Model Accuracy Exploration with CIFAR-10

CSE455 Final Project

Jack Chuang, Warren Shen, Thomas Lin, John Cheng
March 2022

Outline

- Motivation
- Previous Work & Dataset
- Approaches and Training Results
- Experiments for Baseline Reinforcement
- Conclusion

Motivation

- To participate in the CIFAR-10 Competition
- To improve the performance of CIFAR-10 in coursework
- To Integrate pre-exist proposals and build our own model
- To achieve the target accuracy of 90 percent

Previous Work & Dataset

- 8 notable proposals for CIFAR-10 from ML Compiled
- Idea base on EfficientNet and Cutout
- Referenced code from GitHub
- Resources from Tensorflow and Keras

- Dataset: CIFAR-10 from official website

Approaches and Training Results

Scaling

- **Coefficient**

  depth: $d = \alpha^{\varphi}$

  width: $w = \beta^{\varphi}$

  resolution: $r = \gamma^{\varphi}$

- **Constraints**

  $\alpha \times \beta^2 \times \gamma^2 = 2$

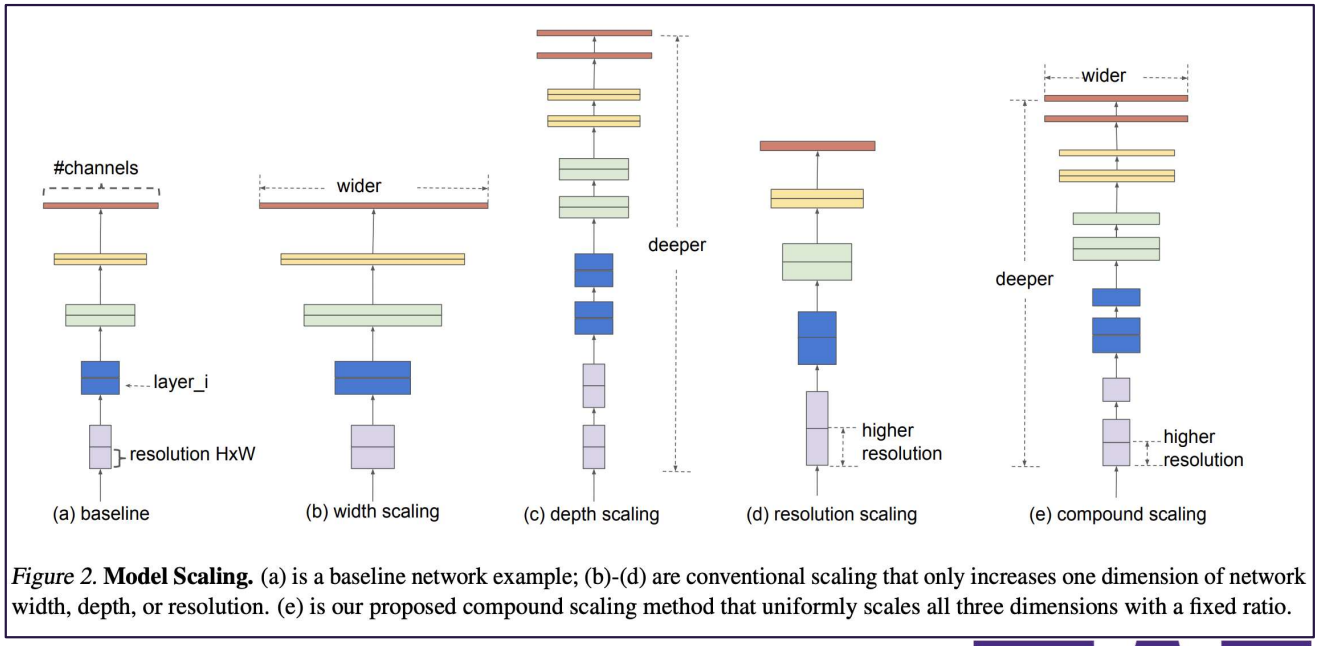


Figure 2. **Model Scaling.** (a) is a baseline network example; (b)-(d) are conventional scaling that only increases one dimension of network width, depth, or resolution. (e) is our proposed compound scaling method that uniformly scales all three dimensions with a fixed ratio.

Our Own Model of Baseline

> Dense Layer

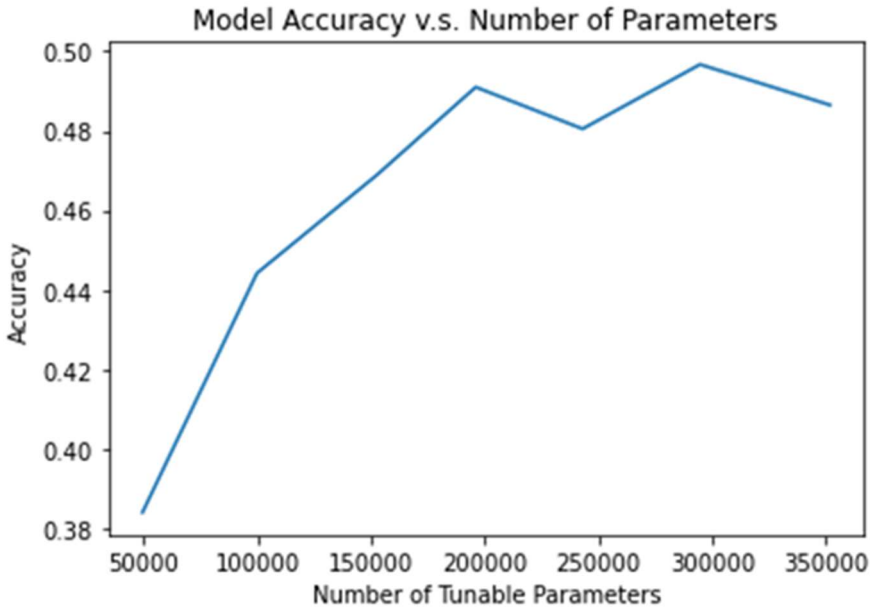
```
_________________________________________________________________
 Layer (type)                 Output Shape
=================================================================
 dense_83 (Dense)             (None, 16)

 dense_84 (Dense)             (None, 10)

=================================================================
Total params: 49,338
Trainable params: 49,338
Non-trainable params: 0
```

> Convolutional Layer

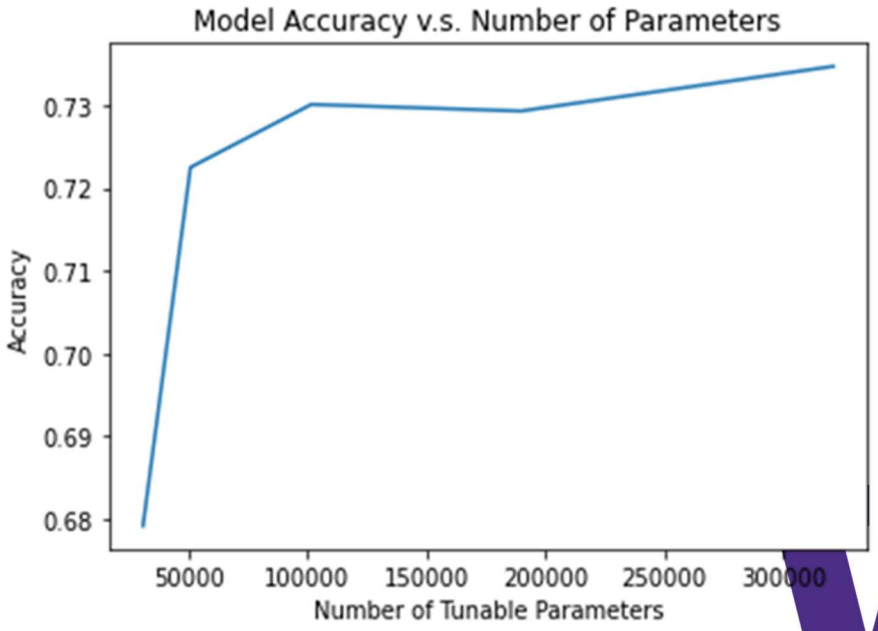
```
_________________________________________________________________
 Layer (type)                 Output Shape
=================================================================
 conv2d_1 (Conv2D)            (None, 32, 32, 16)

 max_pooling2d (MaxPooling2D  (None, 16, 16, 16)
 )

 flatten (Flatten)            (None, 4096)

 dense (Dense)                (None, 10)

=================================================================
Total params: 41,418
Trainable params: 41,418
Non-trainable params: 0
```

Training Results

> Dense Layer



> Convolutional Layer



Experiments for Baseline Reinforcement

Baseline Network

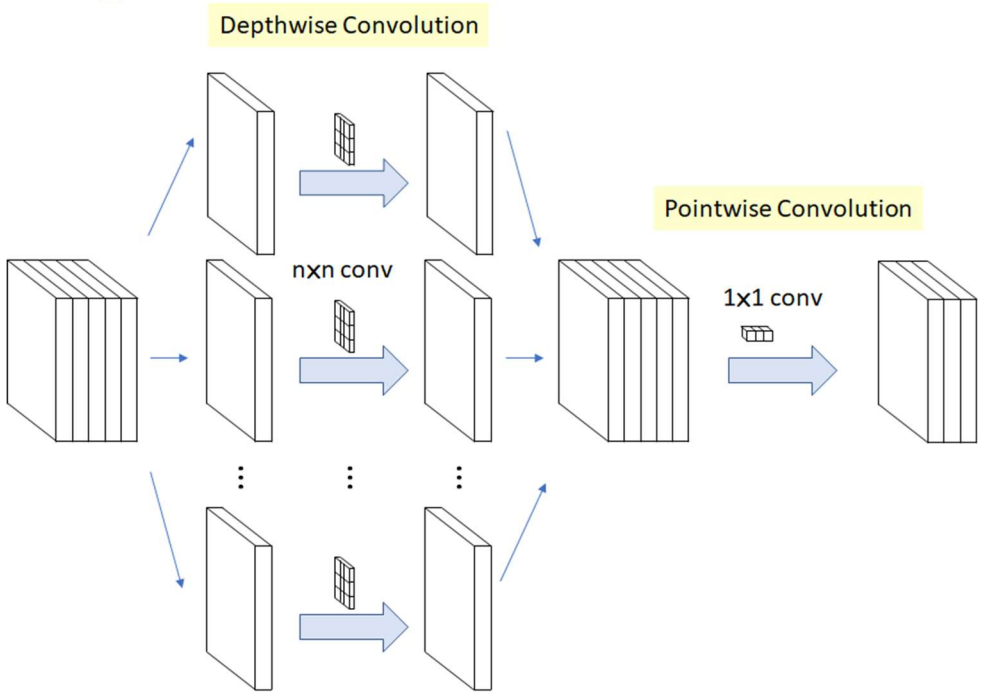
- To increase accuracy
  - Scaling
  - Baseline

Baseline Network (cont.)

- Baseline
  - MnasNet
  - VGG19
  - ResNet34

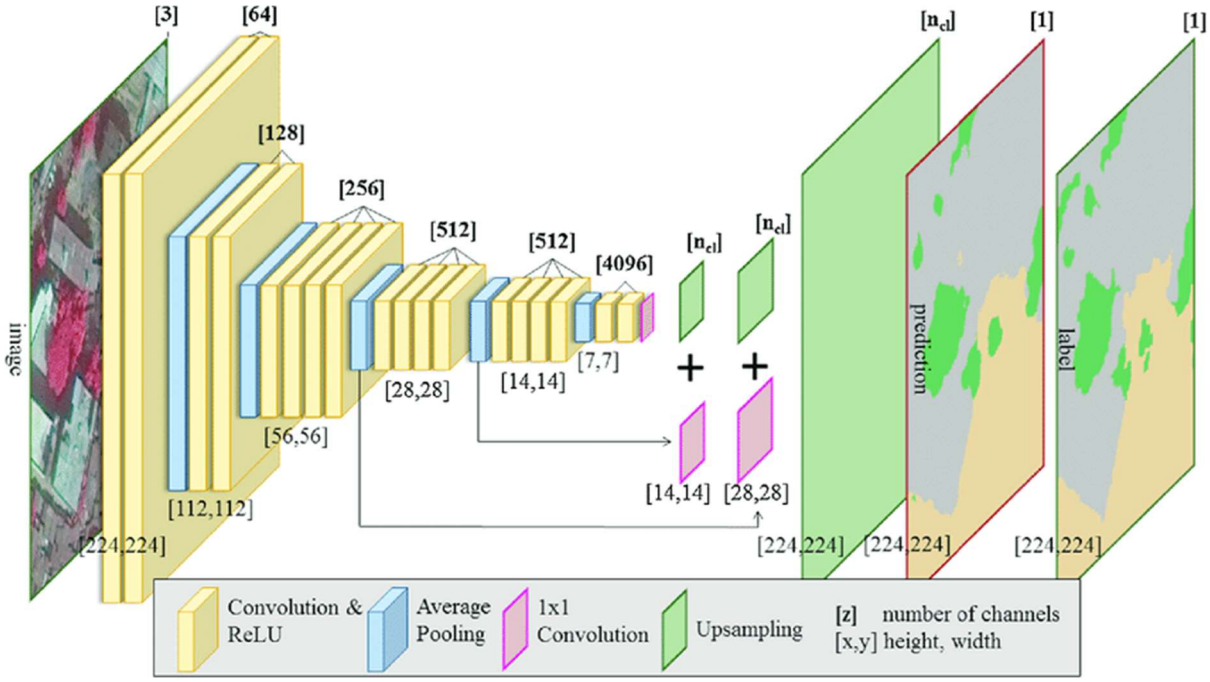
Baseline Network (cont.)

- MnasNet
  - Depthwise convolution
  - Every input filter has its kernel



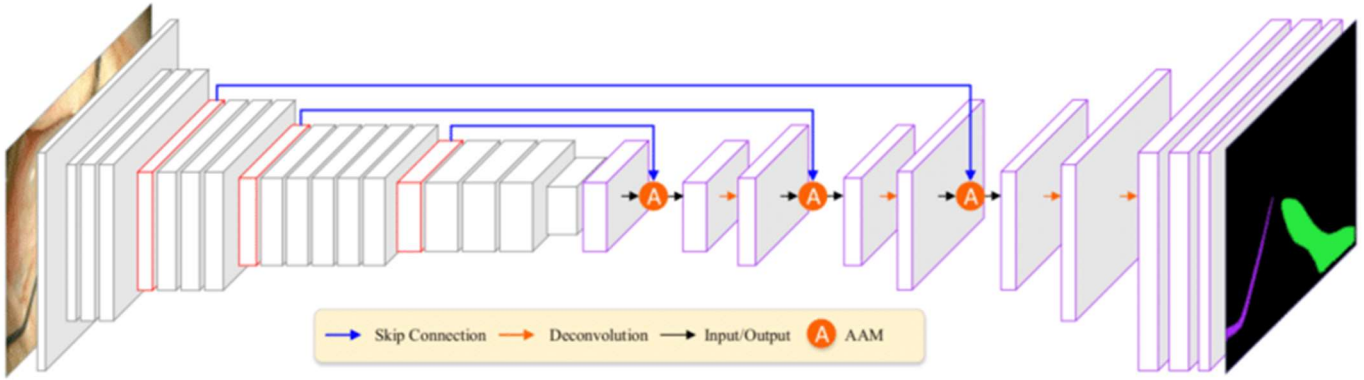
Baseline Network (cont.)

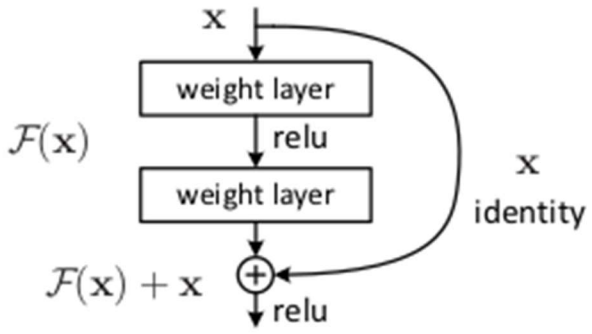
- VGG-19
  - More layers and filters!!



Baseline Network (cont.)

- ResNet34
  - Identity mapping





Baseline Network (cont.)

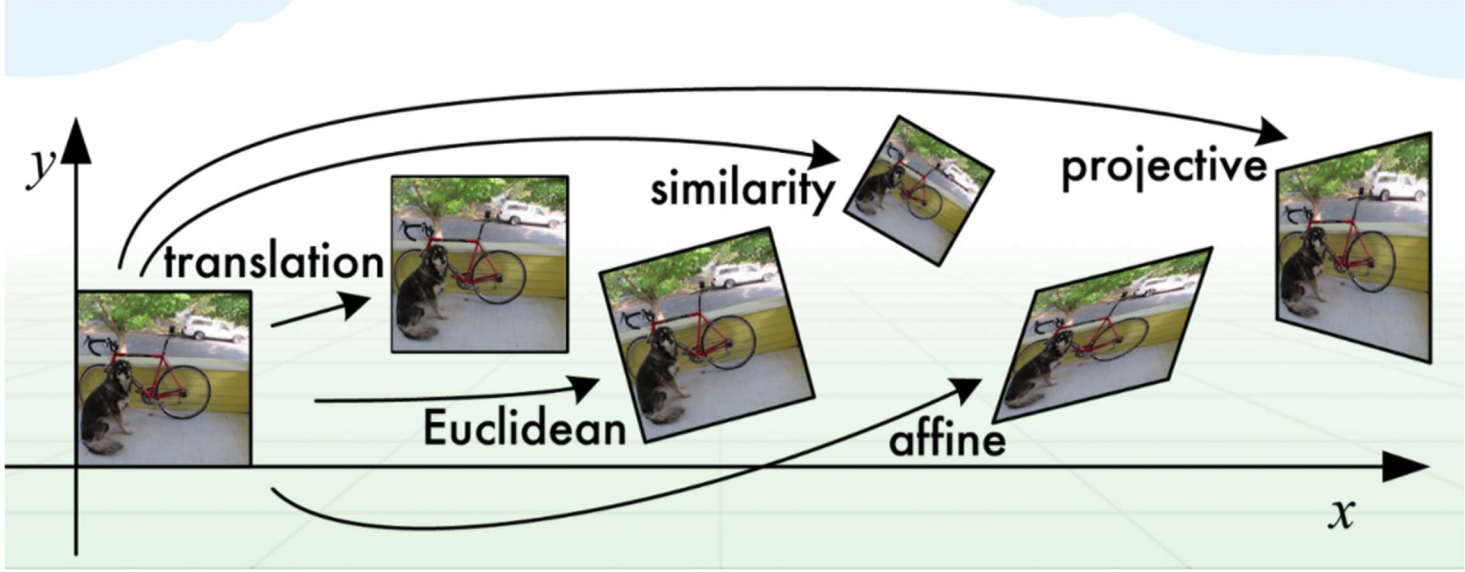
	MnasNet	VGG19	ResNet34
Accuracy	80%	84%	87%
Total Number of parameters	21,530,851	20,051,530	41,782,666

Regularization

- Data Augmentation (Preprocessing)
- Cutout
- Dropout

Affine transformation (Preprocessing)

> Linear conversion



source: lecture 07 slide

Dropout

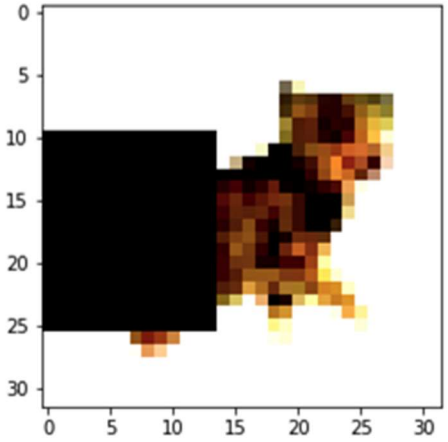
- Setting hidden unit activations to zero
- Less powerful in CNN

Cutout (Regularization)

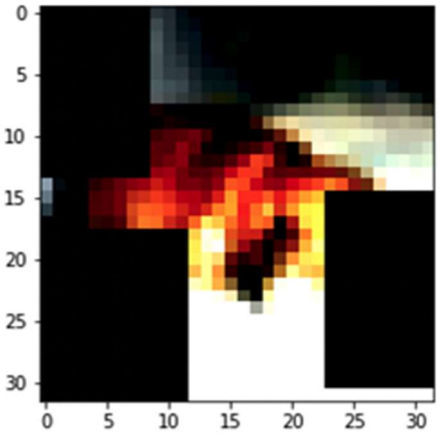
source: Cutout paper

Method	C10	C10+	C100	C100+	SVHN
ResNet18 [5]	10.63 ± 0.26	4.72 ± 0.21	36.68 ± 0.57	22.46 ± 0.31	-
ResNet18 + cutout	9.31 ± 0.18	3.99 ± 0.13	34.98 ± 0.29	21.96 ± 0.24	-
WideResNet [22]	6.97 ± 0.22	3.87 ± 0.08	26.06 ± 0.22	18.8 ± 0.08	1.60 ± 0.05
WideResNet + cutout	$\mathbf{5.54 \pm 0.08}$	3.08 ± 0.16	$\mathbf{23.94 \pm 0.15}$	18.41 ± 0.27	$\mathbf{1.30 \pm 0.03}$
Shake-shake regularization [4]	-	2.86	-	15.85	-
Shake-shake regularization + cutout	-	$\mathbf{2.56 \pm 0.07}$	-	$\mathbf{15.20 \pm 0.21}$	-

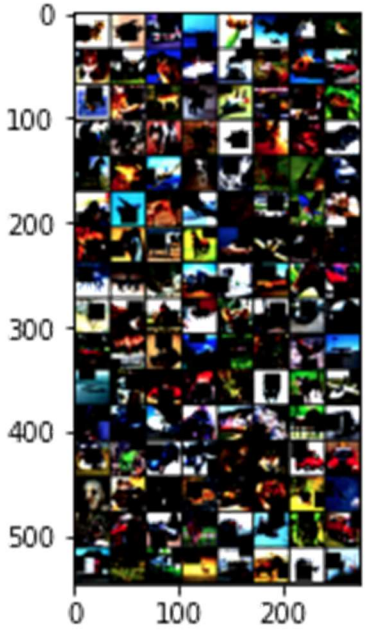
- Hole Number
- Hole size



Hole number = 1



Hole number = 3



Conclusion

Conclusion

- Improved training accuracy with CIFAR-10 from 50% to 87%
- Contributed a new model with reinforced scaling and baseline
- Designed the model for good hardware accessibility

Thanks for Listening

Discussion

Problem encountered

- Hardware limitation
- Transformation between PyTorch and TensorFlow

Next Steps

- To accomplish integration between baseline and scaling
- To improve training accuracy to 92 percent
- To apply transformative learning on different datasets
- To design new layers to upgrade the EfficientNet

What difference from others? beneficial?

- Integrated concepts from multiple papers
- Completed a training model with fine accessibility