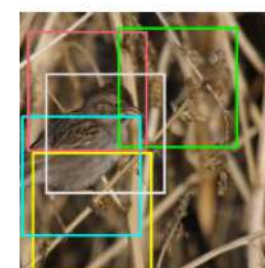
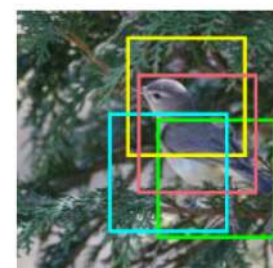
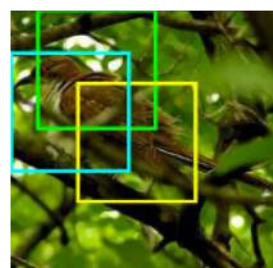
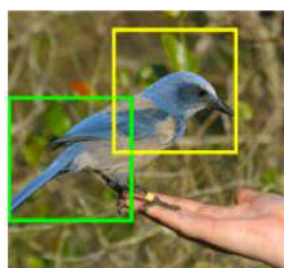
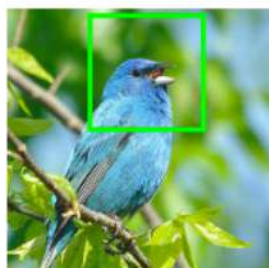


Dynamic Computational Time for Recurrent Visual Attention

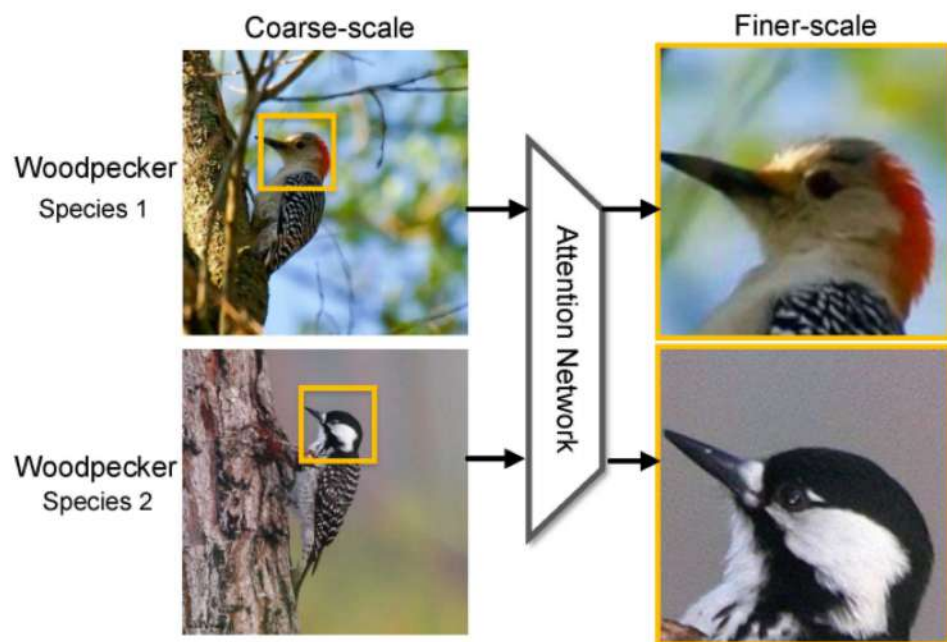
Zhichao Li, Yi Yang, Xiao Liu, Shilei Wen, Feng Zhou, Wei Xu

Baidu Research, Institute of Deep Learning (IDL)



Motivation

- Attention Models for Fine-Grained Recognition
 - Extracting discriminative regions or parts for classification
 - Constant computational complexity for high resolution images



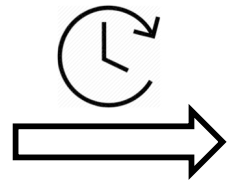
Picture from Jianlong Fu et al. 2017

Motivation

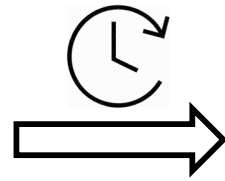
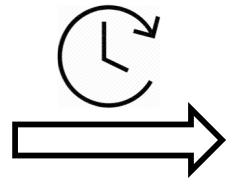
- Attention Models for Fine-Grained Recognition
 - Extracting discriminative regions or parts for classification
 - Constant computational complexity for high resolution images
- How many attentions do we need to recognize the bird?



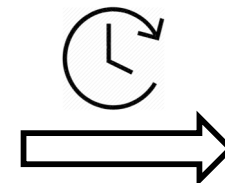
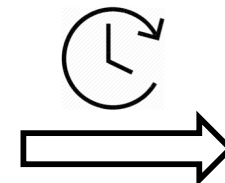
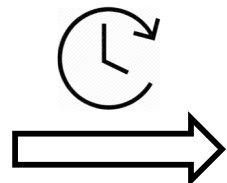
Different Inputs – Different Process Time



Tawny Owl



Tawny Owl

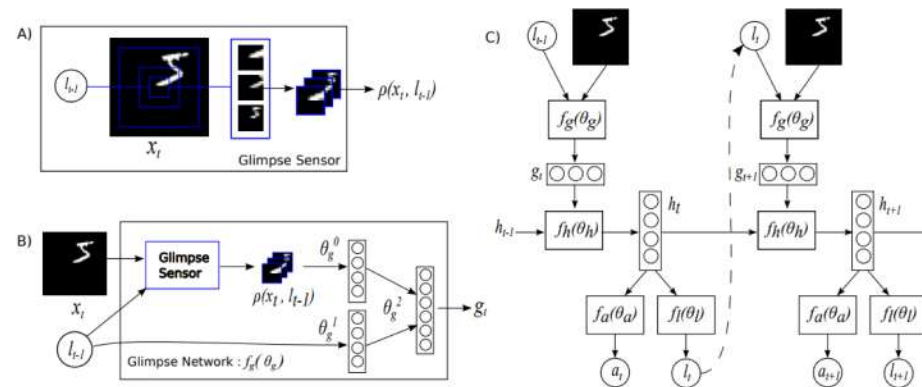


Tawny Owl

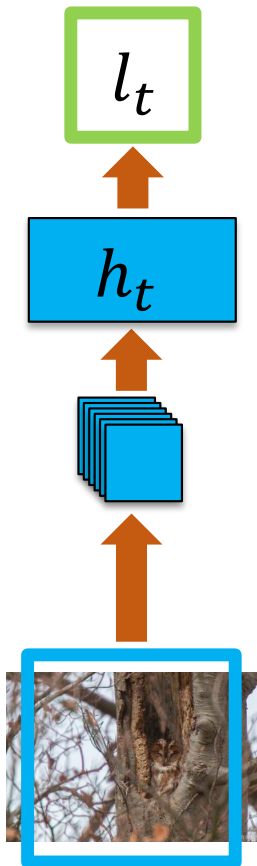
The Recurrent Attention Model (RAM)

Recurrent Models of Visual Attention

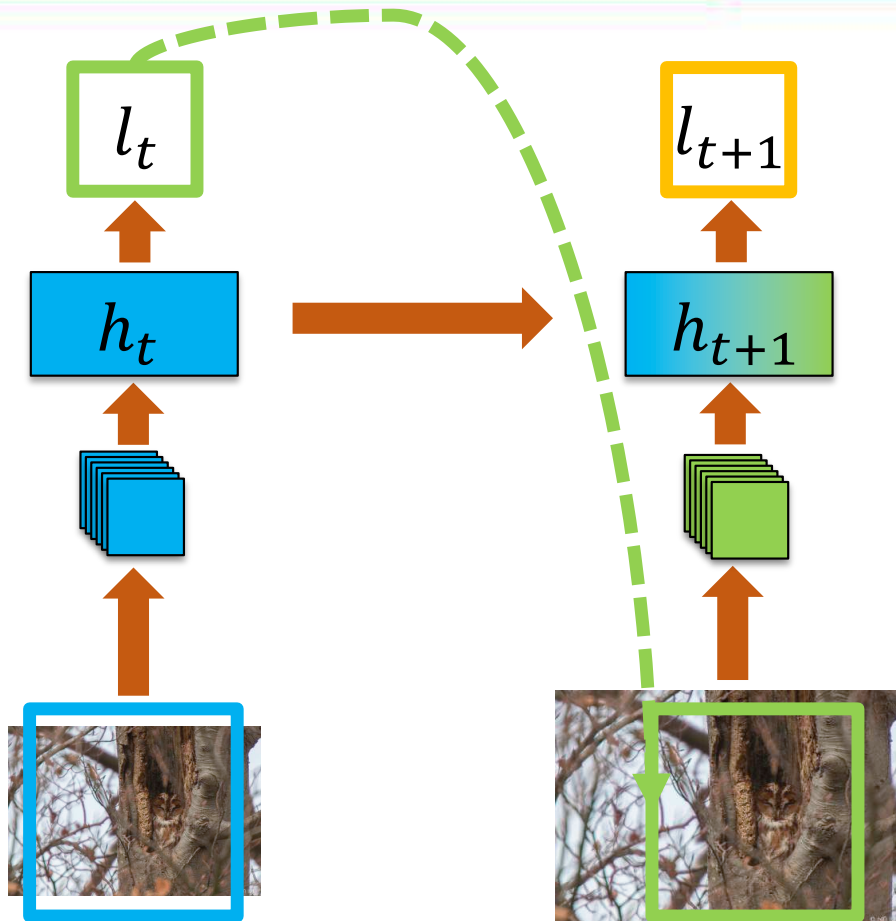
Volodymyr Mnih Nicolas Heess Alex Graves Koray Kavukcuoglu
Google DeepMind



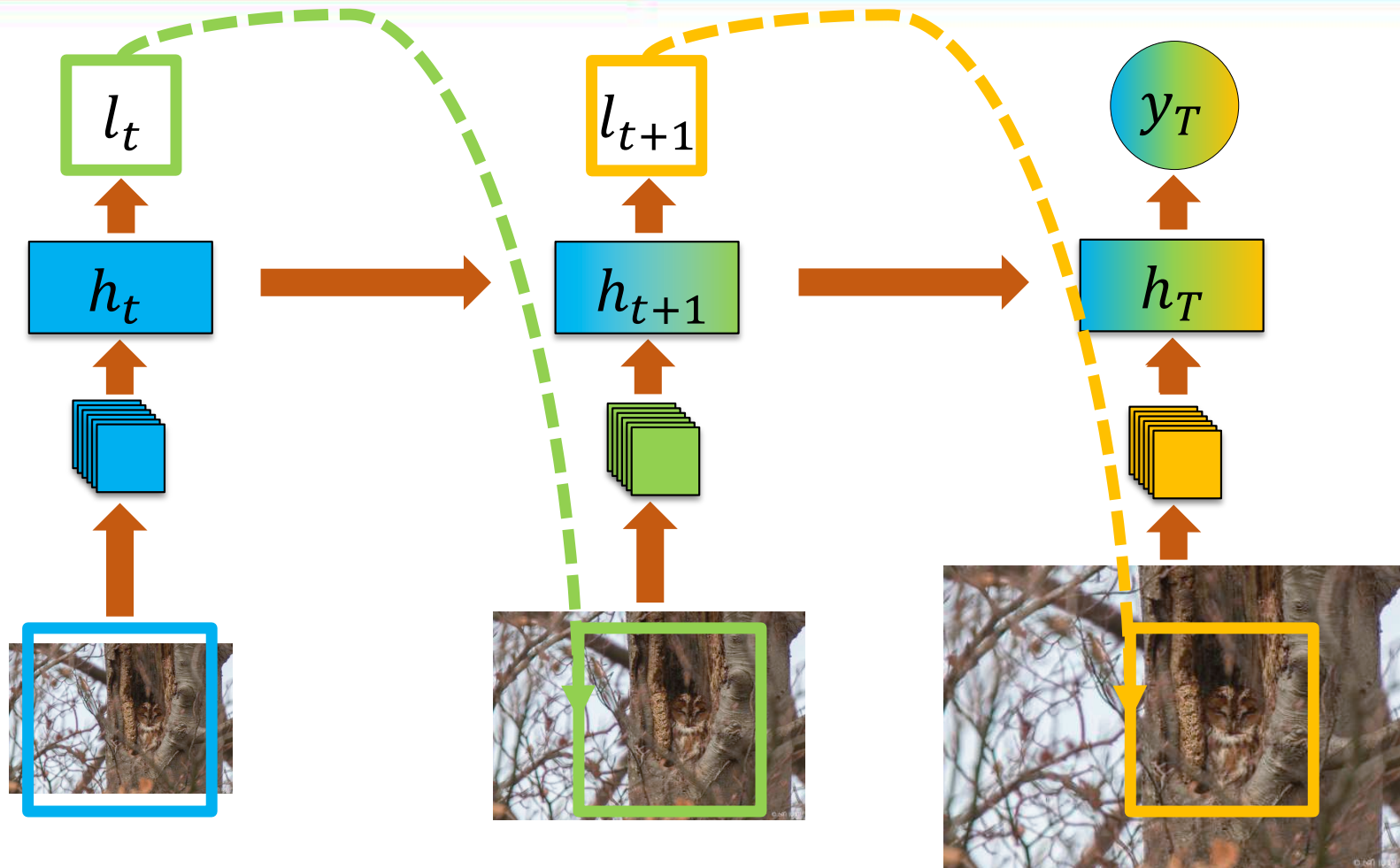
The Recurrent Attention Model (RAM)



The Recurrent Attention Model (RAM)



The Recurrent Attention Model (RAM)



The Recurrent Attention Model (RAM)

1. Feature Extraction Module:

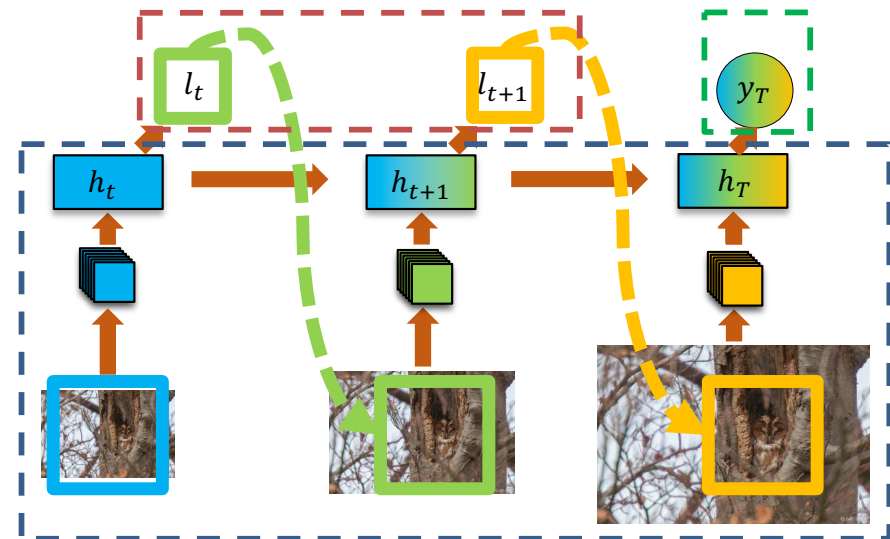
$$h_t = f_h(h_{t-1}, \phi(x, l_{t-1}), \theta_h)$$

2. Attention Module:

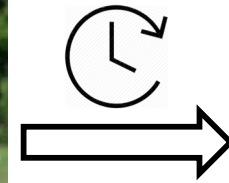
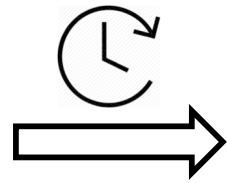
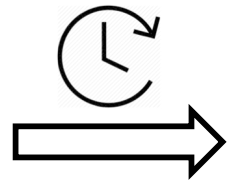
$$l_t \sim \pi(l | f_l(h_t, \theta_l))$$

3. Classification Module:

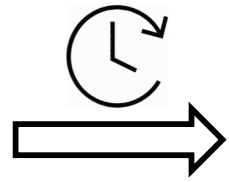
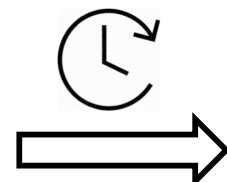
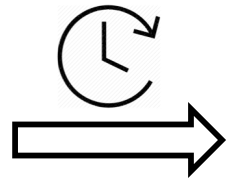
$$y_t = \arg \max_y P(y | f_c(h_t, \theta_c))$$



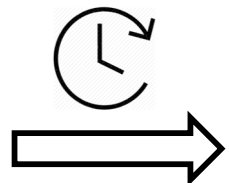
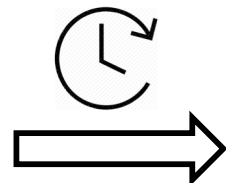
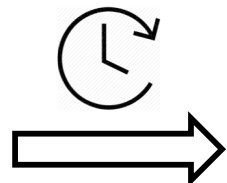
RAM as a Fixed Number of Iterations



Tawny Owl

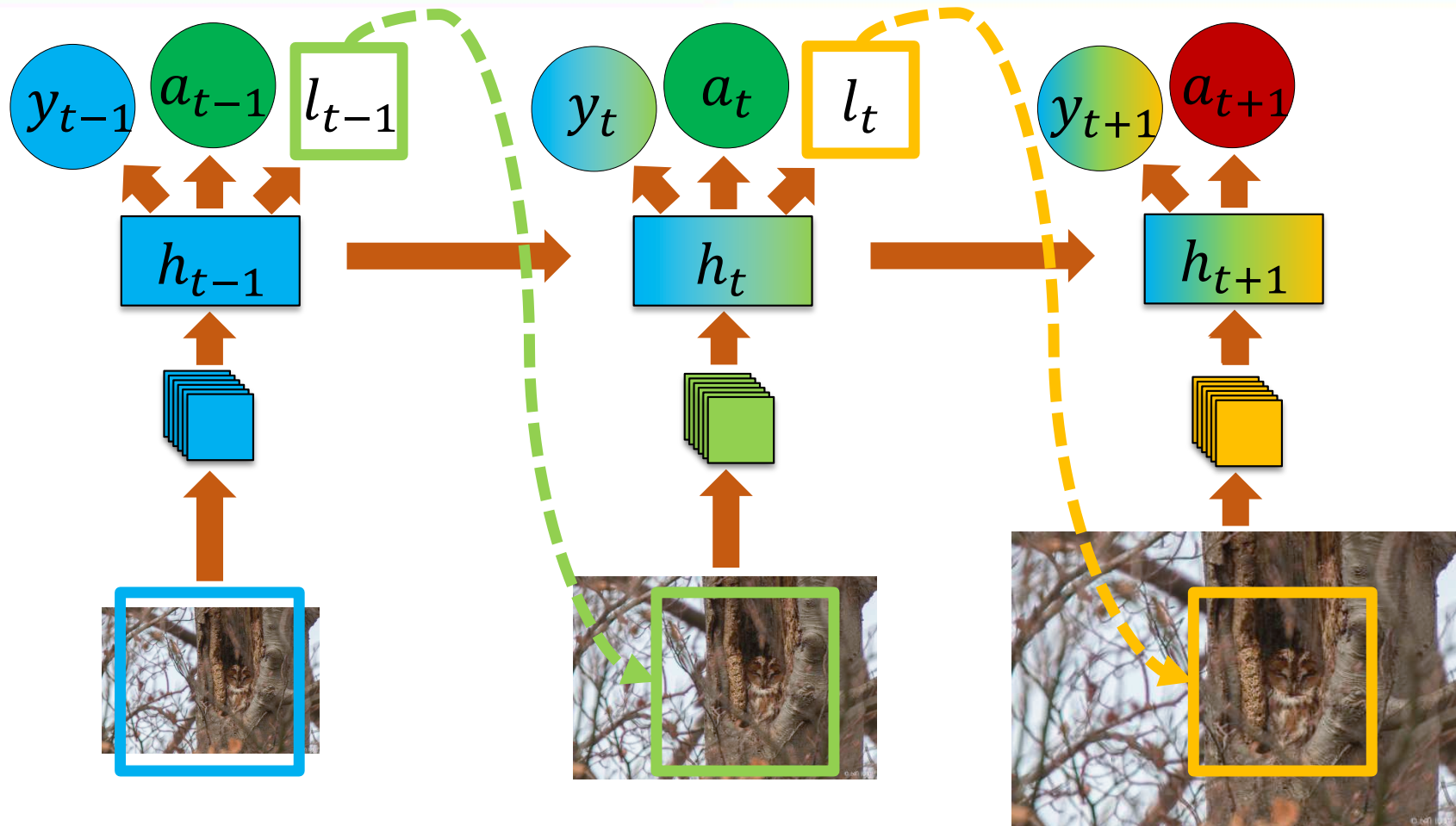


Tawny Owl



Tawny Owl

Dynamic Computational Time for RAM



Dynamic Computational Time for RAM

1. Feature Extraction Module:

$$h_t = f_h(h_{t-1}, \phi(x, l_{t-1}), \theta_h)$$

2. Classification Module:

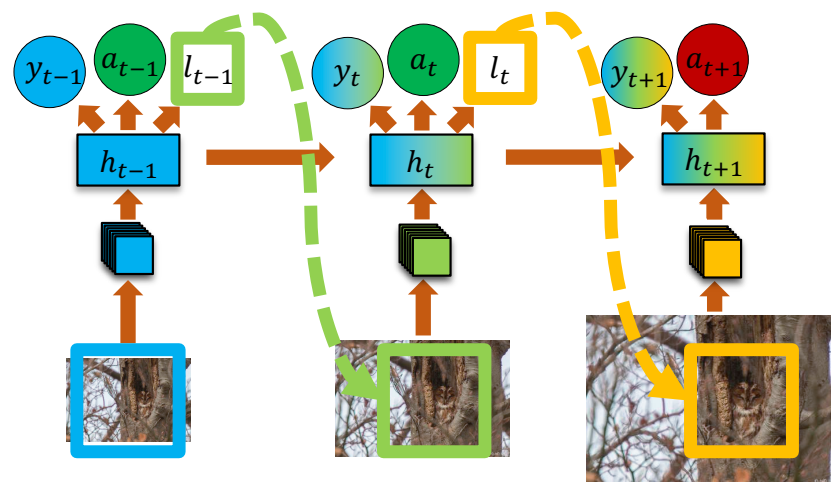
$$y_t = \arg \max_y P(y | f_c(h_t, \theta_c))$$

3. Attention Module:

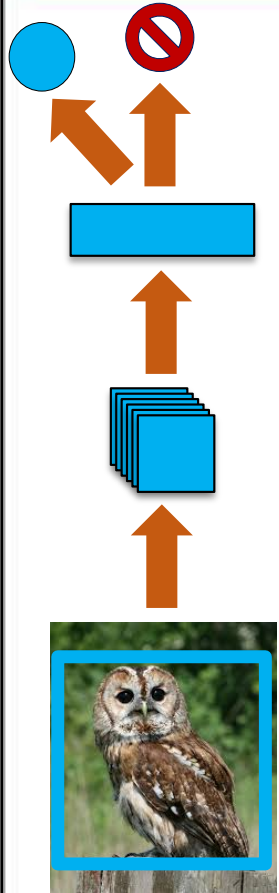
$$l_t \sim \pi(l | f_l(h_t, \theta_l))$$

4. Stopping Module:

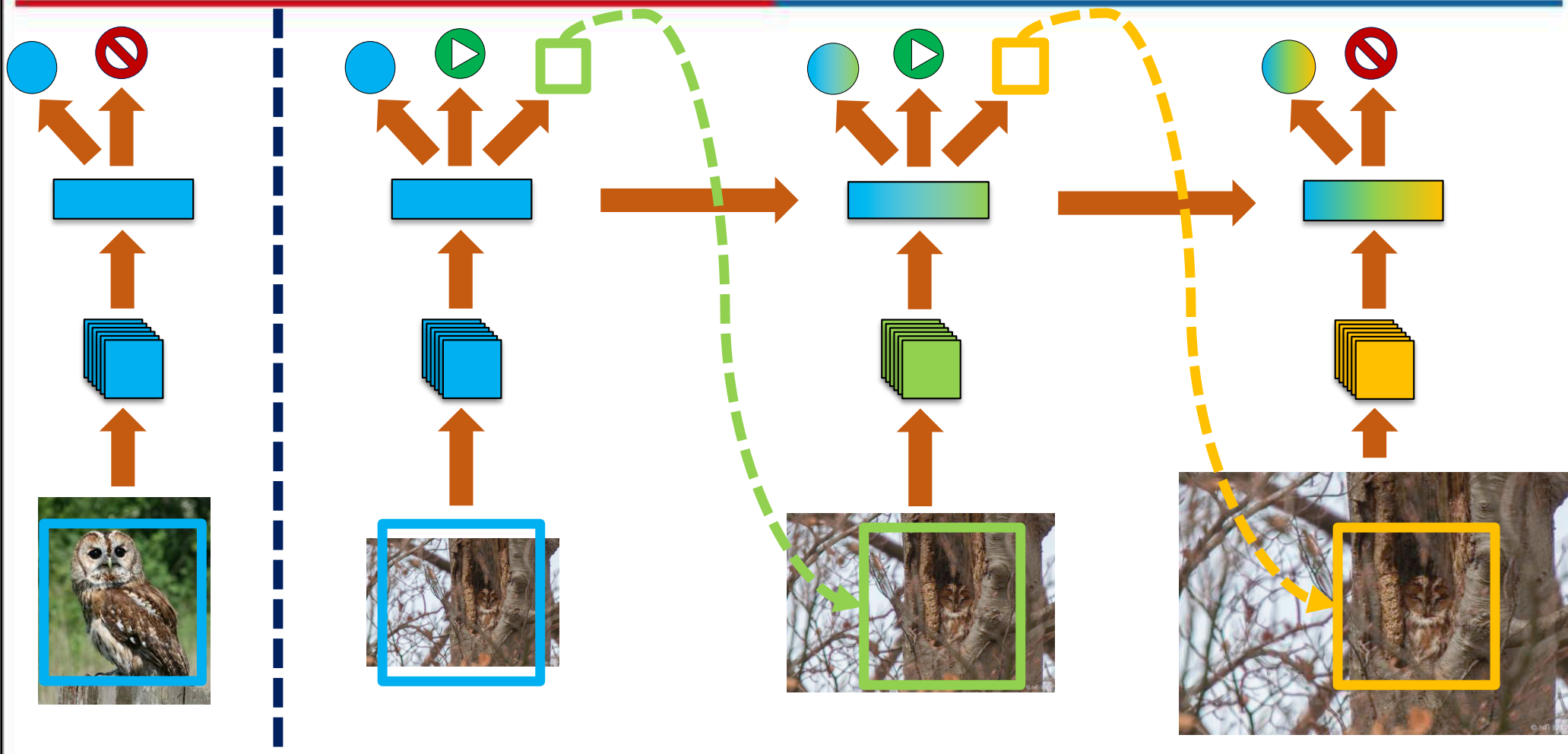
$$a_t \sim \pi(a | f_a(h_t, \theta_a))$$



Different Inputs – Different Process Time



Different Inputs – Different Process Time



Related Work

Adaptive Computation Time for Recurrent Neural Networks

Alex Graves
 Google DeepMind
 gravesa@google.com

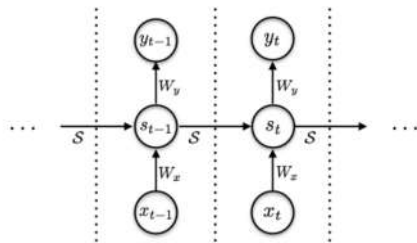


Figure 1: RNN Computation Graph. An RNN unrolled over two input steps (separated by vertical dotted lines). The input and output weights W_x, W_y , and the state transition operator S are shared over all steps.

$$N(t) = \min\{M, \min\{n' : \sum_{n=1}^{n'} h_t^n \geq 1 - \epsilon\}\}$$

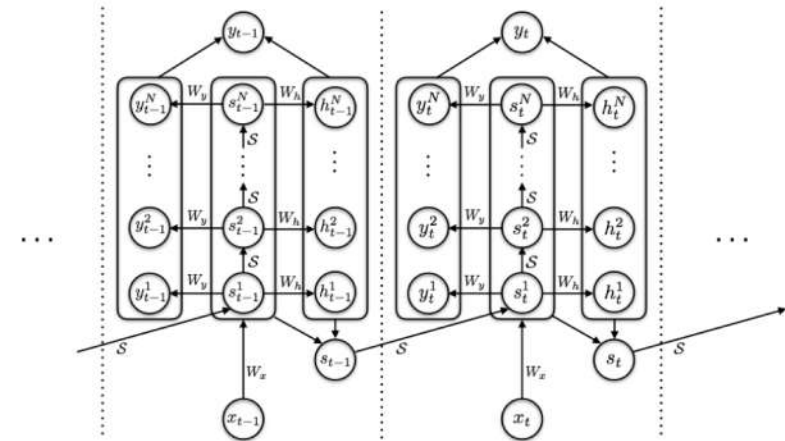


Figure 2: RNN Computation Graph with Adaptive Computation Time. The graph is equivalent to Figure 1, only with each state and output computation expanded to a variable number of intermediate updates. Arrows touching boxes denote operations applied to all units in the box, while arrows leaving boxes denote summations over all units in the box.

Related Work



Feedback Networks

Amir R. Zamir^{1,2*} Te-Lin Wu^{1*} Lin Sun¹ William Shen¹ Jitendra Malik² Silvio Savarese¹
¹ Stanford University ² University of California, Berkeley
<http://feedbacknet.stanford.edu/>

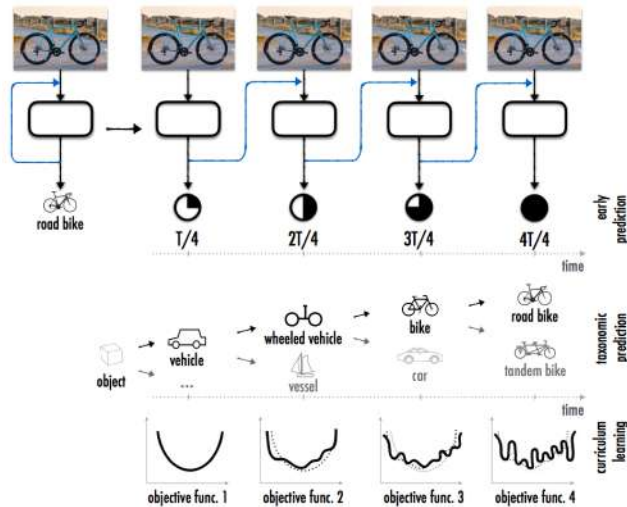


Figure 1. A feedback based learning model. The basic idea is to make predictions in an iterative manner based on a notion of the thus-far outcome. This provides several core advantages: I. enabling early predictions (given total inference time T , early predictions are made in fractions of T); II. naturally conforming to a taxonomy in the output space (one branch of the taxonomy is explored with each iteration); and III. better grounds for curriculum learning.

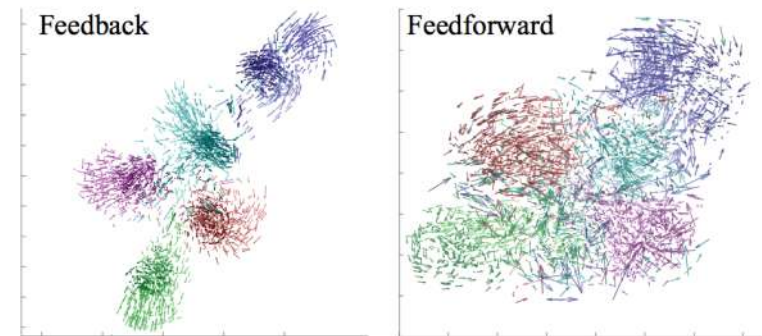


Figure 9. Timed-tSNE plots for five random CIFAR100 classes showing how the representation evolves through depth/iterations for each method (i.e. how a datapoint moved in representation space). The brighter the arrow, the earlier the depth/iteration. Feedback's representation is relatively disentangled throughout, while feedforward's representation gets disentangled only towards the end. (Best see on screen. Vector lengths are shown in half to avoid cluttering.)

Related Work

Spatially Adaptive Computation Time for Residual Networks

Michael Figurnov^{1*} Maxwell D. Collins² Yukun Zhu² Li Zhang² Jonathan Huang²

Dmitry Vetrov^{1,3} Ruslan Salakhutdinov⁴

¹Higher School of Economics ²Google Inc. ³Yandex ⁴Carnegie Mellon University

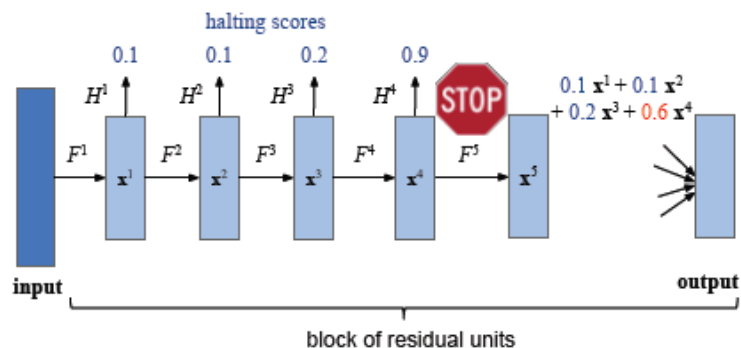


Figure 3: Adaptive Computation Time (ACT) for one block of residual units. The computation halts as soon as the cumulative sum of the halting score reaches 1. The remainder is $R = 1 - h^1 - h^2 - h^3 = 0.6$, the number of evaluated units $N = 4$, and the ponder cost is $\rho = N + R = 4.6$. See alg. 1. ACT provides a deterministic and end-to-end learnable policy of choosing the amount of computation.

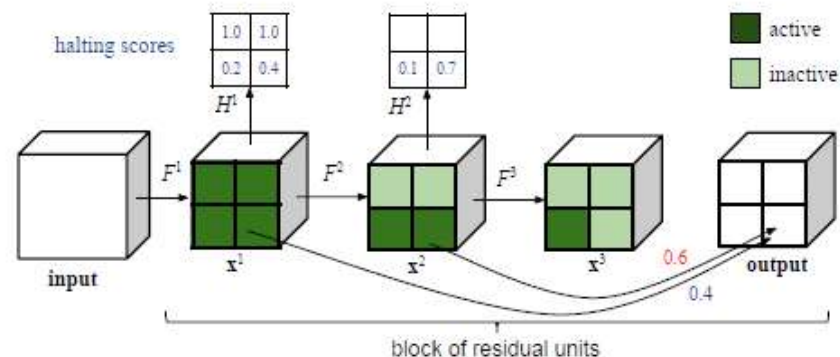
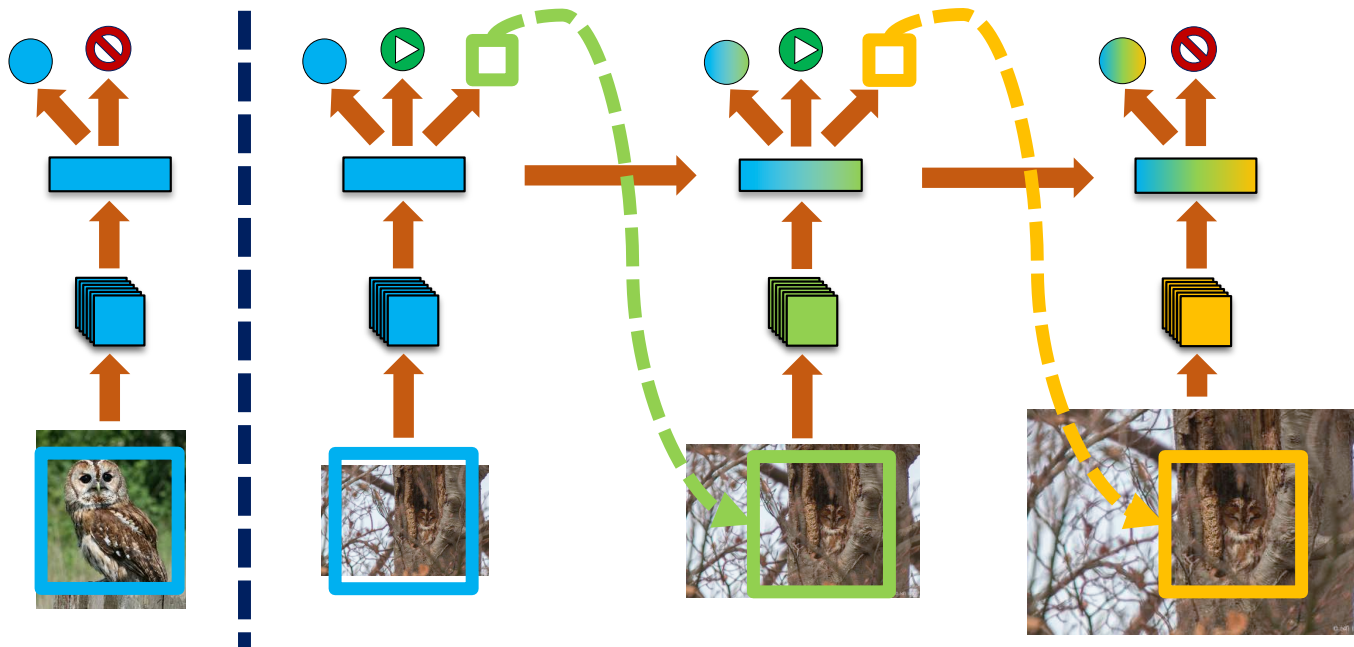


Figure 4: Spatially Adaptive Computation Time (SACT) for one block of residual units. We apply ACT to each spatial position of the block. As soon as position's cumulative halting score reaches 1, we mark it an inactive. See alg. 2. SACT learns to choose the appropriate amount of computation for each spatial position in the block.

Training – With Ground Truth Class Labels

$$\mathcal{L} = \mathbb{E}_{\mathcal{S}} [L_{\mathcal{S}}(x, \theta)] = \sum_{\mathcal{S}} P(\mathcal{S}|x, \theta) L_{\mathcal{S}}(x, \theta)$$



Training

$$\mathcal{L} = \mathbb{E}_{\mathcal{S}} [L_{\mathcal{S}}(x, \theta)] = \sum_{\mathcal{S}} P(\mathcal{S}|x, \theta) L_{\mathcal{S}}(x, \theta)$$

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \theta} &= \sum_{\mathcal{S}} \left(\frac{\partial P(\mathcal{S})}{\partial \theta} L_{\mathcal{S}} + P(\mathcal{S}) \frac{\partial L_{\mathcal{S}}}{\partial \theta} \right) \\ &= \sum_{\mathcal{S}} \left(P(\mathcal{S}) \frac{\partial \log P(\mathcal{S})}{\partial \theta} L_{\mathcal{S}} + P(\mathcal{S}) \frac{\partial L_{\mathcal{S}}}{\partial \theta} \right) \\ &= \mathbb{E}_{\mathcal{S}} \left[\frac{\partial \log P(\mathcal{S}|x, \theta)}{\partial \theta} L_{\mathcal{S}}(x, \theta) + \frac{\partial L_{\mathcal{S}}(x, \theta)}{\partial \theta} \right] \end{aligned}$$

Training (Policy Gradient)



$$\mathcal{L} = \mathbb{E}_{\mathcal{S}} [L_{\mathcal{S}}(x, \theta)] = \sum_{\mathcal{S}} P(\mathcal{S}|x, \theta) L_{\mathcal{S}}(x, \theta)$$

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial \theta} &= \sum_{\mathcal{S}} \left(\frac{\partial P(\mathcal{S})}{\partial \theta} L_{\mathcal{S}} + P(\mathcal{S}) \frac{\partial L_{\mathcal{S}}}{\partial \theta} \right) \\ &= \sum_{\mathcal{S}} \left(P(\mathcal{S}) \frac{\partial \log P(\mathcal{S})}{\partial \theta} L_{\mathcal{S}} + P(\mathcal{S}) \frac{\partial L_{\mathcal{S}}}{\partial \theta} \right) \\ &= \mathbb{E}_{\mathcal{S}} \left[\frac{\partial \log P(\mathcal{S}|x, \theta)}{\partial \theta} L_{\mathcal{S}}(x, \theta) + \frac{\partial L_{\mathcal{S}}(x, \theta)}{\partial \theta} \right] \end{aligned}$$

$$\frac{\partial \mathcal{L}}{\partial \theta} \approx \frac{1}{M} \sum_{i=1}^M \left(\frac{\partial \log P(\mathcal{S}_i|x, \theta)}{\partial \theta} L_{\mathcal{S}_i}(x, \theta) + \frac{\partial L_{\mathcal{S}_i}(x, \theta)}{\partial \theta} \right)$$

More Formally



Use reward to replace loss:

$$\frac{\partial \mathcal{L}}{\partial \theta} \approx \sum_n \sum_{\mathcal{S}} \left(-\frac{\partial \log P(\mathcal{S}|x_n, \theta)}{\partial \theta} R_n + \frac{\partial L_{\mathcal{S}}(x_n, y_n, \theta)}{\partial \theta} \right)$$

Sample attention and stopping:

$$P(\mathcal{S}|x_n, \theta) = \prod_{t=1}^{T(n)} \pi(l_t | f_l(h_t, \theta_l)) \pi(a_t | f_a(h_t, \theta_a))$$

Discounted cumulative reward:

$$R_n = \sum_{t=1}^{T(n)} \gamma^t r_{nt}$$

Intermediate supervision:

$$L_{\mathcal{S}}(x_n, y_n, \theta) = \sum_{t=1}^{T(n)} L_t(x_n, y_n, \theta_h, \theta_c)$$

Discounted Reward for Early Stopping



- The discounted reward is designed for early stopping
- The discount factor γ controls the trade-off between accuracy and computational complexity

Discounted cumulative reward:

$$R_n = \sum_{t=1}^{T(n)} \gamma^t r_{nt}$$

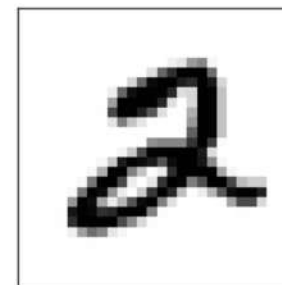
Immediate reward:

$$r_{nt} = \begin{cases} 1, & \text{if } y_{nt} = y_n^* \text{ and } t = T(n) \\ 0, & \text{otherwise} \end{cases}$$

Experiments: Fine-Grained Recognition

Dataset	#Classes	#Train	#Test	BBox
MNIST [14]	10	60000	10000	-
CUB-200-2011 [15]	200	5994	5794	✓
Stanford Cars [16]	196	8144	8041	✓

Table 1. Statistics of the three dataset. CUB-200-2011 and Stanford Cars are both benchmark datasets in fine-grained recognition.

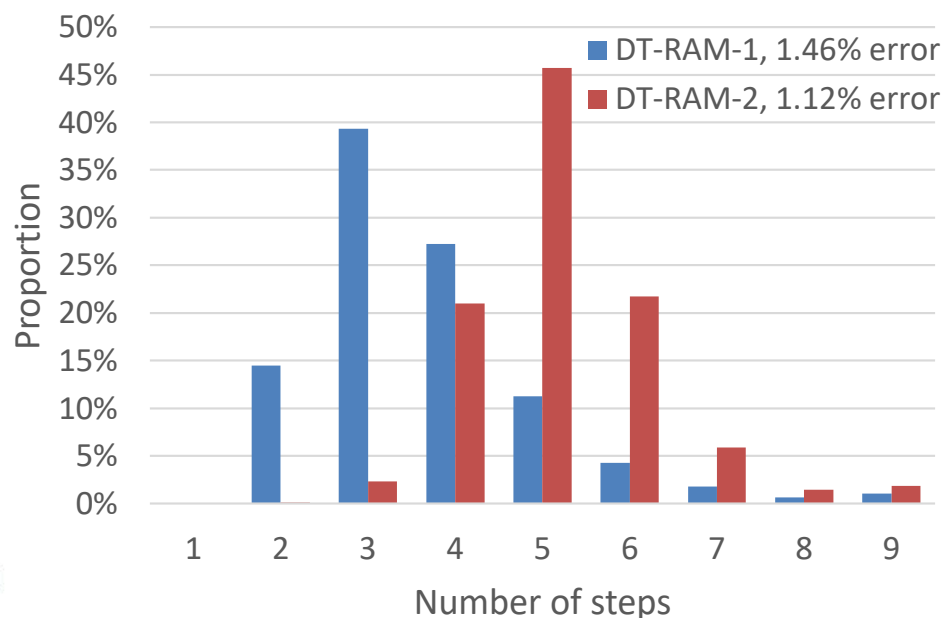


Experiments on MNIST

- Image Resolution: 28x28, Crop Resolution: 8x8

MNIST	# Steps	Error(%)
FC, 2 layers (256 hidden each)	-	1.69
Convolutional, 2 layers	-	1.21
RAM 2 steps	2	3.79
RAM 4 steps	4	1.54
RAM 5 steps	5	1.34
RAM 7 steps	7	1.07
DT-RAM-1 3.6 steps	3.6	1.46
DT-RAM-2 5.2 steps	5.2	1.12

Table 2. Comparison to related work on MNIST. All the RAM results are from [6].



Experiments on CUB-Birds and Cars



- Baseline: Residual Net 50 pre-trained on ImageNet
- Image Resolution: 512x512, Crop Resolution: 224x224

Experiments on CUB-Birds and Cars



- Baseline: Residual Net 50 pre-trained on ImageNet
- Image Resolution: 512x512, Crop Resolution: 224x224

CUB-200-2011	Accuracy(%)	Acc w. Box(%)
Zhang <i>et al.</i> [63]	73.9	76.4
Branson <i>et al.</i> [56]	75.7	85.4*
Simon <i>et al.</i> [64]	81.0	-
Krause <i>et al.</i> [48]	82.0	82.8
Lin <i>et al.</i> [54]	84.1	85.1
Jaderberg <i>et al.</i> [59]	84.1	-
Kong <i>et al.</i> [53]	84.2	-
Liu <i>et al.</i> [62]	84.3	84.7
Liu <i>et al.</i> [9]	85.4	85.5
ResNet-50 [23]	84.5	-
RAM 3 steps	86.0	-
DT-RAM 1.9 steps	86.0	-

Table 3. Comparison to related work on CUB-200-2011 dataset. * Testing with both ground truth box and parts.

Stanford Cars	Accuracy(%)	Acc w. Box(%)
Chai <i>et al.</i> [65]	78.0	-
Gosselin <i>et al.</i> [66]	82.7	87.9
Girshick <i>et al.</i> [67]	88.4	-
Lin <i>et al.</i> [54]	91.3	-
Wang <i>et al.</i> [68]	-	92.5
Liu <i>et al.</i> [62]	91.5	93.1
Krause <i>et al.</i> [48]	92.6	92.8
ResNet-50 [23]	92.3	-
RAM 3 steps	93.1	-
DT-RAM 1.9 steps	93.1	-

Table 4. Comparison to related work on Stanford Cars dataset.

Experiments on CUB-Birds and Cars



- Baseline: Residual Net 50 pre-trained on ImageNet
- Image Resolution: 512x512, Crop Resolution: 224x224

CUB-200-2011	Accuracy(%)	Acc w. Box(%)
Zhang <i>et al.</i> [63]	73.9	76.4
Branson <i>et al.</i> [56]	75.7	85.4*
Simon <i>et al.</i> [64]	81.0	-
Krause <i>et al.</i> [48]	82.0	82.8
Lin <i>et al.</i> [54]	84.1	85.1
Jaderberg <i>et al.</i> [59]	84.1	-
Kong <i>et al.</i> [53]	84.2	-
Liu <i>et al.</i> [62]	84.3	84.7
Liu <i>et al.</i> [9]	85.4	85.5
ResNet-50 [23]	84.5	-
RAM 3 steps	86.0	-
DT-RAM 1.9 steps	86.0	-

Table 3. Comparison to related work on CUB-200-2011 dataset. * Testing with both ground truth box and parts.

Stanford Cars	Accuracy(%)	Acc w. Box(%)
Chai <i>et al.</i> [65]	78.0	-
Gosselin <i>et al.</i> [66]	82.7	87.9
Girshick <i>et al.</i> [67]	88.4	-
Lin <i>et al.</i> [54]	91.3	-
Wang <i>et al.</i> [68]	-	92.5
Liu <i>et al.</i> [62]	91.5	93.1
Krause <i>et al.</i> [48]	92.6	92.8
ResNet-50 [23]	92.3	-
RAM 3 steps	93.1	-
DT-RAM 1.9 steps	93.1	-

Table 4. Comparison to related work on Stanford Cars dataset.

Qualitative Results

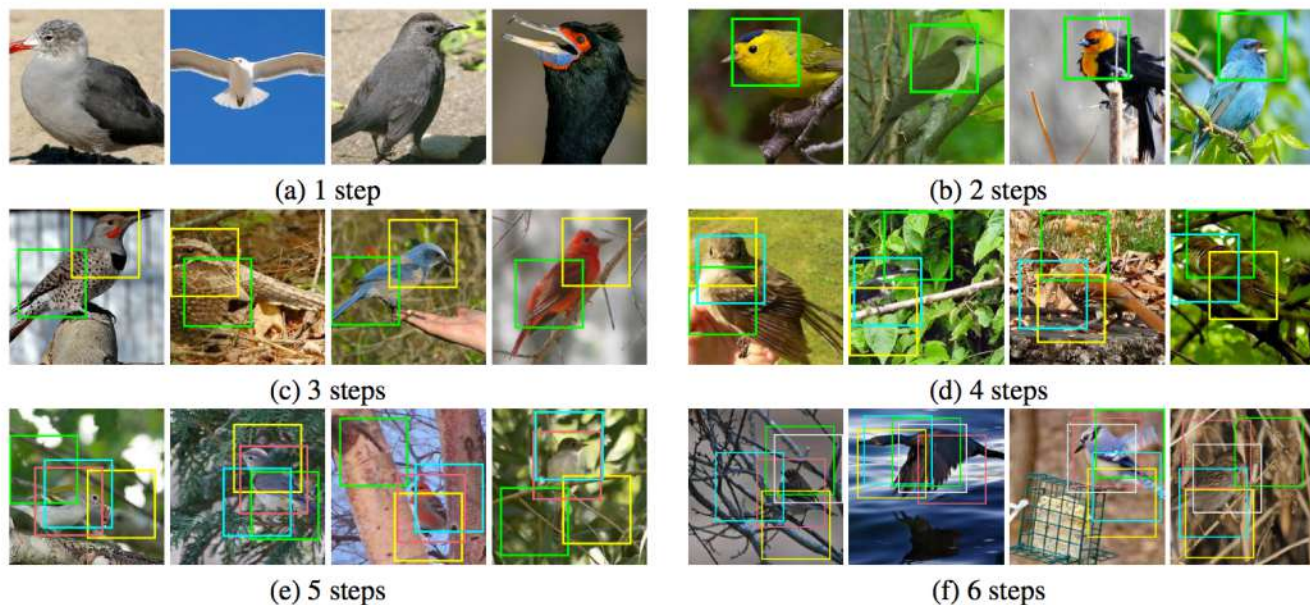


Figure 7. Qualitative results of DT-RAM on CUB-200-2011 *testing* set. We show images with different ending steps from 1 to 6. Each bounding box indicates an attention region. Bounding box colors are displayed in order. The first step uses the full image as input hence there is no bounding box. From step 1 to step 6, we observe a gradual increase of background clutter and recognition difficulty, matching our hypothesis for using dynamic computation time for different types of images.

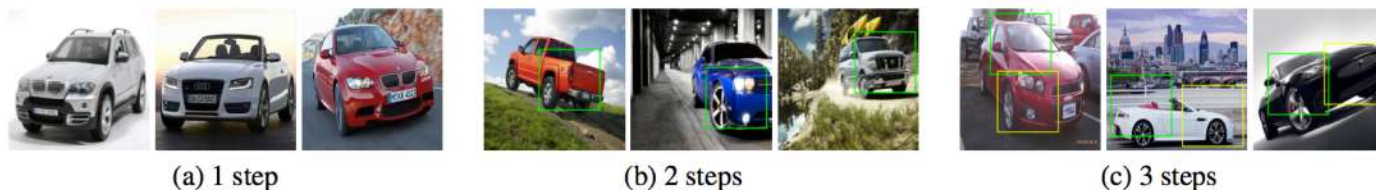


Figure 8. Qualitative results of DT-RAM on Stanford Car *testing* set. We only manage to train a 3-step model with 512×512 resolution.

Diagnostic Experiments (On CUB-Birds)



- Baseline: Residual Net 34 pre-trained on ImageNet
- Image Resolution: 256x256, Crop Resolution: 100x100

Model	# Steps	Accuracy(%)
ResNet-34	1	79.9
RAM 2 steps	2	80.7
RAM 3 steps	3	81.1
RAM 4 steps	4	81.5
RAM 5 steps	5	81.8
RAM 6 steps	6	81.8
DT-RAM (6 max steps)	3.6	81.8

Table 6. Comparison to RAM on CUB-200-2011. Note that the 1-step RAM is the same as the ResNet.

Diagnostic Experiments (On CUB-Birds)



- Baseline: Residual Net 34 pre-trained on ImageNet
- Image Resolution: 256x256, Crop Resolution: 100x100

Model	# Steps	Accuracy(%)
ResNet-34	1	79.9
RAM 2 steps	2	80.7
RAM 3 steps	3	81.1
RAM 4 steps	4	81.5
RAM 5 steps	5	81.8
RAM 6 steps	6	81.8
DT-RAM (6 max steps)	3.6	81.8

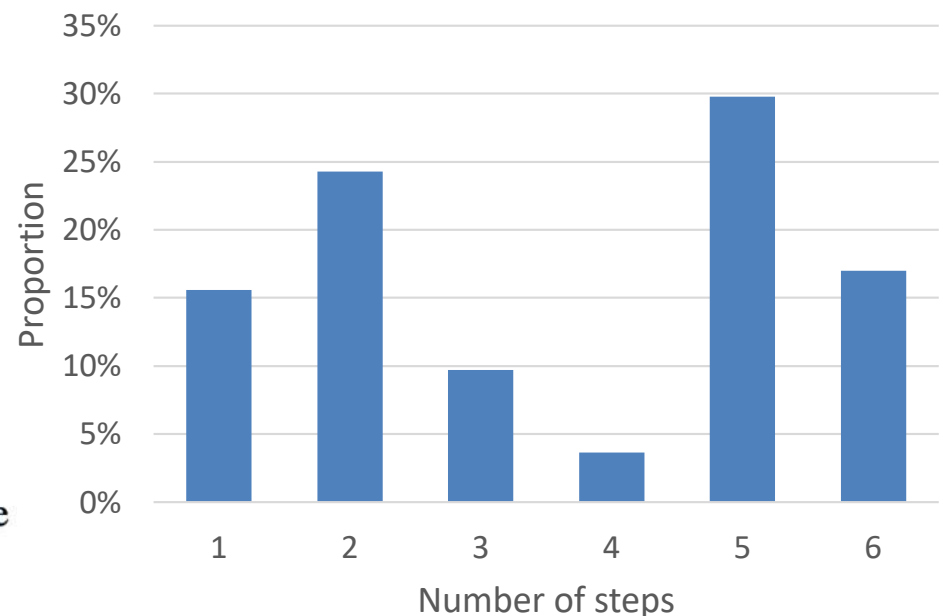


Table 6. Comparison to RAM on CUB-200-2011. Note that the 1-step RAM is the same as the ResNet.

Diagnostic Experiments (On CUB-Birds)



- How does it compare against a fixed policy?
- Fixed policy: Stop if one class confidence is above threshold.

Threshold	# Steps	Accuracy(%)
0	1	79.9
0.4	1.4	80.7
0.5	1.6	81.0
0.6	1.9	81.2
0.9	3.6	81.3
1.0	6	81.8
DT-RAM (6 max steps)	3.6	81.8

Table 7. Comparison to a fixed stopping policy on CUB-200-2011. The fixed stopping policy runs on RAM (6 steps) such that the recurrent attention stops if one of the class softmax probabilities is above the threshold.

Diagnostic Experiments (On CUB-Birds)



How does Resolution and Depth affect:

Resolution	ResNet-34	RAM-34	Resnet-50	RAM-50
224×224	79.9	81.8	81.5	82.8
448×448	-	-	84.5	86.0

Table 5. The effect of input resolution and network depth on ResNet and its RAM extension.

How does Curriculum Learning affect:

# Steps	1	2	3	4	5	6
w.o C.L.	79.9	80.7	80.5	80.9	80.3	80.0
w. C.L.	79.9	80.7	81.1	81.5	81.8	81.8

Table 8. The effect of Curriculum Learning on RAM.

How does Intermediate Supervision affect:

# Steps	1	2	3	4	5	6
w.o I.S.	79.9	78.8	76.1	74.8	74.9	74.7
w. I.S.	79.9	80.7	81.1	81.5	81.8	81.8

Table 9. The effect of Intermediate Supervision on RAM.

Take Home Message



- Residual net is a **STRONG** baseline for fine-grained
 - On CUB-Bird 2011: 84.5%, On Stanford Cars: 92.3%

Take Home Message



- Residual net is a **STRONG** baseline for fine-grained
 - On CUB-Bird 2011: 84.5%, On Stanford Cars: 92.3%
- Attention model reaches new state-of-the-art
 - Bird: 84.5% -> 86.0%
 - Car: 92.3% -> 93.1%

Take Home Message



- Residual net is a **STRONG** baseline for fine-grained
 - On CUB-Bird 2011: 84.5%, On Stanford Cars: 92.3%
- Attention model reaches new state-of-the-art
 - Bird: 84.5% -> 86.0%
 - Car: 92.3% -> 93.1%
- Dynamic Time seems a worth idea:
 - DT-RAM: 1.9 steps ~ RAM: 3 steps

Take Home Message



- Carefully tuned Residual Net
 - Scale augmentation (~1.2% improve in ImageNet)
 - Where to put ReLU and BN (~0.6% improved in CIFAR)
 - Strided convolution(~0.3% improved in ImageNet)
 - smoothing factor in BN (~0.2% improved in ImageNet)
 - Color augmentation(slightly improved)
 - Weight decay

Note : all improved base in resnet-50

Thank you!

- Code Available:
 - <https://github.com/baidu-research/DT-RAM>
 - Written in Torch

