Spatially Adaptive Computation Time for Residual Networks Supplementary materials

In the supplementary materials we present the implementation details and additional experimental results.

A. Implementation details

A.1. Image classification (ImageNet)

Optimization hyperparameters. ResNet, ACT and SACT use the same hyperparameters. We train the networks with 50 workers running asynchronous SGD with momentum 0.9, weight decay 0.0001 and batch size 32. The training is halted upon convergence after $150 - 160$ epochs. Learning rate is initially set to 0.05 and lowered by a factor of 10 after every 30 epochs. Batch normalization parameters are: epsilon 1e-5, moving average decay 0.997. The parameters of the network are initialized with a variance scaling initializer [1].

Data augmentation. We use the Inception v3 data augmentation procedure[1] which includes horizontal flipping, scale, aspect ratio, color augmentation. For the ImageNet images with a provided bounding box, we perform cropping based on the distorted bounding box. For evaluation, we take a single central crop of 87.5% of the original image's area and then resize this crop to the target resolution.

A.2. Object detection (COCO)

ResNet and SACT models use the same hyperparameters. The images are upscaled (preserving aspect ratio) so that the smaller side is at least 600 pixels. For data augmentation, we use random horizontal flipping as is described in [2]. We do not employ atrous convolution algorithm.

Optimization hyperparameters. We use distributed training with 9 workers running asynchronous SGD with momentum 0.9 and a batch size of 1. The learning rate is initially set to 0.0003 and lowered by a factor of 10 after the 800 thousandth and 1 millionth training iterations (batches). The training proceeds for a total of 1.2 million iterations. Batch normalization parameters are fixed to the values obtained on ImageNet during training.

Faster R-CNN hyperparameters. Other than the training method, our hyperparameters for the Faster R-CNN model closely follow those recommended by the original paper [2]. The anchors are generated in the same way, sampled from a regular grid of stride 16. One change relative to the original paper is the addition of an additional anchor size, so full set of anchor box sizes are $\{64, 128, 256, 512\}$, with the height and width also varying for each choice of the aspect ratios $\{0.5, 1, 2\}$. We use 300 object proposals per image. Non-maximum suppression is performed with 0.6 IoU threshold. For each proposal, the features are cropped into a 28×28 box with `crop_and_resize` TensorFlow operation, then pooled to 7×7.

A.3. Visual saliency (cat2000)

Here we describe the postprocessing procedure used in the visual saliency experiments. Consider a ponder cost map $\rho_{ij},\ i \in \{1, \dots, H\},\ j \in \{1, \dots, W\}$. Let G_s be a Gaussian filter with a standard deviation s.

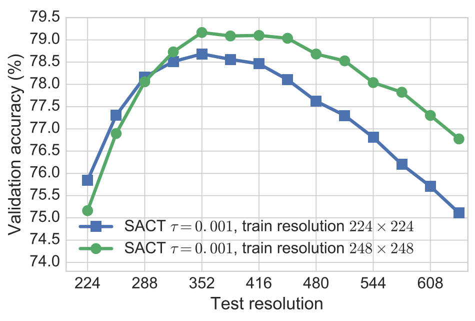
We first normalize this map to $[0, 1]$:

$$\rho^n_{ij} = \frac{\rho_{ij} - \rho_{\min}}{\rho_{\max} - \rho_{\min}},\ i \in \{1, \dots, H\},\ j \in \{1, \dots, W\}. \tag{1}$$

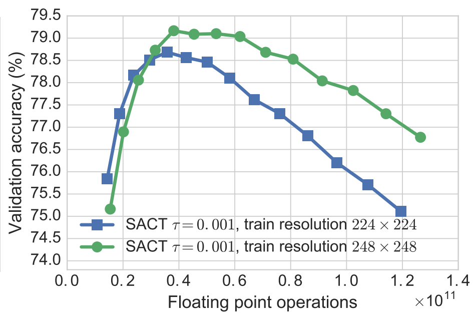
where

$$\rho_{\min} = \min_{ij} \rho_{ij} \tag{2}$$

[1] https://github.com/tensorflow/models/blob/master/inception/inception/image_processing.py



(a) Resolution *vs*. accuracy



(b) FLOPs *vs*. accuracy for varying test resolution

Figure 1: ImageNet validation set. Comparison of SACT with different training resolutions.

and similarly for $\max$.

Then, we blur the ponder cost map by convolving the Gaussian filter with ρ:

$$\rho^{nb} = G_s * \rho^n. \tag{3}$$

We obtain a baseline map B_{ij} by rescaling the reference centered Gaussian[2] to $H \times W$ resolution. We use this map with a weight $\gamma > 0$.

Finally, the postprocessed ponder cost map is defined as a normalized blurred ponder cost map plus the weighted center baseline map:

$$\rho_{ij}^{nbc} = \rho_{ij}^{nb} + \gamma B_{ij},\ i \in \{1, \ldots, H\},\ j \in \{1, \ldots, W\}. \tag{4}$$

ρ^{nbc} depends on two hyperparameters: the Gaussian filter standard deviation s and baseline map weight γ. The values are tuned via grid search. In the experiments in the paper, we use $s = 10$, $\gamma = 0.005$ for both models.

B. Additional ImageNet results

We present the extended results of ACT, SACT and ResNet models on ImageNet validation set, including the number of residual units per block, in table 1.

The ImageNet models in the paper are trained with 224×224 images. Even though all the models are fully convolutional and can be applied to images of any resolution during test time, increasing the training resolution can improve the quality of the model at the cost of longer training and higher GPU memory requirements. We have explored training of SACT model with a resolution of 248×248. This resolution is the highest we could fit into GPU memory for a batch size of 32 (decreasing the batch size deteriorates the accuracy). Comparison of SACT models trained with resolutions 224×224 and 248×248 is presented on fig. 1. Interestingly, both considered models achieve the highest accuracy at test resolution 352×352. This accuracy is 78.68% for the training resolution of 224×224 and 79.16% for the training resolution of 248×248. In the second case, the FLOPs are 6.6% higher. When the training resolution is increased, the accuracy at lower resolutions is predictably worse, while the accuracy at higher resolutions is better. The results suggests that training at higher resolutions is beneficial, and that the improved scale tolerance is not diminished when the training resolution is increased.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *CVPR*, 2015. 1

[2] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *NIPS*, 2015. 1

[2] https://github.com/cvzoya/saliency/blob/master/code_forOptimization/center.mat

Network	FLOPs	Residual units	Accuracy	Recall@5
ResNet-50	8.18×10^9	$3, 4, 6, 3$	74.56%	92.37%
ResNet-101	1.56×10^{10}	$3, 4, 23, 3$	76.01%	93.15%
ACT $\tau = 0.01$	$6.38 \times 10^9 \pm 3.31 \times 10^8$	$2.9 \pm 0.3, 2.7 \pm 0.5, 3.3 \pm 0.4, 3.0 \pm 0.0$	73.11%	91.52%
Baseline	6.43×10^9	$3, 3, 3, 3$	73.03%	91.68%
ACT $\tau = 0.005$	$8.12 \times 10^9 \pm 2.12 \times 10^8$	$3.0 \pm 0.0, 4.0 \pm 0.1, 5.9 \pm 0.5, 3.0 \pm 0.0$	73.95%	92.01%
Baseline	8.18×10^9	$3, 4, 6, 3$	74.34%	92.19%
ACT $\tau = 0.001$	$1.15 \times 10^{10} \pm 1.19 \times 10^9$	$3.0 \pm 0.0, 4.0 \pm 0.0, 13.7 \pm 2.7, 3.0 \pm 0.0$	75.05%	92.58%
Baseline	1.17×10^{10}	$3, 4, 14, 3$	75.69%	93.02%
ACT $\tau = 0.0005$	$1.34 \times 10^{10} \pm 1.21 \times 10^9$	$3.0 \pm 0.0, 4.0 \pm 0.0, 17.9 \pm 2.8, 3.0 \pm 0.0$	75.37%	92.76%
Baseline	1.34×10^{10}	$3, 4, 18, 3$	75.88%	93.02%
SACT $\tau = 0.01$	$6.61 \times 10^9 \pm 2.57 \times 10^8$	$2.6 \pm 0.5, 2.4 \pm 0.6, 4.0 \pm 0.9, 2.7 \pm 0.6$	73.28%	91.44%
Baseline	6.43×10^9	$3, 2, 4, 3$	73.33%	91.67%
SACT $\tau = 0.005$	$1.11 \times 10^{10} \pm 4.57 \times 10^8$	$2.3 \pm 0.4, 3.8 \pm 0.4, 13.1 \pm 2.6, 2.7 \pm 0.5$	75.61%	92.75%
Baseline	1.08×10^{10}	$2, 4, 13, 3$	75.57%	92.86%
SACT $\tau = 0.001$	$1.44 \times 10^{10} \pm 3.76 \times 10^8$	$3.0 \pm 0.0, 3.9 \pm 0.2, 19.6 \pm 2.4, 2.9 \pm 0.2$	75.84%	93.09%
Baseline	1.43×10^{10}	$3, 4, 20, 3$	76.06%	93.17%

(a) Test resolution 224×224

Network	FLOPs	Residual units	Accuracy	Recall@5
ResNet-50	2.02×10^{10}	$3, 4, 6, 3$	76.82%	93.80%
ResNet-101	3.85×10^{10}	$3, 4, 23, 3$	78.37%	94.60%
ACT $\tau = 0.01$	$1.58 \times 10^{10} \pm 8.22 \times 10^8$	$2.9 \pm 0.3, 2.7 \pm 0.5, 3.3 \pm 0.5, 3.0 \pm 0.0$	75.82%	93.18%
Baseline	1.59×10^{10}	$3, 3, 3, 3$	75.61%	93.14%
ACT $\tau = 0.005$	$2.01 \times 10^{10} \pm 4.19 \times 10^8$	$3.0 \pm 0.0, 4.0 \pm 0.1, 6.0 \pm 0.4, 3.0 \pm 0.0$	76.55%	93.57%
Baseline	2.02×10^{10}	$3, 4, 6, 3$	76.62%	93.63%
ACT $\tau = 0.001$	$2.95 \times 10^{10} \pm 2.59 \times 10^9$	$3.0 \pm 0.0, 4.0 \pm 0.0, 14.6 \pm 2.4, 3.0 \pm 0.0$	77.65%	94.14%
Baseline	2.88×10^{10}	$3, 4, 14, 3$	77.73%	94.31%
ACT $\tau = 0.0005$	$3.31 \times 10^{10} \pm 2.85 \times 10^9$	$3.0 \pm 0.0, 4.0 \pm 0.0, 18.0 \pm 2.6, 3.0 \pm 0.0$	77.84%	94.17%
Baseline	3.31×10^{10}	$3, 4, 18, 3$	78.10%	94.43%
SACT $\tau = 0.01$	$1.65 \times 10^{10} \pm 6.22 \times 10^8$	$2.6 \pm 0.5, 2.5 \pm 0.6, 4.1 \pm 0.8, 2.8 \pm 0.6$	76.34%	93.43%
Baseline	1.59×10^{10}	$3, 2, 4, 3$	75.99%	93.26%
SACT $\tau = 0.005$	$2.78 \times 10^{10} \pm 1.13 \times 10^9$	$2.3 \pm 0.5, 3.9 \pm 0.3, 13.4 \pm 2.7, 2.8 \pm 0.4$	78.39%	94.48%
Baseline	2.67×10^{10}	$2, 4, 13, 3$	77.57%	94.21%
SACT $\tau = 0.001$	$3.58 \times 10^{10} \pm 9.15 \times 10^8$	$3.0 \pm 0.0, 4.0 \pm 0.2, 19.9 \pm 2.4, 3.0 \pm 0.2$	78.68%	94.70%
Baseline	3.53×10^{10}	$3, 4, 20, 3$	78.23%	94.38%

(b) Test resolution 352×352

Table 1: ImageNet validation set. Comparison of ResNet, ACT, SACT and the respective baselines. All models are trained with 224×224 resolution images. $(x \pm y)$ denotes mean value x and one standard deviation y.