MutualNet: Adaptive ConvNet via Mutual Learning from Network Width and Resolution

Taojiannan Yang[1], Sijie Zhu[1], Chen Chen[1], Shen Yan[2], Mi Zhang[2], and Andrew Willis[1]

[1] University of North Carolina at Charlotte
{tyang30,szhu3,chen.chen,arwillis}@uncc.edu
[2] Michigan State University
{yanshen6,mizhang}@msu.edu

Abstract. We propose the width-resolution mutual learning method (MutualNet) to train a network that is executable at dynamic resource constraints to achieve adaptive accuracy-efficiency trade-offs at runtime. Our method trains a cohort of sub-networks with different widths (i.e., number of channels in a layer) using different input resolutions to mutually learn multi-scale representations for each sub-network. It achieves consistently better ImageNet top-1 accuracy over the state-of-the-art adaptive network US-Net under different computation constraints, and outperforms the best compound scaled MobileNet in EfficientNet by 1.5%. The superiority of our method is also validated on COCO object detection and instance segmentation as well as transfer learning. Surprisingly, the training strategy of MutualNet can also boost the performance of a single network, which substantially outperforms the powerful AutoAugmentation in both efficiency (GPU search hours: 15000 vs. 0) and accuracy (ImageNet: 77.6% vs. 78.6%). *Code is available at https://github.com/taoyang1122/MutualNet.*

1 Introduction

Deep neural networks have triumphed over various perception tasks. However, deep networks usually require large computational resources, making them hard to deploy on mobile devices and embedded systems. This motivates research in reducing the redundancy in deep neural networks by designing efficient convolutional blocks [14, 25, 29, 39] or pruning unimportant network connections [1, 20, 21]. However, these works ignore the fact that the computational cost is determined by both the network scale and input scale. Only focusing on reducing network scale cannot achieve the optimal accuracy-efficiency trade-off. EfficientNet [33] has acknowledged the importance of balancing among network depth, width and resolution. But it considers network scale and input scale separately. The authors conduct grid search over different configurations and choose the best-performed one, while we argue that *network scale and input scale should be considered jointly in learning* to take full advantage of the information embedded in different configurations.

Table 1: Comparison between our framework and previous works.

Model	Adaptive	Network Scale	Input Scale	Mutual Learning (NS&IS)
MobileNet [14, 29]	✗	✓	✗	✗
ShuffleNet [25, 39]	✗	✓	✗	✗
EfficientNet [33]	✗	✓	✓	✗
US-Net [36]	✓	✓	✗	✗
MutualNet (Ours)	✓	✓	✓	✓

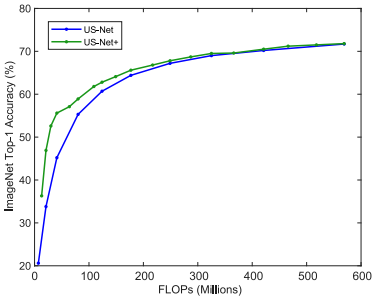


Fig. 1: Accuracy-FLOPs curves of US-Net+ and US-Net.

Another issue that prevents deep networks from practical deployment is that the resource budget (e.g., battery condition) varies in real-world applications, while traditional networks are only able to run at a specific constraint (e.g., FLOP). To address this issue, SlimNets [36, 37] are proposed to train a single model to meet the varying resource budget at runtime. They only reduce the network width to meet lower resource budgets. As a result, the model performance drops dramatically as computational resource goes down. Here, we provide a concrete example to show the importance of balancing between input resolution and network width for achieving better accuracy-efficiency trade-offs. Specifically, to meet a dynamic resource constraint from 13 to 569 MFLOPs on MobileNet v1 backbone, US-Net [36] needs a network width range of [0.05,1.0]$\times$ given a 224$\times$224 input, while this constraint can also be met via a network width of [0.25,1.0]$\times$ by adjusting the input resolution from {224, 192, 160, 128} during test time. We denote the latter model as US-Net+. As shown in Fig. 1, simply combining different resolutions with network widths *during inference* can achieve a better accuracy-efficiency trade-off than US-Net without additional efforts.

Inspired by the observations above, we propose a *mutual learning* scheme which incorporates both network width and input resolution into a unified learning framework. As depicted in Fig. 2, our framework feeds different sub-networks with different input resolutions. Since sub-networks share weights with each other, each sub-network can learn the knowledge shared by other sub-networks, which enables them to capture *multi-scale representations from both network level and input level*. Table 1 provides a comparison between our framework and previous works. In summary, we make the following contributions:

– We highlight the importance of input resolution for efficient network design. Previous works either ignore it or treat it independently from network structure. In contrast, we embed network width and input resolution in a unified

mutual learning framework to learn a deep neural network (MutualNet) that can achieve adaptive accuracy-efficiency trade-offs at runtime.
- We carry out extensive experiments to demonstrate the effectiveness of our MutualNet. It significantly outperforms independently-trained networks and US-Net on various network structures, datasets and tasks under different constraints. To the best of our knowledge, we are the first to benchmark arbitrary-constraint adaptive networks on object detection and instance segmentation.
- We conduct comprehensive ablation studies to analyze the proposed mutual learning scheme. We further demonstrate that our framework is promising to serve as a plug-and-play strategy to boost the performance of a single network, which substantially outperforms the popular performance-boosting methods, e.g., data augmentations [7, 9, 22], SENet [15] and knowledge distillation [26].
- The proposed framework is a general training scheme and model-agnostic. It can be applied to any networks without making any adjustments to the structure. This makes it compatible with other state-of-the-art techniques (e.g., Neural Architecture Search (NAS) [13, 32], AutoAugmentation [7, 22]).

2 Related Work

Light-weight Network. There has recently been a flurry of interest in designing light-weight networks. MobileNet [14] factorizes the standard 3×3 convolution into a 3×3 depthwise convolution and a 1×1 pointwise convolution which reduce computation cost by several times. ShuffleNet [39] separates the 1×1 convolution into group convolutions to further boost computation efficiency. MobileNet v2 [29] proposes the inverted residual and linear block for low-complexity networks. ShiftNet [35] introduces a zero-flop shift operation to reduce computation cost. Most recent works [13, 32, 34] also apply neural architecture search methods to search efficient networks. However, none of them considers the varying resource constraint during runtime in real-world applications. *To meet different resource budgets, these methods need to deploy several models and switch among them, which is not scalable.*

Adaptive Neural Networks. To meet the dynamic constraints in real-world applications, MSDNet [16] proposes a multi-scale and coarse to fine densenet framework. It has multiple classifiers and can make early predictions to meet varying resource demands. NestedNet [18] uses a nested sparse network which consists of multiple levels to enable nested learning. S-Net [37] introduces a 4-width framework to incorporate different complexities into one network, and proposes the switchable batch normalization for slimmable training. [27] leverages knowledge distillation to train a multi-exit network. However, these approaches can only execute at a limited number of constraints. US-Net [36] can instantly adjust the runtime network width for arbitrary accuracy-efficiency trade-offs. But its performance degrades significantly as the budget lower bound goes down. [3] proposes progressive shrinking to finetune sub-networks from a well-trained large network, but the training process is complex and expensive.

Multi-scale Representation Learning. The effectiveness of multi-scale representation has been explored in various tasks. FPN [23] fuses pyramid features

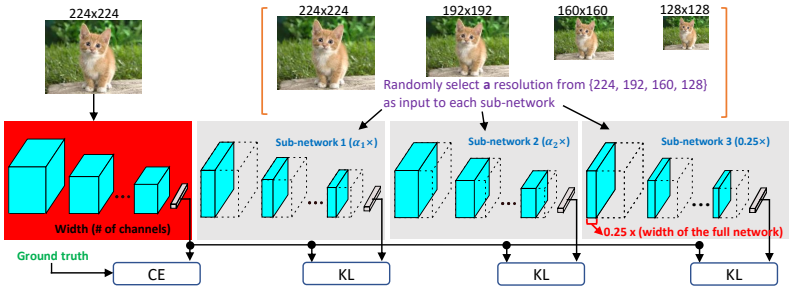


Fig. 2: The training process of our proposed MutualNet. The network width range is [0.25, 1.0]×, input resolution is chosen from {224, 192, 160, 128}. This can achieve a computation range of [13, 569] MFLOPs on MobileNet v1 backbone. We follow the *sandwich rule* [36] to sample 4 networks, i.e., upper-bound full width network (1.0×), lower-bound width network (0.25×), and **two random width ratios** $\alpha_1, \alpha_2 \in (0.25, 1)$. For the full-network, we constantly choose 224×224 resolution. For the other three sub-networks, we randomly select its input resolution. The full-network is optimized with the ground-truth label. Sub-networks are optimized with the prediction of the full-network. Weights are shared among different networks to facilitate mutual learning. CE: Cross Entropy loss. KL: Kullback–Leibler Divergence loss.

for object detection and segmentation. [17] proposes a multi-grid convolution to pass message across the scale space. HRNet [31] designs a multi-branch structure to exchange information across different resolutions. However, these works resort to the multi-branch fusion structure which is unfriendly to parallelization [25]. Our method does not modify the network structure, and the learned multi-scale representation is not only from image scale but also from network scale.

3 Methodology

3.1 Preliminary

Sandwich Rule. US-Net [36] trains a network that is executable at any resource constraint. The solution is to randomly sample several network widths for training and accumulate their gradients for optimization. However, the performance of the sub-networks is bounded by the smallest width (e.g., 0.25×) and the largest width (e.g., 1.0×). Thus, the *sandwich rule* is introduced to sample the smallest and largest widths plus two random ones for each training iteration.
Inplace Distillation. Knowledge distillation [12] is an effective method to transfer knowledge from a teacher network to a student network. Following the *sandwich rule*, since the largest network is sampled in each iteration, it is natural to use the largest network, which is supervised by the ground truth labels, as the teacher to guide smaller sub-networks in learning. This gives a better performance than training all sub-networks with ground truth labels.
Post-statistics of Batch Normalization (BN). US-Net proposes that each sub-network needs their own BN statistics (mean and variance), but it is insuf-

ficient to store the statistics of all the sub-networks. Therefore, US-Net collects BN statistics for the desired sub-network after training. Experimental results show that 2,000 samples are sufficient to get accurate BN statistics.

3.2 Rethinking Efficient Network Design

The computation cost of a vanilla convolution is $C_1 \times C_2 \times K \times K \times H \times W$, where C_1 and C_2 are the number of input and output channels, K is the kernel size, H and W are output feature map sizes. Most previous works only focus on reducing $C_1 \times C_2$. The most widely used group convolution decomposes standard convolution into groups to reduce the computation to $C_1 \times (C_2/g) \times K \times K \times H \times W$, where g is the number of groups. A larger g gives a lower computation but leads to higher memory access cost (MAC) [25], making the network inefficient in practical applications. Pruning methods [1,10,21] also only consider reducing structure redundancies.

In our approach, we shift the attention to reducing $H \times W$, i.e., lowering input resolution for the following reasons. First, as demonstrated in Fig. 1, balancing between width and resolution achieves better accuracy-efficiency trade-offs. Second, downsampling input resolution does not necessarily hurt the performance. It even sometimes benefits the performance. [6] points out that lower image resolution may produce better detection accuracy by reducing focus on redundant details. Third, different resolutions contain different information [5]. Lower resolution images may contain more global structures while higher resolution ones may encapsulate more fine-grained patterns. Learning multi-scale representations from different scaled images and features has been proven effective in previous works [17, 23, 31]. But these methods resort to a multi-branch structure which is unfriendly to parallelization [25]. Motivated by these observations, we propose a mutual learning framework to consider network scale and input resolution simultaneously for effective network accuracy-efficiency trade-offs.

3.3 Mutual Learning Framework

Sandwich Rule and Mutual Learning. As discussed in Section 3.2, different resolutions contain different information. We want to take advantage of this attribute to learn robust representations and better width-resolution trade-offs. The *sandwich rule* in US-Net can be viewed as a scheme of mutual learning [40] where an ensemble of networks are learned collaboratively. Since the sub-networks share weights with each other and are optimized jointly, they can transfer their knowledge to each other. Larger networks can take advantage of the features captured by smaller networks. Also, smaller networks can benefit from the stronger representation ability of larger networks. In light of this, we feed each sub-network with different input resolutions. By sharing knowledge, each sub-network is able to capture multi-scale representations.

Model Training. We present an example to illustrate our framework in Fig. 2. We train a network where its width ranges from 0.25× to 1.0×. We first follow the *sandwich rule* to sample four sub-networks, i.e., the smallest (0.25×),

the largest ($1.0\times$) and *two random width ratios* $\alpha_1, \alpha_2 \in (0.25, 1)$. Then, unlike traditional ImageNet training with 224×224 input, we resize the input image to four resolutions {224, 196, 160, 128} and feed them into different sub-networks. We denote the weights of a sub-network as $W_{0:w}$, where $w \in (0, 1]$ is the width of the sub-network and $0 : w$ means the sub-network adopts the first $w \times 100\%$ weights of each layer of the full network. $I_{R=r}$ represents a $r \times r$ input image. Then $N(W_{0:w}, I_{R=r})$ represents the output of a sub-network with width w and input resolution $r \times r$. For the largest sub-network (i.e., the full-network in Fig. 2), we always train it with the highest resolution (224×224) and ground truth label y. The loss for the full network is

$$loss_{full} = CrossEntropy(N(W_{0:1}, I_{R=224}),\, y). \tag{1}$$

For the other sub-networks, we randomly pick an input resolution from {224, 196, 160, 128} and train it with the output of the full-network. The loss for the i-th sub-network is

$$loss_{sub_i} = KLDiv(N(W_{0:w_i}, I_{R=r_i}),\, N(W_{0:1}, I_{R=224})), \tag{2}$$

where $KLDiv$ is the Kullback-Leibler divergence. The total loss is the summation of the full-network and sub-networks, i.e.,

$$loss = loss_{full} + \sum_{i=1}^{3} loss_{sub_i}. \tag{3}$$

The reason for training the full-network with the highest resolution is that the highest resolution contains more details. Also, the full-network has the strongest learning ability to capture the discriminatory information from the image data.

Mutual Learning from Width and Resolution. In this part, we explain why the proposed framework can mutually learn from different widths and resolutions. For ease of demonstration, we only consider two network widths $0.4\times$ and $0.8\times$, and two resolutions 128 and 192 in this example. As shown in Fig. 3, sub-network $0.4\times$ selects input resolution 128, sub-network $0.8\times$ selects input resolution 192. Then we can define the gradients for sub-network $0.4\times$ and $0.8\times$ as $\frac{\partial l_{W_{0:0.4}, I_{R=128}}}{\partial W_{0:0.4}}$ and $\frac{\partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0:0.8}}$, respectively. Since sub-network $0.8\times$ shares weights with $0.4\times$, we can decompose its gradient as

$$\frac{\partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0:0.8}} = \frac{\partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0:0.4}} \oplus \frac{\partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0.4:0.8}} \tag{4}$$

where $\oplus$ is vector concatenation. Since the gradients of the two sub-networks are accumulated during training, the total gradients are computed as

$$\begin{aligned} \frac{\partial L}{\partial W} &= \frac{\partial l_{W_{0:0.4}, I_{R=128}}}{\partial W_{0:0.4}} + \frac{\partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0:0.8}} \\ &= \frac{\partial l_{W_{0:0.4}, I_{R=128}}}{\partial W_{0:0.4}} + \left(\frac{\partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0:0.4}} \oplus \frac{\partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0.4:0.8}} \right) \\ &= \frac{\partial l_{W_{0:0.4}, I_{R=128}} + \partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0:0.4}} \oplus \frac{\partial l_{W_{0:0.8}, I_{R=192}}}{\partial W_{0.4:0.8}} \end{aligned} \tag{5}$$

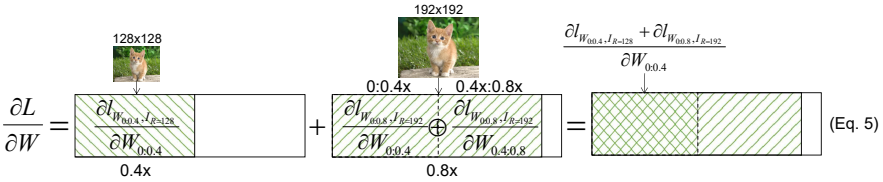


Fig. 3: Illustration of the mutual learning from network width and input resolution.

Therefore, the gradient for sub-network $0.4\times$ is $\frac{\partial l_{W_{0:0.4},I_{R=128}}+\partial l_{W_{0:0.8},I_{R=192}}}{\partial W_{0:0.4}}$, and it consists of two parts. The first part is derived from itself $(0 : 0.4\times)$ with 128 input resolution. The second part is derived from sub-network $0.8\times$ (i.e., $0 : 0.4\times$ portion) with 192 input resolution. Thus the sub-network is able to capture multi-scale representations from different input resolutions and network scales. Due to the random sampling of network width, every sub-network is able to learn multi-scale representations in our framework.

Model Inference. The trained model is executable at various width-resolution configurations. The goal is to find the best configuration under a particular resource constraint. A simple way to achieve this is via a query table. For example, in MobileNet v1, we sample network width from $0.25\times$ to $1.0\times$ with a step-size of $0.05\times$, and sample network resolution from {224, 192, 160, 128}. We test all these width-resolution configurations on a validation set and choose the best one under a given constraint (FLOPs or latency). Since there is no re-training, the whole process is once for all.

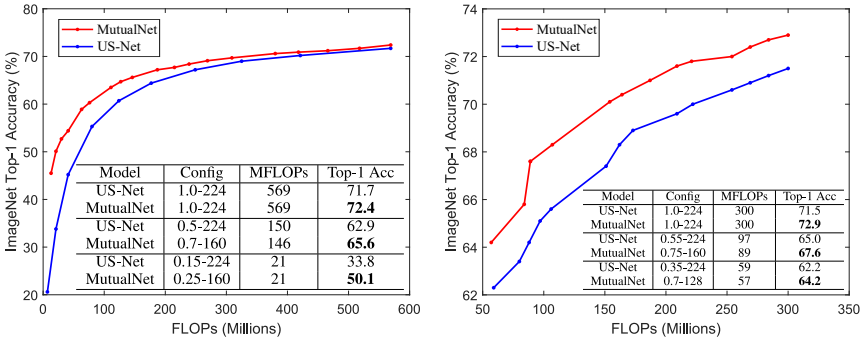
4 Experiments

In this section, we first present our results on ImageNet [8] classification to illustrate the effectiveness of MutualNet. Next, we conduct extensive ablation studies to analyze the mutual learning scheme. Finally, we apply MutualNet to transfer learning datasets and COCO [24] object detection and instance segmentation to demonstrate its robustness and generalization ability.

4.1 Evaluation on ImageNet Classification

We compare our MutualNet with US-Net and independently-trained networks on the ImageNet dataset. We evaluate our framework on two popular light-weight structures, MobileNet v1 [14] and MobileNet v2 [29]. These two networks also represent non-residual and residual structures respectively.

Implementation Details. We compare with US-Net under the **same** dynamic FLOPs constraints ([13, 569] MFLOPs on MobileNet v1 and [57, 300] MFLOPs on MobileNet v2). US-Net uses width scale $[0.05, 1.0]\times$ on MobileNet v1 and $[0.35, 1.0]\times$ on MobileNet v2 based on the 224×224 input resolution. **To meet the same dynamic constraints**, our method uses width scale $[0.25, 1.0]\times$ on



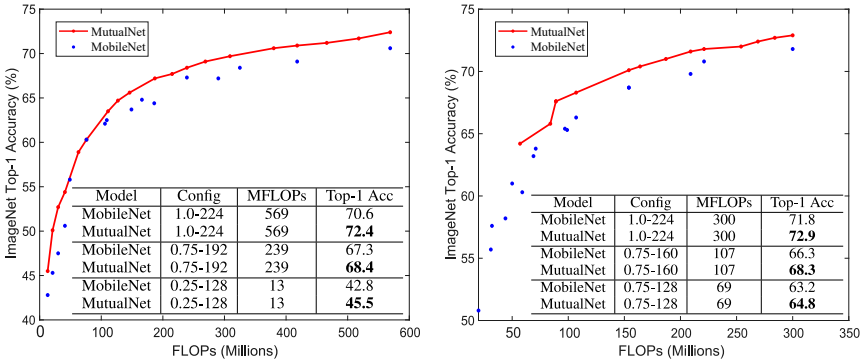
(a) MobileNet v1 backbone (b) MobileNet v2 backbone

Fig. 4: Accuracy-FLOPs curves of our proposed MutualNet and US-Net. (a) is based on MobileNet v1 backbone. (b) is based on MobileNet v2 backbone.

MobileNet v1 and [0.7, 1.0]× on MobileNet v2 with downsampled input resolutions {224, 192, 160, 128}. Due to the lower input resolutions, our method is able to use higher width lower bounds (i.e., 0.25× and 0.7×) than US-Net. The other training settings are the same as US-Net.

Comparison with US-Net. We first compare our framework with US-Net on MobileNet v1 and MobileNet v2 backbones. The Accuracy-FLOPs curves are shown in Fig. 4. We can see that our framework consistently outperforms US-Net on both MobileNet v1 and MobileNet v2 backbones. Specifically, we achieve significant improvements under small computation costs. This is because our framework considers both network width and input resolution, and can find a better balance between them. For example, if the resource constraint is 150 MFLOPs, US-Net has to reduce the width to 0.5× given its constant input resolution 224, while our MutualNet can meet this budget by a balanced configuration of (0.7× - 160), leading to a better accuracy (65.6% (Ours) vs. 62.9% (US-Net) as listed in the table of Fig. 4(a)). On the other hand, our framework is able to learn multi-scale representations which further boost the performance of each sub-network. We can see that even for the same configuration (e.g., 1.0×-224) our approach clearly outperforms US-Net, i.e., 72.4% (Ours) vs. 71.7% (US-Net) on MobileNet v1, and 72.9% (Ours) vs. 71.5% (US-Net) on MobileNet v2 (Fig. 4).

Comparison with Independently Trained Networks. Different scaled MobileNets are trained separately in [14, 29]. The authors consider width and resolution as independent factors, thus cannot leverage the information contained in different configurations. We compare the performance of MutualNet with independently-trained MobileNets under different width-resolution configurations in Fig. 5. For MobileNet v1, widths are selected from {1.0×, 0.75×, 0.5×, 0.25×}, and resolutions are selected from {224, 192, 160, 128}, leading to 16 configurations in total. Similarly, MobileNet v2 selects configurations from {1.0×, 0.75×, 0.5×, 0.35×} and {224, 192, 160, 128}. From Fig. 5, our frame-



Model	Config	MFLOPs	Top-1 Acc
MobileNet	1.0-224	569	70.6
MutualNet	1.0-224	569	**72.4**
MobileNet	0.75-192	239	67.3
MutualNet	0.75-192	239	**68.4**
MobileNet	0.25-128	13	42.8
MutualNet	0.25-128	13	**45.5**

(a) MobileNet v1 backbone

Model	Config	MFLOPs	Top-1 Acc
MobileNet	1.0-224	300	71.8
MutualNet	1.0-224	300	**72.9**
MobileNet	0.75-160	107	66.3
MutualNet	0.75-160	107	**68.3**
MobileNet	0.75-128	69	63.2
MutualNet	0.75-128	69	**64.8**

(b) MobileNet v2 backbone

Fig. 5: Accuracy-FLOPs curves of our MutualNet and independently-trained MobileNets. (a) is MobileNet v1 backbone. (b) is MobileNet v2 backbone. The results for different MobileNets configurations are taken from the papers [14, 29].

work consistently outperforms MobileNets. Even for the same width-resolution configuration (although it may not be the best configuration MutualNet finds at that specific constraint), MutualNet can achieve much better performance. This demonstrates that MutualNet not only finds the better width-resolution balance but also learns stronger representations by the mutual learning scheme.

4.2 Ablation Study

Balanced Width-Resolution Configuration via Mutual Learning. As evident in Fig. 1, we can apply different resolutions to US-Net *during inference* to yield improvement over the original US-Net. However, this cannot achieve the optimal width-resolution balance due to lack of width-resolution mutual learning. In the experiment, we test US-Net at width scale $[0.25, 1.0]\times$ with input resolutions $\{224, 192, 160, 128\}$ and denote this improved model as US-Net+. In Fig. 6, we plot the Accuracy-FLOPs curves of our method and US-Net+ based on MobileNet v1 backbone, and highlight the selected input resolutions with different colors. As we decrease the FLOPs (569 $\rightarrow$ 468 MFLOPs), our MutualNet first reduces network width to meet the constraint while keeping the 224×224 resolution (red line in Fig. 6). After 468 MFLOPs, MutualNet selects lower input resolution (192) and continues reducing the width to meet the constraint. On the other hand, US-Net+ cannot find such balance. It always slims the network width and uses the same (224) resolution as the FLOPs decreasing until it goes to really low. This is because US-Net+ does not incorporate input resolution into the learning framework. Simply applying different resolutions during inference cannot achieve the optimal width-resolution balance.

Difference with EfficientNet. EfficientNet acknowledges the importance of balancing among network width, depth and resolution. But they are considered

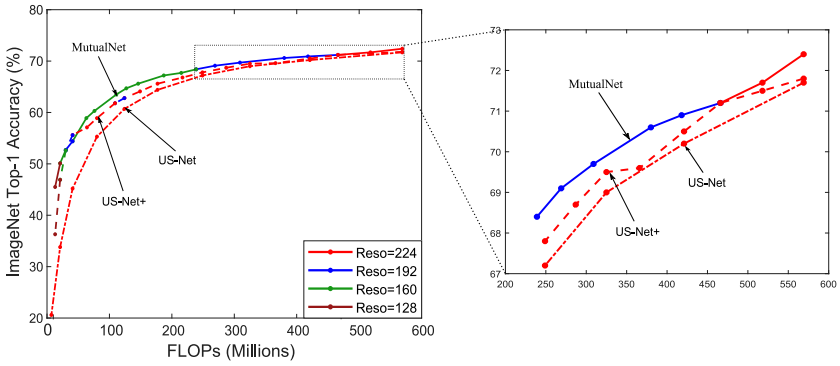


Fig. 6: The Accuracy-FLOPs curves are based on MobileNet v1 backbone. We highlight the selected resolution under different FLOPs with different colors. For example, the solid green line indicates when the constraint range is [41, 215] MFLOPs, our method constantly selects input resolution 160 but reduces the width to meet the resource constraint. Best viewed in color.

Table 2: ImageNet Top-1 accuracy on MobileNet v1 backbone. d: depth, w: width, r: resolution.

Model	Best Model Scaling	FLOPs	Top-1 Acc
EfficientNet [33]	$d = 1.4, w = 1.2, r = 1.3$	2.3B	75.6%
MutualNet	$w = 1.6, r = 1.3$	2.3B	**77.1%**

as independent factors. The authors use grid search over these three dimensions and train each configuration independently to find the optimal one under certain constraint, while our MutualNet incorporates width and resolution in a unified framework. We compare with the best model scaling EfficientNet finds for MobileNet v1 at 2.3 BFLOPs (scale up baseline by 4.0×). Similar to this scale setting, we train our framework with a width range of $[1.0\times, 2.0\times]$ (scaled by [1.0×,4.0×]), and select resolutions from {224, 256, 288, 320}. This makes MutualNet executable in the range of [0.57, 4.5] BFLOPs. We pick the best performed width-resolution configuration at 2.3 BFLOPs. The results are compared in Table 2. MutualNet achieves significantly better performance than EfficientNet because it can capture multi-scale representations for each model scaling due to the width-resolution mutual learning.

Difference with Multi-scale Data Augmentation. In multi-scale data augmentation, the network may take images of different resolutions in different iterations. But within each iteration, the network weights are still optimized in the same resolution direction. While our method randomly samples several sub-networks which share weights with each other. Since sub-networks can select different image resolutions, the weights are optimized in a mixed resolution

Table 3: Comparison between MutualNet and multi-scale data augmentation.

Model	ImageNet Top-1 Acc
MobileNet v2 (1.0× - 224) - Baseline	71.8%
Baseline + Multi-scale data augmentation	72.0%
MutualNet (MobileNet v2 backbone)	**72.9%**

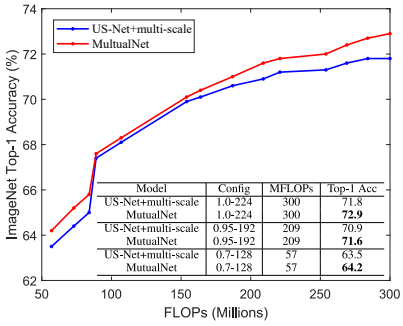


Fig. 7: MutualNet and US-Net + multi-scale data augmentation.

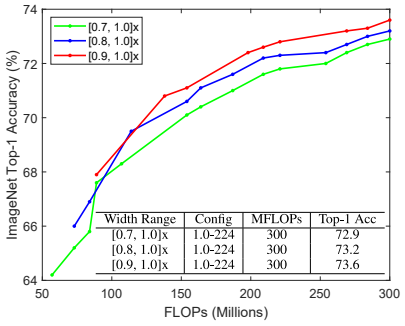


Fig. 8: Accuracy-FLOPs curves of different width lower bounds.

direction in each iteration as illustrated in Fig. 3. This enables each sub-network to effectively learn multi-scale representations from both network width and resolution. To validate the superiority of our mutual learning scheme, we apply multi-scale data augmentation to MobileNet and US-Net and explain the difference with MutualNet.

MobileNet + Multi-scale data augmentation. We train MobileNet v2 (1.0× width) with multi-scale images. To have a fair comparison, input images are randomly sampled from scales {224, 192, 160, 128} and the other settings are the same as MutualNet. As shown in Table 3, multi-scale data augmentation only marginally improves the baseline (MobileNet v2) while our MutualNet (MobileNet v2 backbone) clearly outperforms both of them by considerable margins.

US-Net + Multi-scale data augmentation. Different from our framework which feeds different scaled images to different sub-networks, in this experiment, we *randomly* choose a scale from {224, 192, 160, 128} and feed the *same* scaled image to all sub-networks in each iteration. That is each sub-network takes the same image resolution. In this way, the weights are still optimized towards a single resolution direction in each iteration. For example, as illustrated in Fig. 3, the gradient of the sub-network 0.4× in MutualNet is $\frac{\partial l_{W_{0:0.4},I_{R=128}}+\partial l_{W_{0:0.8},I_{R=192}}}{\partial W_{0:0.4}}$, while in US-Net + multi-scale it would be $\frac{\partial l_{W_{0:0.4},I_{R=128}}+\partial l_{W_{0:0.8},I_{R=128}}}{\partial W_{0:0.4}}$. With more sub-networks and input scales, the difference between their gradient flows becomes more distinct. As shown in Fig. 7, our method clearly performs better than *US-Net + multi-scale data augmentation* over the entire FLOPs spectrum.

Table 4: Top-1 Accuracy (%) on Cifar-10 and Cifar-100.

WideResNet -28-10	GPU search hours	C-10	C-100
Baseline	**0**	96.1	81.2
Cutout [9]	**0**	96.9	81.6
AA [7]	5000	**97.4**	82.9
Fast AA [22]	3.5	97.3	82.7
MutualNet	**0**	97.2	**83.8**

Table 5: Top-1 Accuracy (%) on ImageNet.

ResNet-50	Additional Cost	Top-1 Acc
Baseline	\	76.5
Cutout [9]	\	77.1
KD [26]	Teacher Network	76.5
SENet [15]	SE block	77.6
AA [7]	15000 GPU hours	77.6
Fast AA [22]	450 GPU hours	77.6
MutualNet	\	**78.6**

The experiment is based on MobileNet v2 with the same settings as in Sec. 4.1. *These experiments demonstrate that the improvement comes from our mutual learning scheme rather than the multi-scale data augmentation.*

Effects of Width Lower Bound. The executable constraint range and model performance are affected by the width lower bound. To study its effects, we conduct experiments with three different lower bounds (0.7×, 0.8×, 0.9×) on MobileNet v2. The results in Fig. 8 show that a higher lower bound gives better overall performance, but the executable range is narrower. One interesting observation is that the performance of the full-network (1.0×-224) is also largely improved as the width lower bound increases from 0.7× to 0.9×. This property is not observed in US-Net. We attribute this to the robust and well-generalized multi-scale representations which can be effectively re-used by the full-network, while in US-Net, the full-network cannot effectively benefit from sub-networks.

Boosting Single Network Performance. As discussed above, the performance of the full-network is greatly improved as we increase the width lower bound. Therefore, we apply our training framework to improve the performance of a single full network. We compare our method with the popular performance-boosting techniques (e.g., AutoAugmentation (AA) [7,22], SENet [15] and Knowledge Distillation (KD) [26]) to show its superiority. We conduct experiments using Wide-ResNet [38] on Cifar-10 and Cifar-100 and ResNet-50 on ImageNet. MutualNet adopts the width range [0.9, 1.0]× as it achieves the best-performed full-network in Fig. 8. The resolution is sampled from {32, 28, 24, 20} on Cifar-10 and Cifar-100 and {224, 192, 160, 128} on ImageNet. Wide-ResNet is trained for 200 epochs following [38]. ResNet is trained for 120 epochs. The results are compared in Table 4 and Table 5. MutualNet achieves substantial improvements over other techniques. It is important to note that MutualNet is a **general training scheme** which does not need the expensive searching procedure or additional network blocks or stronger teacher networks. Moreover, MutualNet training is as easy as the regular training process, and is orthogonal to other performance-boosting techniques, e.g., AutoAugmentation [7, 22]. Therefore, it can be easily combined with those methods.

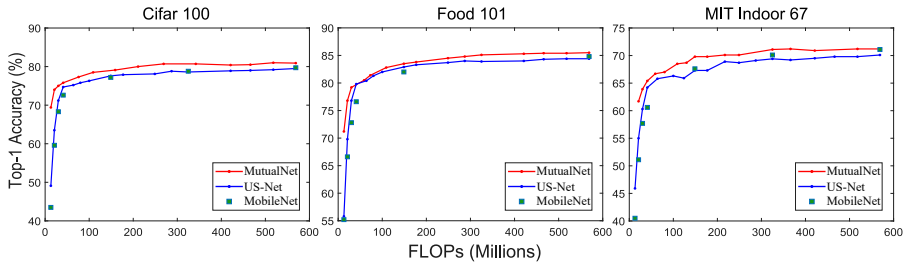


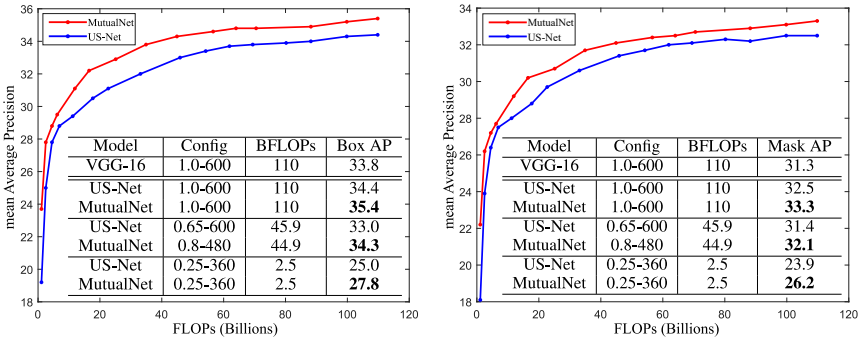
Fig. 9: Accuracy-FLOPs curves on different transfer learning datasets.

4.3 Transfer Learning

To evaluate the representations learned by our method, we further conduct experiments on three popular transfer learning datasets, Cifar-100 [19], Food-101 [2] and MIT-Indoor67 [28]. Cifar-100 is superordinate-level object classification, Food-101 is fine-grained classification and MIT-Indoor67 is scene classification. Such large variety is suitable to evaluate the robustness of the learned representations. We compare our approach with US-Net and MobileNet v1. We fine-tune ImageNet pre-trained models with a batch size of 256, initial learning rate of 0.1 with cosine decay schedule and a total of 100 epochs. Both MutualNet and US-Net are trained with width range $[0.25, 1.0]\times$ and tested with resolutions from $\{224, 192, 160, 128\}$. The results are shown in Fig. 9. Again, our MutualNet achieves consistently better performance than US-Net and MobileNet. This verifies that MutualNet is able to learn well-generalized representations.

4.4 Object Detection and Instance Segmentation

We also evaluate our method on COCO object detection and instance segmentation [24]. The experiments are based on Mask-RCNN-FPN [11, 23] and MMDetection [4] toolbox on VGG-16 [30] backbone. We first pre-train VGG-16 on ImageNet following US-Net and MutualNet respectively. Both methods are trained with width range $[0.25, 1.0]\times$. Then we fine-tune the pre-trained models on COCO. The FPN neck and detection head are shared among different sub-networks. For simplicity, we don't use *inplace distillation*. Rather, each sub-network is trained with the ground truth. The other training procedures are the same as training ImageNet classification. Following common settings in object detection, US-Net is trained with image resolution 1000×600. Our method randomly selects resolutions from $1000 \times \{600, 480, 360, 240\}$. All models are trained with $2\times$ schedule for better convergence and tested with different image resolutions. The mean Average Precision (AP at IoU=0.50:0.05:0.95) are presented in Fig. 10. These results reveal that our MutualNet significantly outperforms US-Net under all resource constraints. Specifically, for the full network ($1.0\times$-600), MutualNet significantly outperforms both US-Net and independent network. This again validates the effectiveness of our width-resolution mutual learning



Model	Config	BFLOPs	Box AP
VGG-16	1.0-600	110	33.8
US-Net	1.0-600	110	34.4
MutualNet	1.0-600	110	**35.4**
US-Net	0.65-600	45.9	33.0
MutualNet	0.8-480	44.9	**34.3**
US-Net	0.25-360	2.5	25.0
MutualNet	0.25-360	2.5	**27.8**

(a) Object Detection

Model	Config	BFLOPs	Mask AP
VGG-16	1.0-600	110	31.3
US-Net	1.0-600	110	32.5
MutualNet	1.0-600	110	**33.3**
US-Net	0.65-600	45.9	31.4
MutualNet	0.8-480	44.9	**32.1**
US-Net	0.25-360	2.5	23.9
MutualNet	0.25-360	2.5	**26.2**

(b) Instance Segmentation

Fig. 10: Average Precision - FLOPs curves of MutualNet and US-Net.

Fig. 11: Object detection and instance segmentation examples.

scheme. Fig. 11 provides some visual examples which reveal that MutualNet is more robust to small-scale and large-scale objects than US-Net.

5 Conclusion and Future Work

This paper highlights the importance of simultaneously considering network width and input resolution for efficient network design. A new framework namely MutualNet is proposed to mutually learn from network width and input resolution for adaptive accuracy-efficiency trade-offs. Extensive experiments have shown that it significantly improves inference performance per FLOP on various datasets and tasks. The mutual learning scheme is also an effective training strategy for boosting single network performance. The generality of the proposed framework allows it to translate well to generic problem domains.

Acknowledgements This material is based upon work supported by the National Science Foundation under Grant CNS-1910844.

References

1. Anwar, S., Hwang, K., Sung, W.: Structured pruning of deep convolutional neural networks. ACM Journal on Emerging Technologies in Computing Systems (JETC) **13**(3), 32 (2017)
2. Bossard, L., Guillaumin, M., Van Gool, L.: Food-101 – mining discriminative components with random forests. In: European Conference on Computer Vision (2014)
3. Cai, H., Gan, C., Wang, T., Zhang, Z., Han, S.: Once for all: Train one network and specialize it for efficient deployment. In: International Conference on Learning Representations (2020), `https://openreview.net/forum?id=HylxE1HKwS`
4. Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., et al.: Mmdetection: Open mmlab detection toolbox and benchmark. arXiv preprint arXiv:1906.07155 (2019)
5. Chen, Y., Fang, H., Xu, B., Yan, Z., Kalantidis, Y., Rohrbach, M., Yan, S., Feng, J.: Drop an octave: Reducing spatial redundancy in convolutional neural networks with octave convolution. arXiv preprint arXiv:1904.05049 (2019)
6. Chin, T.W., Ding, R., Marculescu, D.: Adascale: Towards real-time video object detection using adaptive scaling. arXiv preprint arXiv:1902.02910 (2019)
7. Cubuk, E.D., Zoph, B., Mane, D., Vasudevan, V., Le, Q.V.: Autoaugment: Learning augmentation strategies from data. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 113–123 (2019)
8. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. pp. 248–255. Ieee (2009)
9. DeVries, T., Taylor, G.W.: Improved regularization of convolutional neural networks with cutout. arXiv preprint arXiv:1708.04552 (2017)
10. Han, S., Pool, J., Tran, J., Dally, W.: Learning both weights and connections for efficient neural network. In: Advances in neural information processing systems. pp. 1135–1143 (2015)
11. He, K., Gkioxari, G., Dollár, P., Girshick, R.: Mask r-cnn. In: Proceedings of the IEEE international conference on computer vision. pp. 2961–2969 (2017)
12. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531 (2015)
13. Howard, A., Sandler, M., Chu, G., Chen, L.C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., et al.: Searching for mobilenetv3. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1314–1324 (2019)
14. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
15. Hu, J., Shen, L., Sun, G.: Squeeze-and-excitation networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 7132–7141 (2018)
16. Huang, G., Chen, D., Li, T., Wu, F., van der Maaten, L., Weinberger, K.Q.: Multi-scale dense networks for resource efficient image classification. arXiv preprint arXiv:1703.09844 (2017)
17. Ke, T.W., Maire, M., Yu, S.X.: Multigrid neural architectures. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6665–6673 (2017)
18. Kim, E., Ahn, C., Oh, S.: Nestednet: Learning nested sparse structures in deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 8669–8678 (2018)

19. Krizhevsky, A.: Learning multiple layers of features from tiny images. Tech. rep. (2009)
20. Lemaire, C., Achkar, A., Jodoin, P.M.: Structured pruning of neural networks with budget-aware regularization. arXiv preprint arXiv:1811.09332 (2018)
21. Li, H., Kadav, A., Durdanovic, I., Samet, H., Graf, H.P.: Pruning filters for efficient convnets. arXiv preprint arXiv:1608.08710 (2016)
22. Lim, S., Kim, I., Kim, T., Kim, C., Kim, S.: Fast autoaugment. In: Advances in Neural Information Processing Systems. pp. 6662–6672 (2019)
23. Lin, T.Y., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature pyramid networks for object detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2117–2125 (2017)
24. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014)
25. Ma, N., Zhang, X., Zheng, H.T., Sun, J.: Shufflenet v2: Practical guidelines for efficient cnn architecture design. In: Proceedings of the European Conference on Computer Vision (ECCV). pp. 116–131 (2018)
26. Mishra, A., Marr, D.: Apprentice: Using knowledge distillation techniques to improve low-precision network accuracy. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=B1ae1lZRb
27. Phuong, M., Lampert, C.H.: Distillation-based training for multi-exit architectures. In: The IEEE International Conference on Computer Vision (ICCV) (October 2019)
28. Quattoni, A., Torralba, A.: Recognizing indoor scenes. 2009 IEEE Conference on Computer Vision and Pattern Recognition pp. 413–420 (2009)
29. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, L.C.: Mobilenetv2: Inverted residuals and linear bottlenecks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4510–4520 (2018)
30. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
31. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 5693–5703 (2019)
32. Tan, M., Chen, B., Pang, R., Vasudevan, V., Sandler, M., Howard, A., Le, Q.V.: Mnasnet: Platform-aware neural architecture search for mobile. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2820–2828 (2019)
33. Tan, M., Le, Q.: EfficientNet: Rethinking model scaling for convolutional neural networks. In: Chaudhuri, K., Salakhutdinov, R. (eds.) Proceedings of the 36th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 97, pp. 6105–6114. PMLR, Long Beach, California, USA (09–15 Jun 2019), http://proceedings.mlr.press/v97/tan19a.html
34. Wu, B., Dai, X., Zhang, P., Wang, Y., Sun, F., Wu, Y., Tian, Y., Vajda, P., Jia, Y., Keutzer, K.: Fbnet: Hardware-aware efficient convnet design via differentiable neural architecture search. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 10734–10742 (2019)
35. Wu, B., Wan, A., Yue, X., Jin, P., Zhao, S., Golmant, N., Gholaminejad, A., Gonzalez, J., Keutzer, K.: Shift: A zero flop, zero parameter alternative to spatial convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 9127–9135 (2018)

36. Yu, J., Huang, T.: Universally slimmable networks and improved training techniques. arXiv preprint arXiv:1903.05134 (2019)
37. Yu, J., Yang, L., Xu, N., Yang, J., Huang, T.: Slimmable neural networks. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=H1gMCsAqY7
38. Zagoruyko, S., Komodakis, N.: Wide residual networks. In: Richard C. Wilson, E.R.H., Smith, W.A.P. (eds.) Proceedings of the British Machine Vision Conference (BMVC). pp. 87.1–87.12. BMVA Press (September 2016). https://doi.org/10.5244/C.30.87, https://dx.doi.org/10.5244/C.30.87
39. Zhang, X., Zhou, X., Lin, M., Sun, J.: Shufflenet: An extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 6848–6856 (2018)
40. Zhang, Y., Xiang, T., Hospedales, T.M., Lu, H.: Deep mutual learning. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4320–4328 (2018)