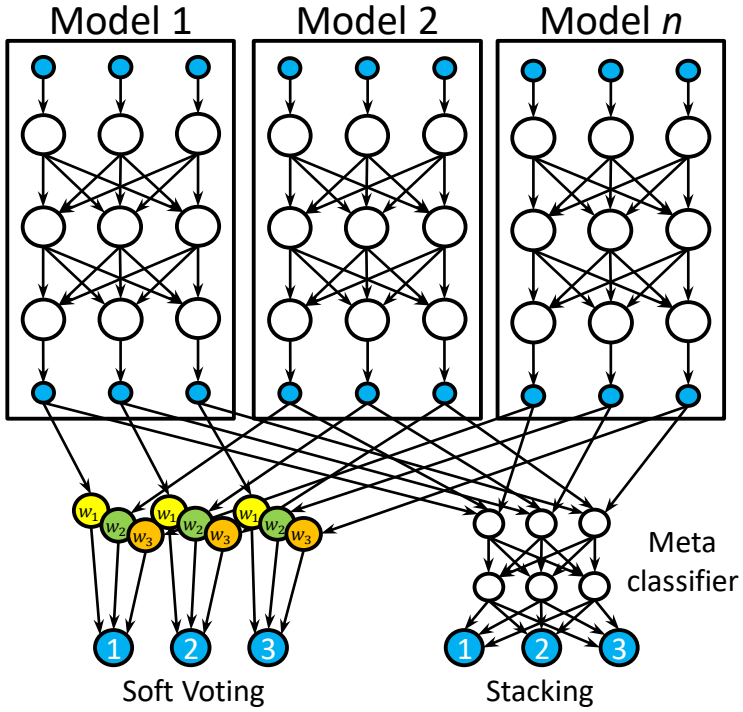
# State of the Art – Pruning

- **Pruning approaches** are usually applied during or after training [15] thus introducing even more complexity to the training process and computational resources, which is a disadvantage in itself
- **Advanced pruning** methods thus are applied before the training starts by adapting the network structure to a more efficient design
- Training a uniformly slimmed network with randomly initialized parameters performs better or similar to training and fine-tuning a network when the computational budgets are equal



# State of the Art – Ensemble Learning

- **Ensemble learning** describes the approach of using multiple models combined to an ensemble to achieve better generalization
- In contrast to pruning, ensemble learning **introduces** more parameters to the system
- Soft voting, where each model is trained separately, combines the softmax outputs of each model in a **weighted** fashion to compute the ensemble output
- Stacking is another method that introduces an extra meta-classifier with negligible size [11] that computes the most propable output of the ensemble networks



# Downsizing the Network

- Parameter size of a 2D-convolutional layer:

$$n_p = C_{\text{in}} C_{\text{out}} K^2$$

$C_{\text{in}}, C_{\text{out}}$: input and output channel number

$K^2$: Kernel size of the filter

- Number of floating-point operations (FLOPs):

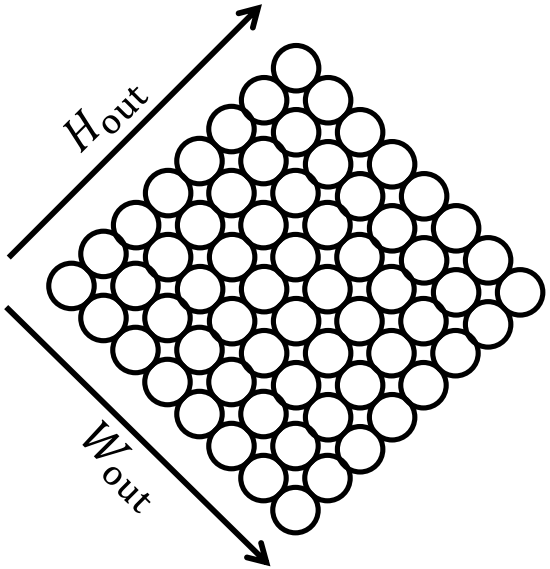
$$\text{FLOPs} = 2 C_{\text{in}} C_{\text{out}} H_{\text{out}} W_{\text{out}} K^2$$

$H_{\text{out}}, W_{\text{out}}$: output height and width of the feature map

- Introducing a multiplier $\alpha$:

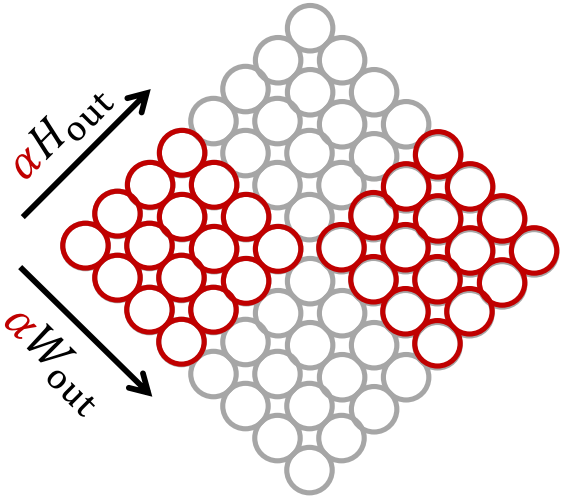
$$\alpha^2 n_p = \alpha C_{\text{in}} \alpha C_{\text{out}} K^2$$

$$\alpha^2 \text{FLOPs} = 2 \alpha C_{\text{in}} \alpha C_{\text{out}} H_{\text{out}} W_{\text{out}} K^2$$



# Training Downsized Network

- With $\alpha = 0.5$, the parameter size of a network is only $0.25$ of the original and results in half the parameter size for two downsized networks
- Applying ensemble methods to these networks yields very competitive classifying results while requiring less computational power and only half the parameter size
- Downsized network serves as a base-line with comparable accuracy to the full-sized model
- Pruned network is best trained from scratch

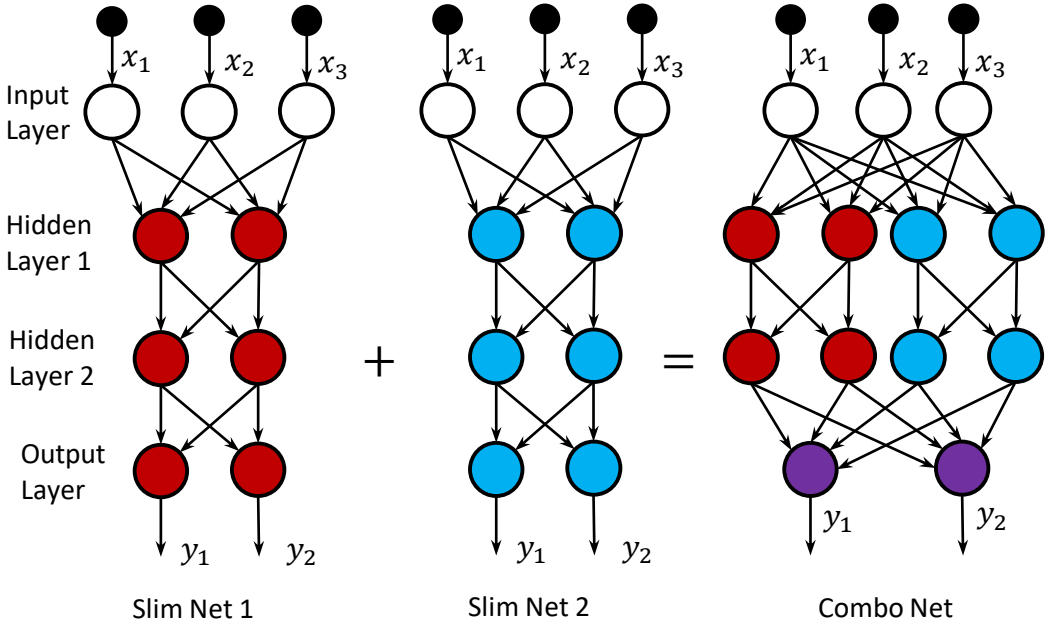


# Our Approach: ComboNet

- ComboNet is an alternative method to soft voting and stacking, as introduced earlier
- Input layer sends data to the subnets and has no parameters, so it can be shared accross nets
- Output layer is obtained by combining the output nodes of each network in a shared layer
- Adjusted Weights $W$ and biases $B$:

$$W_{\text{ComboNet}} = \frac{1}{N}[W_1; W_2; \ldots; W_N]$$

$$B_{\text{ComboNet}} = \frac{1}{N}\sum_{i=1}^{N} B_i$$



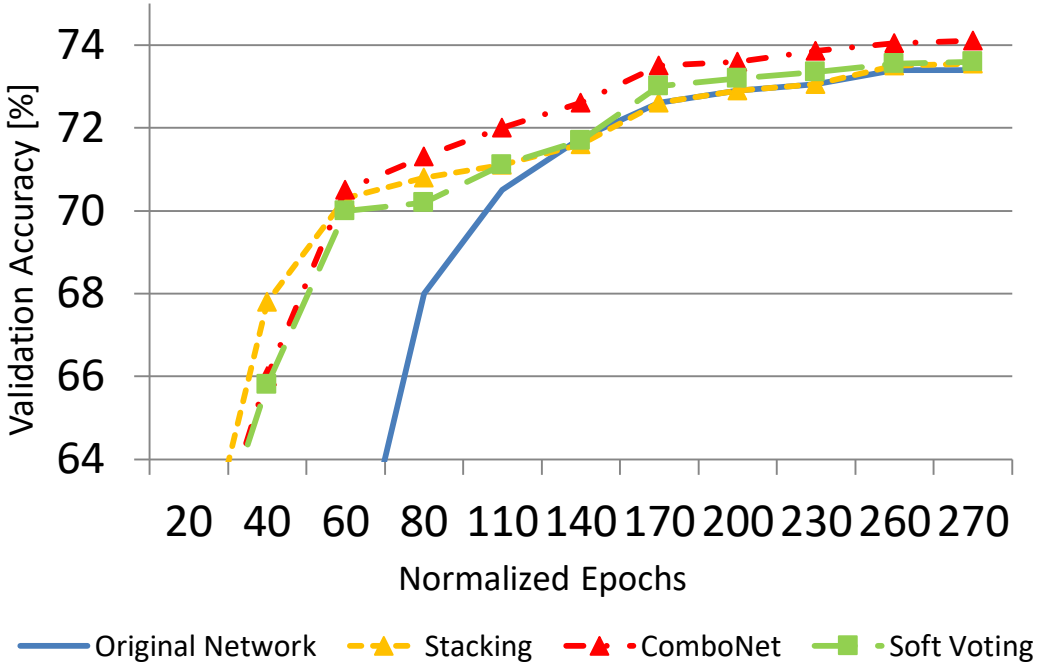
# Experiments

- To verify the advantages of our approach, we test three different neural network architectures with different $\alpha$: VGG-16 [3], MobileNetV2 [18] and WRN28-10 [14]
- We use images from three different data sets: CIFAR-10, CIFAR-100 and Oxford-Pet
- All images are rescaled to $224 \times 224 \times 3$ and contain only 10 classes
- Data augmentation is applied using random flipping and cropping
- Each network is assigned a budget of FLOPs for training which does not need to be exhausted if their maximum performance is reached
- For stacking, the meta-classifier consists of two dense layers, soft voting uses equal weights and the proposed ComboNet consists of networks with different values for $\alpha$
- We train each network for 100 epochs

# Results – Accuracy Compared to Full-Sized Network

| $\alpha$ | $N$ | Ensemble Method | VGG-16 | | | MobileNetV2 | | WRN | |
|---|---|---|---|---|---|---|---|---|---|
| | | | CIFAR-10 | CIFAR-100 | Oxford-Pet | CIFAR-10 | CIFAR-100 | CIFAR-10 | CIFAR-100 |
| $\sqrt{0.5}$ | 1 | | -0.06 | -0.16 | 1.29 | -5.44 | -2.45 | | |
| 0.5 | 1 | | -0.69 | -2.18 | 0.18 | -2.00 | -5.04 | -0.52 | -1.01 |
| 0.5 | 2 | Stacking | **0.43** | -0.01 | 0.82 | **-0.16** | -1.78 | 0.07 | 0.01 |
| 0.5 | 2 | Soft Voting | 0.37 | 0.32 | **1.63** | -0.45 | -1.59 | **0.14** | 0.40 |
| 0.5 | 2 | ComboNet | 0.41 | **0.65** | **1.63** | -0.23 | **-0.44** | 0.12 | **0.49** |
| 0.25 | 4 | Stacking | -0.16 | | | | | -0.34 | -0.58 |
| 0.25 | 4 | Soft Voting | -0.28 | | | | | -0.21 | 0.16 |
| 0.25 | 4 | ComboNet | -0.03 | | | | | -0.10 | **0.49** |

# Results – Training and Duration




| α | N | VGG-16 | MobileNetV2 | WRN28-10 |
|---|---|---|---|---|
| 1 | 1 | 67.77 | 17.84 | 106.26 |
| 0.5 | 1 | 26.36 | 10.20 | 35.85 |
| 0.5 | 2 | 54.52 | 19.50 | 72.90 |
| 0.25 | 8 | 101.76 | 73.54 | 145.52 |

Processing times for one batch of images with a resolution of $224 \times 224 \times 3$ in milliseconds on a computer with an Intel Xeon 10-core processor, equipped with one GTX1080 Ti with 11 GB V-RAM. The program that is used is PyTorch.

Validation accuracy of one test run during training. All ensemble methods are compared. The number of sub-networks is two in all cases with $\alpha = 0.5$.

# Conclusion

- In this paper, we investigated using the ensemble of slim networks to replace full-sized networks
- We are able to show the superiority of combining two slim networks with $\alpha$ set to 0.5
- We not only save half the amount of parameters and FLOPs but also deliver better accuracy
- We reduce the per-epoch training time which allows more hyperparameter configurations during training
- Our approach is superior to stacking and soft voting in most cases examined