

Dynamic Computational Time for Visual Attention

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

{lizhichao01, yangyi05, liuxiao12, zhoufeng09, wenshilei, xuwei06}@baidu.com

Abstract

We propose a dynamic computational time model to accelerate the average processing time for recurrent visual attention (RAM). Rather than attention with a fixed number of steps for each input image, the model learns to decide when to stop on the fly. To achieve this, we add an additional continue/stop action per time step to RAM and use reinforcement learning to learn both the optimal attention policy and stopping policy. The modification is simple but could dramatically save the average computational time while keeping the same recognition performance as RAM. Experimental results on CUB-200-2011 and Stanford Cars dataset demonstrate the dynamic computational model can work effectively for fine-grained image recognition. The source code of this paper can be obtained from <https://github.com/baidu-research/DT-RAM>

1. Introduction

Human have the remarkable ability of selective visual attention [1, 2]. Cognitive science explains this as the “Biased Competition Theory” [3, 4] that human visual cortex is enhanced by top-down guidance during feedback loops. The feedback signals suppress non-relevant stimuli present in the visual field, helping human searching for “goals”. With visual attention, both human recognition and detection performances increase significantly, especially in images with cluttered background [5].

Inspired by human attention, the Recurrent Visual Attention Model (RAM) is proposed for image recognition [6]. RAM is a deep recurrent neural architecture with iterative attention selection mechanism, that mimics the human visual system to suppress non-relevant image regions and extract discriminative features in a complicated environment. This significantly improves the recognition accuracy [7], especially for fine-grained object recognition [8, 9]. RAM also allows the network to process a high resolution image with only limited computational resources. By iteratively attending to different sub-regions (with a fixed resolution), RAM could efficiently process images with various

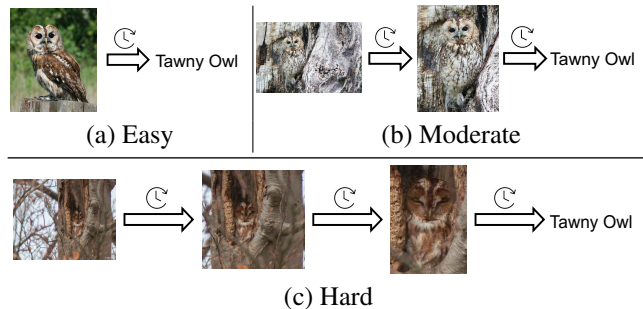


Figure 1. We show the recognition process of 3 tawny owl images with increasing level of difficulty for recognition. When recognizing the same object in different images, human may spend different length of time.

resolutions and aspect ratios in a constant computational time [6, 7].

Besides attention, human also tend to dynamically allocate different computational time when processing different images [5, 10]. The length of the processing time often depends on the task and the content of the input images (e.g. background clutter, occlusion, object scale). For example, during the recognition of a fine-grained bird category, if the bird appears in a large proportion with clean background (Figure 1a), human can immediately recognize the image without hesitation. However, when the bird is under camouflage (Figure 1b) or hiding in the scene with background clutter and pose variation (Figure 1c), people may spend much more time on locating the bird and extracting discriminative parts to produce a confident prediction.

Inspired by this, we propose an extension to RAM named as Dynamic Time Recurrent Attention Model (DT-RAM), by adding an extra binary (continue/stop) action at every time step. During each step, DT-RAM will not only update the next attention, but produce a decision whether stop the computation and output the classification score. The model is a simple extension to RAM, but can be viewed as a first step towards dynamic model during inference [11], where the model structure can vary based on each input instance. This could bring DT-RAM more flexibility and reduce re-

dundant computation to further save computation, especially when the input examples are “easy” to recognize.

Although DT-RAM is an end-to-end recurrent neural architecture, we find it hard to directly train the model parameters from scratch, particularly for challenging tasks like fine-grained recognition. When the total number of steps increases, the delayed reward issue becomes more severe and the variance of gradients becomes larger. This makes policy gradient training algorithms such as REINFORCE [12] harder to optimize. We address this problem with curriculum learning [13]. During the training of RAM, we gradually increase the training difficulty by gradually increasing the total number of time steps. We then initialize the parameters in DT-RAM with the pre-trained RAM and fine-tune it with REINFORCE. This strategy helps the model to converge to a better local optimum than training from scratch. We also find intermediate supervision is crucial to the performance, particularly when training longer sequences.

We demonstrate the effectiveness of our model on public benchmark datasets including MNIST [14] as well as two fine-grained datasets, CUB-200-2011 [15] and Stanford Cars [16]. We also conduct an extensive study to understand how dynamic time works in these datasets. Experimental results suggest that DT-RAM can achieve state-of-the-art performance on fine-grained image recognition. Compared to RAM, the model also uses less average computational time, better fitting devices with computational limitations.

2. Related Work

2.1. Visual Attention Models

Visual attention is a long-standing topic in computer vision [17, 18, 19]. With the recent success of deep neural networks [20, 21, 22, 23], Mnih *et al.* [6] develop the Recurrent Visual Attention Model (RAM) for image recognition, where the attention is modeled with neural networks to capture local regions in the image. Ba *et al.* [7] follow the same framework and apply RAM to recognize multiple objects in images. Sermanet *et al.* [8] further extend RAM to fine-grained image recognition, since fine-grained problems usually require the comparison between local parts. Besides fine-grained recognition, attention models also work for various machine learning problems including machine translation [24], image captioning [25], image question answering [26, 27, 28, 29] and video activity recognition [30].

Based on the differentiable property of attention models, most of the existing work can be divided into two groups: soft attention and hard attention [25]. The soft attention models define attention as a set of continuous variables representing the relative importance of spatial or temporal cues. The model is differentiable hence can be trained with backpropagation. The hard attention models define attention as actions and model the whole problem as a Partially

Observed Markov Decision Process (POMDP) [31]. Such models are usually nondifferentiable to the reward function hence use policy gradient such as REINFORCE [12] to optimize the model parameters. Our model belongs to the hard attention since its stopping action is discrete.

2.2. Feedback Neural Networks

The visual attention models can be also viewed as a special type of feedback neural networks [32, 33, 34, 35]. A feedback neural network is a special recurrent architecture that uses previously computed high level features to back refine low level features. It uses both top-down and bottom-up information to compute the intermediate layers. Besides attention models, feedback neural networks also have other variants. For example, Carreira *et al.* [36] performs human pose estimation with iterative error feedback. Newell *et al.* [37] build a stacked hourglass network for human pose estimation. Hu and Ramanan [38] show that network feedbacks can help better locating human face landmarks. All these models demonstrate top-down information could potentially improve the model discriminative ability [32]. However, these models either fix the number of recurrent steps or use simple rules to decide early stopping.

2.3. Dynamic Computational Time

Graves [11] recently introduce *adaptive computational time* in recurrent neural networks. The model augments the network with a *sigmoidal halting unit* at each time step, whose activation determines the probability whether the computation should stop. Figurnov *et al.* [39] extend [11] to spatially adaptive computational time for residual networks. Their approach is similar but define the *halting units* over spatial positions. Neumann *et al.* [40] extend the similar idea to temporally dependent reasoning. They achieve a small performance benefit on top of a similar model without an adaptive component. Jernite *et al.* [41] learn a scheduler to determine what portion of the hidden state to compute based on the current hidden and input vectors. All these models can vary the computation time during inference, but the stopping policy is based on the cumulative probability of *halting units*, which can be viewed as a fixed policy.

As far as we know, Odena *et al.* [42] is the first attempt that learn to change model behavior at test time with reinforcement learning. Their model adaptively constructs computational graphs from sub-modules on a per-input basis. However, they only verify on small dataset such as MNIST [14] and CIFAR-10 [43]. Ba *et al.* [7] augment RAM with the “end-of-sequence” symbol to deal with variable number of objects in an image, which inspires our work on DT-RAM. However, they still fix the number of attentions for each target. There is also a lack of diagnostic experiments on understanding how “end-of-sequence” symbol affects the dynamics. In this work, we conduct extensive ex-

perimental comparisons on larger scale natural images from fine-grained recognition.

2.4. Fine-Grained Recognition

Fine-grained image recognition has been extensively studied in recent years [44, 45, 46, 47, 48, 16, 49, 50, 51]. Based on the research focus, fine-grained recognition approaches can be divided into representation learning, part alignment models or emphasis on data. The first group attempts to build implicitly powerful feature representations such as bilinear pooling or compact bilinear pooling [52, 53, 54], which turn to be very effective for fine-grained problems. The second group attempts to localize discriminative parts to effectively deal with large intra-class variation as well as subtle inter-class variation [55, 56, 57, 47, 9]. The third group studies the importance of the scale of training data [58]. They achieve significantly better performance on multiple fine-grained dataset by using an extra large set of training images.

With the fast development of deep models such as Bilinear CNN [54] and Spatial Transformer Networks [59], it is unclear whether attention models are still effective for fine-grained recognition. In this paper, we show that the visual attention model, if trained carefully, can still achieve comparable performance as state-of-the-art methods.

3. Model

3.1. Learning with Dynamic Structure

The difference between a dynamic structure model and a fixed structure model is that during inference the model structure \mathcal{S} depends on both the input x and parameter θ .

Given an input x , the probability of choosing a computational structure \mathcal{S} is $P(\mathcal{S}|x, \theta)$. When the model space of \mathcal{S} is defined, this probability can be modeled with a neural network. During training, with a given model structure, the loss is $L_{\mathcal{S}}(x, \theta)$. Hence the overall expected loss for an input x is

$$\mathcal{L} = \mathbb{E}_{\mathcal{S}} [L_{\mathcal{S}}(x, \theta)] = \sum_{\mathcal{S}} P(\mathcal{S}|x, \theta) L_{\mathcal{S}}(x, \theta) \quad (1)$$

The gradient of \mathcal{L} with respect to parameter θ is:

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

The first term in the above expectation is the same as REINFORCE algorithm [12], it makes the structure leading to

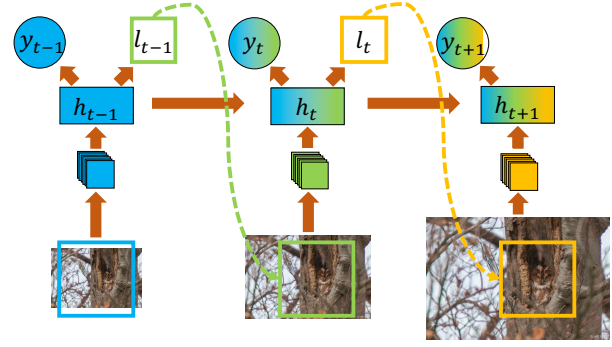


Figure 2. An illustration of the architecture of recurrent attention model. By iteratively attending to more discriminative area l_t , the model could output more confident predictions y_t .

smaller loss more probable. The second term is the standard gradient for neural nets with a fixed structure.

During experiments, it is difficult to directly compute the gradient of the \mathcal{L} over θ because it requires to evaluate exponentially many possible structures during training. Hence to train the model, we first sample a set of structures, then approximate the gradient with Monte Carlo Simulation [12]:

$$\frac{\partial \mathcal{L}}{\partial \theta} \approx \frac{1}{M} \sum_{i=1}^M \left(\frac{\partial \log P(\mathcal{S}_i|x, \theta)}{\partial \theta} L_{\mathcal{S}_i}(x, \theta) + \frac{\partial L_{\mathcal{S}_i}(x, \theta)}{\partial \theta} \right) \quad (2)$$

3.2. Recurrent Attention Model (RAM)

The recurrent attention model is formulated as a Partially Observed Markov Decision Process (POMDP). At each time step, the model works as an agent that executes an action based on the observation and receives a reward. The agent actively control how to act, and it may affect the state of the environment. In RAM, the action corresponds to the location of the attention region. The observation is a local (partially observed) region cropped from the image. The reward measures the quality of the prediction using all the cropped regions and can be delayed. The target of learning is to find the optimal decision policy to generate attentions from observations that maximizes the expected cumulative reward across all time steps.

More formally, RAM defines the input image as x and the total number of attentions as T . At each time step $t \in \{1, \dots, T\}$, the model crops a local region $\phi(x, l_{t-1})$ around location l_{t-1} which is computed from the previous time step. It then updates the internal state h_t with a recurrent neural network

$$h_t = f_h(h_{t-1}, \phi(x, l_{t-1}), \theta_h) \quad (3)$$

which is parameterized by θ_h . The model then computes two branches. One is the location network $f_l(h_t, \theta_l)$ which

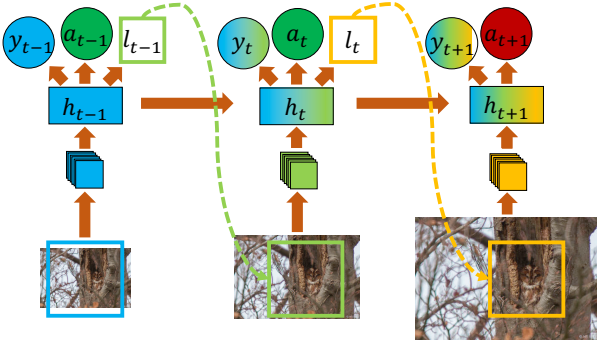


Figure 3. An illustration of the architecture of dynamic time recurrent attention model. An extra binary stopping action a_t is added to each time step. $a_t = 0$ represents “continue” (green solid circle) and $a_t = 1$ represents “stop” (red solid circle).

models the attention policy, parameterized by θ_l . The other is the classification network $f_c(h_t, \theta_c)$ which computes the classification score, parameterized by θ_c . During inference, it samples the attention location based on the policy $\pi(l_t|f_l(h_t, \theta_l))$. Figure 2 illustrates the inference procedure.

3.3. Dynamic Computational Time for Recurrent Attention (DT-RAM)

When the dynamic structure comes to RAM, we simply augment it with an additional set of actions $\{a_t\}$ that decides when it will stop taking further attention and output results. $a_t \in \{0, 1\}$ is a binary variable with 0 representing “continue” and 1 representing “stop”. Its sampling policy is modeled via a stopping network $f_a(h_t, \theta_a)$. During inference, we sample both the attention l_t and stopping a_t with each policy independently.

$$l_t \sim \pi(l_t|f_l(h_t, \theta_l)), \quad a_t \sim \pi(a_t|f_a(h_t, \theta_a)) \quad (4)$$

Figure 3 shows how the model works. Compared to Figure 2, the change is simply by adding a_t to each time step.

Figure 4 illustrates how DT-RAM adapts its model structure and computational time to different input images for image recognition. When the input image is “easy” to recognize (Figure 4 left), we expect DT-RAM stop at the first few steps. When the input image is “hard” (Figure 4 right), we expect the model learn to continue searching for informative regions.

3.4. Training

Given a set of training images with ground truth labels $(x_n, y_n)_{n=1 \dots N}$, we jointly optimize the model parameters by computing the following gradient:

$$\frac{\partial \mathcal{L}}{\partial \theta} \approx \sum_n \sum_S \left(-\frac{\partial \log P(S|x_n, \theta)}{\partial \theta} R_n + \frac{\partial L_S(x_n, y_n, \theta)}{\partial \theta} \right) \quad (5)$$

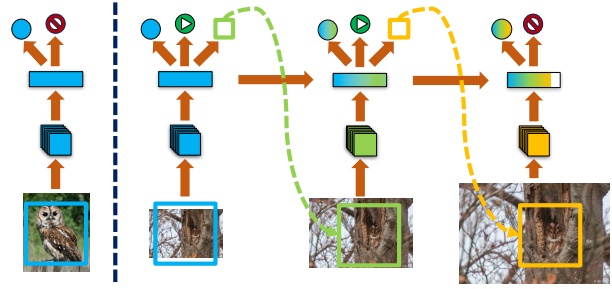


Figure 4. An illustration of how DT-RAM adapts its model structure and computational time to different input images.

where $\theta = \{\theta_f, \theta_l, \theta_a, \theta_c\}$ are the parameters of the recurrent network, the attention network, the stopping network and the classification network respectively.

Compared to Equation 2, Equation 5 is an approximation where we use a negative of reward function R to replace the loss of a given structure L_S in the first term. This training loss is similar to [6, 7]. Although the loss in Equation 2 can be optimized directly, using R can reduce the variance in the estimator [7].

$$P(S|x_n, \theta) = \prod_{t=1}^{T(n)} \pi(l_t|f_l(h_t, \theta_l))\pi(a_t|f_a(h_t, \theta_a)) \quad (6)$$

is the sampling policy for structure S .

$$R_n = \sum_{t=1}^{T(n)} \gamma^t r_{nt} \quad (7)$$

is the cumulative discounted reward over $T(n)$ time steps for the n -th training example. The discount factor γ controls the trade-off between making correct classification and taking more attentions. r_{nt} is the reward at t -th step. During experiments, we use a delayed reward. We set $r_{nt} = 0$ if $t \neq T(n)$ and $r_{nT} = 1$ only if $y = \arg \max_y P(y|f_c(h_T, \theta_c))$.

Intermediate Supervision: Unlike original RAM [6], DT-RAM has intermediate supervision for the classification network at every time step, since its underlying dynamic structure could require the model to output classification scores at any time step. The loss of

$$L_S(x_n, y_n, \theta) = \sum_{t=1}^{T(n)} L_t(x_n, y_n, \theta_h, \theta_c) \quad (8)$$

is the average cross-entropy classification loss over N training samples and $T(n)$ time steps. Note that T depends on n , indicating that each instance may have different stopping times. During experiments, we find intermediate supervision is also effective for the baseline RAM.

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.

Curriculum Learning: During experiments, we adopt a gradual training approach for the sake of accuracy. First, we start with a base convolutional network (e.g. Residual Networks [23]) pre-trained on ImageNet [60]. We then fine-tune the base network on the fine-grained dataset. This gives us a very high baseline. Second, we train the RAM model by gradually increase the total number of time steps. Finally, we initialize DT-RAM with the trained RAM and further fine-tune the whole network with REINFORCE algorithm.

4. Experiments

4.1. Dataset

We conduct experiments on three popular benchmark datasets: MNIST [14], CUB-200-2011 [15] and Stanford Cars [16]. Table 1 summarizes the details of each dataset. MNIST contains 70,000 images with 10 digital numbers. This is the dataset where the original visual attention model tests its performance. However, images in MNIST dataset are often too simple to generate conclusions to natural images. Therefore, we also compare on two challenging fine-grained recognition dataset. CUB-200-2011 [15] consists of 11,778 images with 200 bird categories. Stanford Cars [16] includes 16,185 images of 196 car classes. Both datasets contain a bounding box annotation in each image. CUB-200-2011 also contains part annotation, which we do not use in our algorithm. Most of the images in these two datasets have cluttered background, hence visual attention could be effective for them. All models are trained and tested without ground truth bounding box annotations.

4.2. Implementation Details

MNIST: We use the original digital images with 28×28 pixel resolution. The digits are generally centered in the image. We first train multiple RAM models with up to 7 steps. At each time step, we crop a 8×8 patch from the image based on the sampled attention location. The 8×8 patch only captures a part of a digit, hence the model usually requires multiple steps to produce an accurate prediction.

The attention network, classification network and stopping network all output actions at every time step. The output dimensions of the three networks are 2, 10 and 1 respectively. All three networks are linear layers on top of the

recurrent network. The classification network and stopping network then have softmax layers, computing the discrete probability. The attention network defines a Gaussian policy with a fixed variance, representing the continuous distribution of the two location variables. The recurrent state vector has 256 dimensions. All methods are trained using stochastic gradient descent with batch size of 20 and momentum of 0.9. The reward at the last time step is 1 if the agent classifies correctly and 0 otherwise. The rewards for all other time steps are 0. One can refer to [6] for more training details.

CUB-200-2011 and Stanford Cars: We use the same setting for both dataset. All images are first normalized by resizing to 512×512 . We then crop a 224×224 image patch at every time step except the first step, which is a key difference from MNIST. At the first step, we use a 448×448 crop. This guarantees the 1-step RAM and 1-step DT-RAM to have the same performance as the baseline convolutional network. We use Residual Networks [23] pre-trained on ImageNet [60] as the baseline network. We use the “pool-5” feature as input to the recurrent hidden layer. The recurrent layer is a fully connected layer with ReLU activations. The attention network, classification network and stopping network are all linear layers on top of the recurrent network. The dimensionality of the hidden layer is set to 512.

All models are trained using stochastic gradient descent (SGD) with momentum of 0.9 for 90 epochs. The learning rate is set to 0.001 at the beginning and multiplied by 0.1 every 30 epochs. The batch size is 28 which is the maximum we can use for 512×512 resolution. (For diagnostic experiments with smaller images, we use batch size of 96.) The reward strategy is the same as MNIST. During testing, the actions are chosen to be the maximal probability output from each network. Note that although bounding boxes or part-level annotations are available with these datasets, we do not utilize any of them throughout the experiments.

Computational Time: Our implementation is based on Torch [61]. The computational time heavily depends on the resolution of the input image and the baseline network structure. We run all our experiments on a single Tesla K-40 GPU. The average running time for a ResNet-50 on a 512×512 resolution image is 42ms. A 3-step RAM is 77ms since it runs ResNet-50 on 2 extra 224×224 images.

4.3. Comparison with State-of-the-Art

MNIST: We train two DT-RAM models with different discount factors. We train DT-RAM-1 with a smaller discount factor (0.98) and DT-RAM-2 with a larger discount factor (0.99). The smaller discount factor will encourage the model to stop early in order to obtain a large reward, hence one can expect DT-RAM-1 stops with less number of steps than DT-RAM-2.

Table 2 summarizes the performance of different model-

MNIST	# Steps	Error(%)
FC, 2 layers (256 hiddens 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].

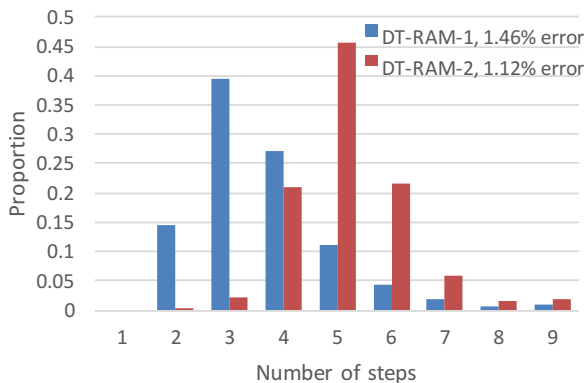


Figure 5. The distribution of number of steps from two different DT-RAM models on MNIST dataset. A model with longer average steps tends to have a better accuracy.

s on MNIST. Comparing to RAM with similar number of steps, DT-RAM achieve a better error rate. For example, DT-RAM-1 gets 1.46% recognition error with an average of 3.6 steps while RAM with 4 steps gets 1.54% error. Similarly, DT-RAM-2 gets 1.12% error with an average of 5.2 steps while RAM with 5 steps has 1.34% error.

Figure 5 shows the distribution of number of steps for the two DT-RAM models across all testing examples. We find a clear trade-off between efficiency and accuracy. DT-RAM-1 which has less computational time, achieves higher error than DT-RAM-2.

CUB-200-2011: We compare our model with all previously published methods. Table 3 summarizes the results. We observe that methods including Bilinear CNN [54, 53] (84.1% - 84.2%), Spatial Transformer Networks [59] (84.1%) and Fully Convolutional Attention Networks [62] (84.3%) all achieve similar performances. [9] further improve the testing accuracy to 85.4% by utilizing attribute labels annotated in the dataset.

Surprisingly, we find that a carefully fine-tuned Residual Network with 50 layers already hit 84.5%, surpassing most of the existing works. Adding further recurrent visual attention (RAM) reaches 86.0%, improving the baseline Resid-

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.

ual Net by 1.5%, leading to a new state-of-the-art on CUB-200-2011. DT-RAM further improves RAM, by achieving the same state-of-the-art performance, with less number of computational time on average (1.9 steps v.s. 3 steps).

Stanford Cars: We also compare extensively on Stanford Cars dataset. Table 4 shows the results. Surprisingly, a fine-tuned 50-layer Residual Network again achieves 92.3% accuracy on the testing set, surpassing most of the existing work. This suggests that a single deep network (without any further modification or extra bounding box supervision) can be the first choice for fine-grained recognition.

A 3-step RAM on top of the Residual Net further improves to 93.1% accuracy, which is so far the new state-of-the-art performance on Stanford Cars dataset. Compared to [62], this is achieved without using bounding box annotation during testing. DT-RAM again achieves the same accuracy as RAM but using 1.9 steps on average. We also observe that the relative improvement of RAM to the baseline model is no longer large (0.8%).

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.

4.4. Ablation Study

We conduct a set of ablation studies to understand how each component affects RAM and DT-RAM on fine-grained recognition. We work with CUB-200-2011 dataset since its images are real and challenging. However, due to the large resolution of the images, we are not able to run many steps with a very deep Residual Net. Therefore, instead of using Resnet-50 with 512×512 image resolution which produces state-of-the-art results, we use Resnet-34 with 256×256 as the baseline model (79.9%). This allows us to train more steps on the bird images to deeply understand the model.

Input Resolution and Network Depth: Table 5 compares the effect on different image resolutions with different network depths. In general, a higher image resolution significantly helps fine-grained recognition. For example, given the same ResNet-50 model, 512×512 resolution with 448×448 crops gets 84.5% accuracy, better than 81.5% from 256×256 with 224×224 crops. This is probably because higher resolution images contain more detailed information for fine-grained recognition. A deeper Residual Network also significantly improves performance. For example, ResNet-50 obtains 81.5% accuracy compared to ResNet-34 with only 79.9%. During experiments, we also train a ResNet-101 on 224×224 crops and get 82.8% recognition accuracy.

RAM v.s. DT-RAM: Table 6 shows how the number of steps affects RAM on CUB-200-2011 dataset. Starting from ResNet-34 which is also the 1 step RAM, the model gradually increases its accuracy with more steps (79.9% → 81.8%). After 5 steps, RAM no longer improves. DT-RAM also reaches the same accuracy. However, it only uses 3.5 steps on average than 6 steps, which is a promising trade-off between computation and performance.

Figure 6 plots the distribution of number of steps from the DT-RAM on the testing images. Surprisingly different from MNIST, the model prefers to stop either at the beginning of the time (1-2 steps), or in the end of the time (5-6 steps). There are very few images that the model choose to stop at 3-4 steps.

Learned Policy v.s. Fixed Policy: One may suspect that instead of learning the optimal stopping policy, whether we can just use a fixed stopping policy on RAM to determine when to stop the recurrent iteration. For example, one can simply learn a RAM model with intermediate supervision at every time step and use a threshold over the classification

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.

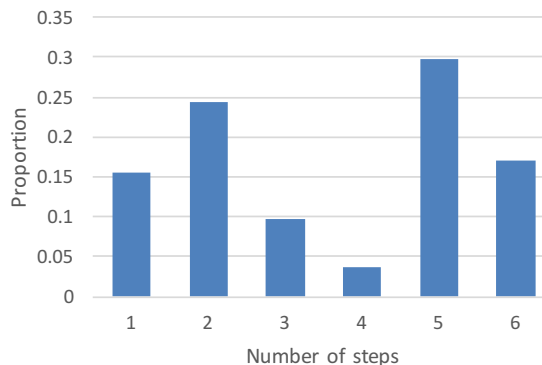


Figure 6. The distribution of number of steps from a DT-RAM model on CUB-200-2011 dataset. Unlike MNIST, the distribution suggests the model prefer either stop at the beginning or in the end.

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.

network $f_c(h_t, \theta_c)$ to determine when to stop. We compare the results between the described fixed policy and DT-RAM. Table 7 shows the comparison. We find that although the fixed stopping policy gives reasonably good results (i.e. 3.6 steps with 81.3% accuracy), DT-RAM still works slightly better (i.e. 3.6 steps with 81.8% accuracy).

Curriculum Learning: Table 8 compares the results on whether using curriculum learning to train RAM. If we learn the model parameters completely from scratch without curriculum, the performance start to decrease with more time steps (79.9% → 80.9% → 80.0%). This is because sim-

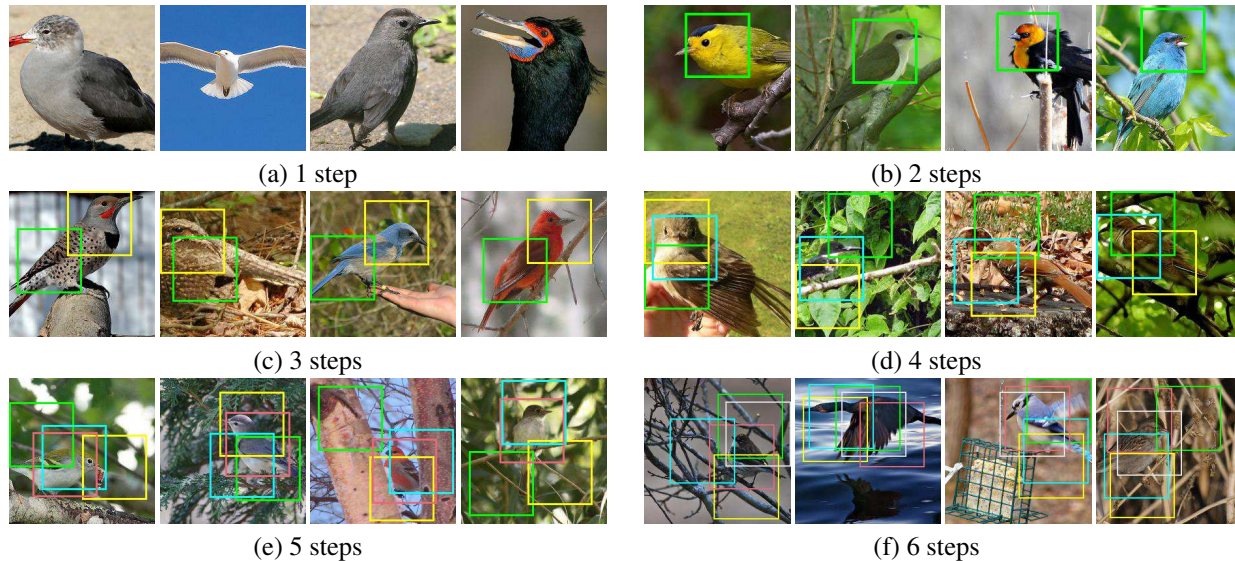


Figure 7. Qualitative results of DT-RAM on CUB-200-2011 *testing* set. We show images with different ending steps from 1 to 6. Each bounding box indicates an attention region. Bounding box colors are displayed in order. The first step uses the full image as input hence there is no bounding box. From 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.

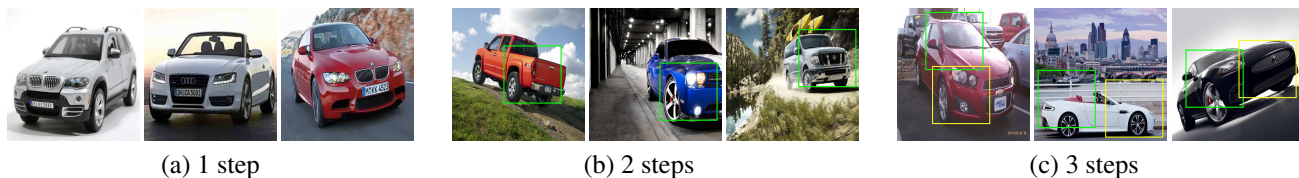


Figure 8. Qualitative results of DT-RAM on Stanford Car *testing* set. We only manage to train a 3-step model with 512×512 resolution.

# 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.

# 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.

ple policy gradient method becomes harder to train with longer sequences. Curriculum learning makes training more stable, since it guarantees the accuracy on adding more steps will not hurt the performance. The testing performance hence gradually increases over number of steps, from 79.9% to 81.8%.

Intermediate Supervision: Table 9 compares the testing results for using intermediate supervision. Note that the original RAM model [6] only computes output at the last time step. Although this works for small dataset like MNIST, when the input images become more challenging and time step increases, the RAM model learned without intermediate supervision starts to get worse. On contrary, adding an intermediate loss at each step makes RAM models with more steps steadily improve the final performance.

Qualitative Results: We visualize the qualitative results of DT-RAM on CUB-200-2011 and Stanford Cars testing

set in Figure 7 and Figure 8 respectively. From step 1 to step 6, we observe a gradual increase of background clutter and recognition difficulty, matching our hypothesis of using dynamic computation time for different types of images.

5. Conclusion and Future Work

In this work we present a simple but novel method for learning to dynamically adjust computational time during inference with reinforcement learning. We apply it on the recurrent visual attention model and show its effectiveness for fine-grained recognition. We believe that such methods will be important for developing dynamic reasoning in deep learning and computer vision. Future work on developing more sophisticated dynamic models for reasoning and apply it to more complex tasks such as visual question answering will be conducted.

References

- [1] Mary Hayhoe and Dana Ballard. Eye movements in natural behavior. *Trends in cognitive sciences*, 9(4):188–194, 2005. [1](#)
- [2] Robert Desimone and John Duncan. Neural mechanisms of selective visual attention. *Annual review of neuroscience*, 18(1):193–222, 1995. [1](#)
- [3] Diane M Beck and Sabine Kastner. Top-down and bottom-up mechanisms in biasing competition in the human brain. *Vision research*, 49(10):1154–1165, 2009. [1](#)
- [4] Robert Desimone. Visual attention mediated by biased competition in extrastriate visual cortex. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 353(1373):1245, 1998. [1](#)
- [5] Radoslaw Martin Cichy, Dimitrios Pantazis, and Aude Oliva. Resolving human object recognition in space and time. *Nature neuroscience*, 17(3):455–462, 2014. [1](#)
- [6] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in neural information processing systems*, pages 2204–2212, 2014. [1](#), [2](#), [4](#), [5](#), [6](#), [8](#)
- [7] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014. [1](#), [2](#), [4](#)
- [8] Pierre Sermanet, Andrea Frome, and Esteban Real. Attention for fine-grained categorization. *arXiv preprint arXiv:1412.7054*, 2014. [1](#), [2](#)
- [9] Xiao Liu, Jiang Wang, Shilei Wen, Errui Ding, and Yuanqing Lin. Localizing by describing: Attribute-guided attention localization for fine-grained recognition. *arXiv preprint arXiv:1605.06217*, 2016. [1](#), [3](#), [6](#)
- [10] Gustavo Deco and Edmund T Rolls. A neurodynamical cortical model of visual attention and invariant object recognition. *Vision research*, 44(6):621–642, 2004. [1](#)
- [11] Alex Graves. Adaptive computation time for recurrent neural networks. *arXiv preprint arXiv:1603.08983*, 2016. [1](#), [2](#)
- [12] Richard S Sutton, David A McAllester, Satinder P Singh, Yishay Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, volume 99, pages 1057–1063, 1999. [2](#), [3](#)
- [13] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009. [2](#)
- [14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. [2](#), [5](#)
- [15] Catherine Wah, Steve Branson, Peter Welinder, Pietro Perona, and Serge Belongie. The caltech-ucsd birds-200-2011 dataset. 2011. [2](#), [5](#)
- [16] Jonathan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision Workshops*, pages 554–561, 2013. [2](#), [3](#), [5](#)
- [17] Laurent Itti, Christof Koch, and Ernst Niebur. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on pattern analysis and machine intelligence*, 20(11):1254–1259, 1998. [2](#)
- [18] Laurent Itti and Christof Koch. Computational modelling of visual attention. *Nature reviews neuroscience*, 2(3):194–203, 2001. [2](#)
- [19] John K Tsotsos, Scan M Culhane, Winky Yan Kei Wai, Yuzhong Lai, Neal Davis, and Fernando Nuflo. Modeling visual attention via selective tuning. *Artificial intelligence*, 78(1-2):507–545, 1995. [2](#)
- [20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. [2](#)
- [21] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. [2](#)
- [22] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015. [2](#)
- [23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. [2](#), [5](#), [6](#)
- [24] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. [2](#)
- [25] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C Courville, Ruslan Salakhutdinov, Richard S Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, volume 14, pages 77–81, 2015. [2](#)
- [26] Huijuan Xu and Kate Saenko. Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In *European Conference on Computer Vision*, pages 451–466. Springer, 2016. [2](#)
- [27] Kan Chen, Jiang Wang, Liang-Chieh Chen, Haoyuan Gao, Wei Xu, and Ram Nevatia. Abc-cnn: An attention based convolutional neural network for visual question answering. *arXiv preprint arXiv:1511.05960*, 2015. [2](#)
- [28] Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*, 2016. [2](#)
- [29] Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question

- answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 21–29, 2016. 2
- [30] Serena Yeung, Olga Russakovsky, Greg Mori, and Li Fei-Fei. End-to-end learning of action detection from frame glimpses in videos. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2678–2687, 2016. 2
- [31] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*, volume 1. MIT press Cambridge, 1998. 2
- [32] Amir R Zamir, Te-Lin Wu, Lin Sun, William Shen, Jitendra Malik, and Silvio Savarese. Feedback networks. *arXiv preprint arXiv:1612.09508*, 2016. 2
- [33] Marijn F Stollenga, Jonathan Masci, Faustino Gomez, and Jürgen Schmidhuber. Deep networks with internal selective attention through feedback connections. In *Advances in Neural Information Processing Systems*, pages 3545–3553, 2014. 2
- [34] Chunshui Cao, Xianming Liu, Yi Yang, Yinan Yu, Jiang Wang, Zilei Wang, Yongzhen Huang, Liang Wang, Chang Huang, Wei Xu, et al. Look and think twice: Capturing top-down visual attention with feedback convolutional neural networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2956–2964, 2015. 2
- [35] Qian Wang, Jiaying Zhang, Sen Song, and Zheng Zhang. Attentional neural network: Feature selection using cognitive feedback. In *Advances in Neural Information Processing Systems*, pages 2033–2041, 2014. 2
- [36] Joao Carreira, Pulkit Agrawal, Katerina Fragkiadaki, and Jitendra Malik. Human pose estimation with iterative error feedback. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4733–4742, 2016. 2
- [37] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016. 2
- [38] Peiyun Hu and Deva Ramanan. Bottom-up and top-down reasoning with hierarchical rectified gaussians. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5600–5609, 2016. 2
- [39] Michael Figurnov, Maxwell D Collins, Yukun Zhu, Li Zhang, Jonathan Huang, Dmitry Vetrov, and Ruslan Salakhutdinov. Spatially adaptive computation time for residual networks. *arXiv preprint arXiv:1612.02297*, 2016. 2
- [40] Mark Neumann, Pontus Stenetorp, and Sebastian Riedel. Learning to reason with adaptive computation. *arXiv preprint arXiv:1610.07647*, 2016. 2
- [41] Yacine Jernite, Edouard Grave, Armand Joulin, and Tomas Mikolov. Variable computation in recurrent neural networks. *arXiv preprint arXiv:1611.06188*, 2016. 2
- [42] Augustus Odena, Dieterich Lawson, and Christopher Olah. Changing model behavior at test-time using reinforcement learning. *arXiv preprint arXiv:1702.07780*, 2017. 2
- [43] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. 2009. 2
- [44] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101—mining discriminative components with random forests. In *European Conference on Computer Vision*, pages 446–461. Springer, 2014. 3
- [45] Thomas Berg, Jiongxin Liu, Seung Woo Lee, Michelle L Alexander, David W Jacobs, and Peter N Belhumeur. Birdsnap: Large-scale fine-grained visual categorization of birds. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2011–2018, 2014. 3
- [46] Yin Cui, Feng Zhou, Yuanqing Lin, and Serge Belongie. Fine-grained categorization and dataset bootstrapping using deep metric learning with humans in the loop. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1153–1162, 2016. 3
- [47] Shaoli Huang, Zhe Xu, Dacheng Tao, and Ya Zhang. Part-stacked cnn for fine-grained visual categorization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1173–1182, 2016. 3
- [48] Jonathan Krause, Hailin Jin, Jianchao Yang, and Li Fei-Fei. Fine-grained recognition without part annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5546–5555, 2015. 3, 6
- [49] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Fei-Fei Li. Novel dataset for fine-grained image categorization: Stanford dogs. In *Proc. CVPR Workshop on Fine-Grained Visual Categorization (FGVC)*, volume 2, 2011. 3
- [50] Jiongxin Liu, Angjoo Kanazawa, David Jacobs, and Peter Belhumeur. Dog breed classification using part localization. In *European Conference on Computer Vision*, pages 172–185. Springer, 2012. 3
- [51] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *Computer Vision, Graphics & Image Processing, 2008. ICVGIP'08. Sixth Indian Conference on*, pages 722–729. IEEE, 2008. 3
- [52] Yang Gao, Oscar Beijbom, Ning Zhang, and Trevor Darrell. Compact bilinear pooling. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 317–326, 2016. 3
- [53] Shu Kong and Charles Fowlkes. Low-rank bilinear pooling for fine-grained classification. *arXiv preprint arXiv:1611.05109*, 2016. 3, 6
- [54] Tsung-Yu Lin, Aruni RoyChowdhury, and Subhransu Maji. Bilinear cnn models for fine-grained visual recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1449–1457, 2015. 3, 6
- [55] Thomas Berg and Peter Belhumeur. Poof: Part-based one-vs.-one features for fine-grained categorization, face verification, and attribute estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 955–962, 2013. 3

- [56] Steve Branson, Grant Van Horn, Serge Belongie, and Pietro Perona. Bird species categorization using pose normalized deep convolutional nets. *arXiv preprint arXiv:1406.2952*, 2014. 3, 6
- [57] Efstratios Gavves, Basura Fernando, Cees GM Snoek, Arnold WM Smeulders, and Tinne Tuytelaars. Fine-grained categorization by alignments. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1713–1720, 2013. 3
- [58] Jonathan Krause, Benjamin Sapp, Andrew Howard, Howard Zhou, Alexander Toshev, Tom Duerig, James Philbin, and Li Fei-Fei. The unreasonable effectiveness of noisy data for fine-grained recognition. In *European Conference on Computer Vision*, pages 301–320. Springer, 2016. 3
- [59] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, pages 2017–2025, 2015. 3, 6
- [60] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. 5
- [61] Ronan Collobert, Koray Kavukcuoglu, and Clément Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, number EPFL-CONF-192376, 2011. 5
- [62] Xiao Liu, Tian Xia, Jiang Wang, Yi Yang, Feng Zhou, and Yuanqing Lin. Fine-grained recognition with automatic and efficient part attention. *arXiv preprint arXiv:1603.06765*, 2016. 6
- [63] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based r-cnns for fine-grained category detection. In *European conference on computer vision*, pages 834–849. Springer, 2014. 6
- [64] Marcel Simon and Erik Rodner. Neural activation constellations: Unsupervised part model discovery with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1143–1151, 2015. 6
- [65] Yuning Chai, Victor Lempitsky, and Andrew Zisserman. Symbiotic segmentation and part localization for fine-grained categorization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 321–328, 2013. 6
- [66] Philippe-Henri Gosselin, Naila Murray, Hervé Jégou, and Florent Perronnin. Revisiting the fisher vector for fine-grained classification. *Pattern Recognition Letters*, 49:92–98, 2014. 6
- [67] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 6
- [68] Yaming Wang, Jonghyun Choi, Vlad Morariu, and Larry S Davis. Mining discriminative triplets of patches for fine-grained classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1163–1172, 2016. 6