Energy-efficient Amortized Inference with Cascaded Deep Classifiers

Jiaqi Guan[1,2], Yang Liu[2], Qiang Liu[3] and Jian Peng[2]

[1] Department of Automation, Tsinghua University
[2] Department of Computer Science, University of Illinois, Urbana-Champaign
[3] Department of Computer Science, The University of Texas at Austin

Introduction

Motivations: More complex deep neural networks have been proposed to further improve performance, but often at the cost of more expensive computation. However, in many real-world scenarios, we encounter a significant constraint of energy or computational cost for real-time inference.

Goal: Predicting both *accurate* and *fast* focusing on test-time energy-efficient inference of image classification.

Contributions:

- Proposed an energy-efficient model by cascading deep classifiers with a policy module.
- Policy model and cascading classifiers are *jointly* trained by *REINFORCE*.
- This model can choose the smallest classifier which is sufficient to make accurate prediction for each input instance.
- Achieve both high accuracy and amortized efficiency on CIFAR and ImageNet datasets.

Method

Energy-constrained Inference of Cascaded Classifiers:

Suppose we have K classifiers $\{C_k\}_{k=1}^K$ with different energy cost $\{\mathcal{F}_k\}_{k=1}^K$, input x with true label y and predicted label $\hat{y}$, and a policy module $\Pi(k|x)$ which decides the probability of assigning input x to classifier C_k.

- Target: Jointly train all classifiers $\{C_k\}$ and the policy $\Pi(k|x)$ to minimize the expected loss function under the constraint that the expected energy cost should be no larger than a desired budget $\mathcal{B}$
- Constrained optimization:

$$\begin{aligned} &\max_{\Pi,\{C_t\}_{t=1}^K} \mathbb{E}_{(x,y)\sim\mathcal{D},k_x\sim\Pi(\cdot|x),\hat{y}\sim C_{k_x}(\cdot|x)}\left[-\mathcal{L}(\hat{y},y)\right] \\ &s.t \quad \mathbb{E}_{(x,y)\sim\mathcal{D},k_x\sim\Pi(\cdot|x)}\left[\mathcal{F}_{k_x}\right] < \mathcal{B}, \end{aligned} \tag{1}$$

where k_x denotes the (random) classifier ID assigned to x.

- Unconstrained optimization:

$$\max_{\Pi,\{C_t\}_{t=1}^K} \mathbb{E}_{(x,y)\sim\mathcal{D},k_x\sim\Pi(\cdot|x),y'\sim C_{k_x}(\cdot|x)}\left[-\mathcal{L}(y',y)-\alpha\mathcal{F}_{k_x}\right] \tag{2}$$

where α controls the trade-off between the predictive loss function and the energy cost.

Energy Efficient Inference via Optimal Stopping:

- Framing Π into a K-step optimal stopping process.

$$\Pi(k|x) = \pi_k(s_k(x))\prod_{t=1}^{k-1}(1-\pi_t(s_t(x))) \tag{3}$$

(s_t: some feature related to classifier C_t, $\pi_t(\cdot)$: stopping probability.)

- Framing the decision module as a Markov decision process.
  - Observation: s_t. We use the output label probability, that is $s_t(x) = C_t(\cdot|x)$.
  - Action: stop or forward
  - Reward: consisting of the negative loss function for prediction and the accumulated energy cost from the first step. We use the FLOPs count as the cost.

$$R(k,x,y,\hat{y}) = -\mathcal{L}(\hat{y},y) - \alpha\sum_{t=1}^{k-1}\mathcal{F}_t \tag{4}$$

- Final goal: Assume the stopping probabilities $\{\pi_t\}$ and classifiers $\{C_t\}$ are parameterized by $\theta = \{\theta^{\pi_t},\theta^{C_t}\}_{t=1}^K$. Our final goal is to find the optimal θ to maximize the expected return, by unrolling the conditional distributions defined by the entire policy:

$$\begin{aligned} J(\theta) &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\left[\mathbb{E}_{k\sim\Pi(\cdot|x),\hat{y}\sim C_k(\cdot|x)}R(k,x,y,\hat{y})\right] \\ &= \mathbb{E}_{(x,y)\sim\mathcal{D}}\Big[\sum_{k=1}^{K}\prod_{t=1}^{k-1}(1-\pi_t(s_t(x);\theta)) \\ &\quad \cdot \pi_k(s_k(x);\theta)\cdot\sum_{\hat{y}}C_k(\hat{y}|x;\theta)\cdot R(k,x,y,\hat{y})\Big]. \end{aligned} \tag{5}$$

Model

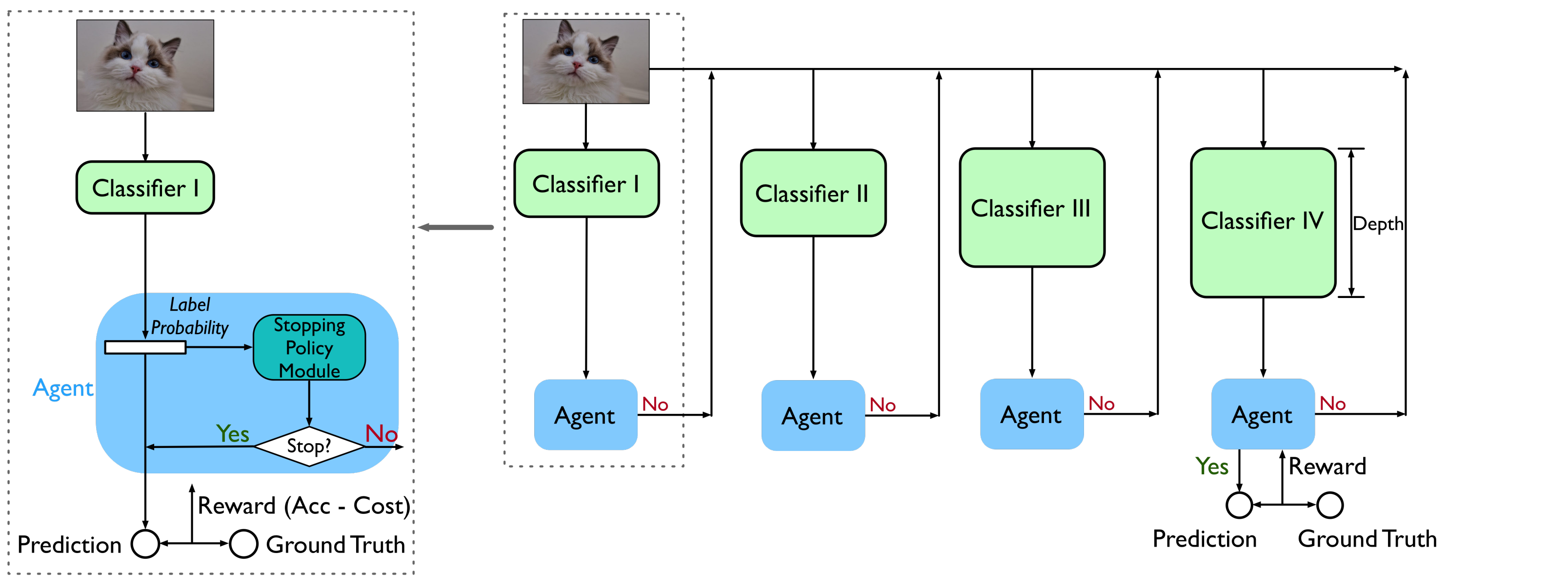


Figure 1: Our proposed model: Given an image in the dataset, starting from smallest model, our agent will decide whether to move to the next deeper model. If we decide to stop at a classifier, we predict the label based on the classifier. Finally, the agent will receive a reward consisting of both the negative loss function for prediction and the accumulated energy cost. Inside our agent, a stopping policy module takes label probability of a classifier's top layer as input and decides whether to stop or continue.

Solving by REINFORCE:

$$\begin{aligned} \widehat{\nabla_\theta J} = \nabla_\theta\Big(&\sum_{t=1}^{k-1}\log(1-\pi_t(s_t(x);\theta)) + \log(\pi_k(s_k(x);\theta)) \\ &+\log(C_k(\hat{y}|x;\theta))\Big)\cdot(R(k,x,y,\hat{y})-b) \end{aligned} \tag{6}$$

The *baseline* b is to reduce the variance in the estimated policy gradient.

Experiment

- **Datasets:** CIFAR-10, CIFAR-100, ImageNet32x32, ImageNet
- **Baselines:** Static ResNets[2], Adaptive Neural Networks[1]
- **Compare with Static ResNet Classifiers:**

CIFAR-10		
Model	Error	FLOPs
ResNet-8	12.33%	5.82%
ResNet-20	9.00%	16.90%
ResNet-32	8.40%	27.98%
ResNet-56	7.70%	50.14%
ResNet-110	7.38%	100.00%
Ours	**7.26%**	**18.16%**

CIFAR-100		
Model	Error	FLOPs
ResNet-8	39.98%	5.82%
ResNet-20	33.13%	16.90%
ResNet-32	31.56%	27.98%
ResNet-56	30.38%	50.14%
ResNet-110	28.63%	100.00%
Ours	**27.76%**	**48.98%**

ImageNet32x32 (Top-5 Error)		
Model	Error	FLOPs
ResNet40-1	39.72%	6.27%
ResNet40-1.5	32.76%	14.09%
ResNet40-2	29.64%	25.03%
ResNet40-3	24.67%	56.27%
ResNet40-4	22.22%	100.00%
Ours	**22.21%**	**66.22%**

ImageNet (Top-5 Error)		
Model	Error	FLOPs
AlexNet	20.08%	18.42%
GoogleNet	10.99%	39.47%
ResNet-50	7.80%	100.00%
ANN	8.14%	56.87%
Ours	**7.82%**	**53.47%**

- **Comparison of Joint Training:**

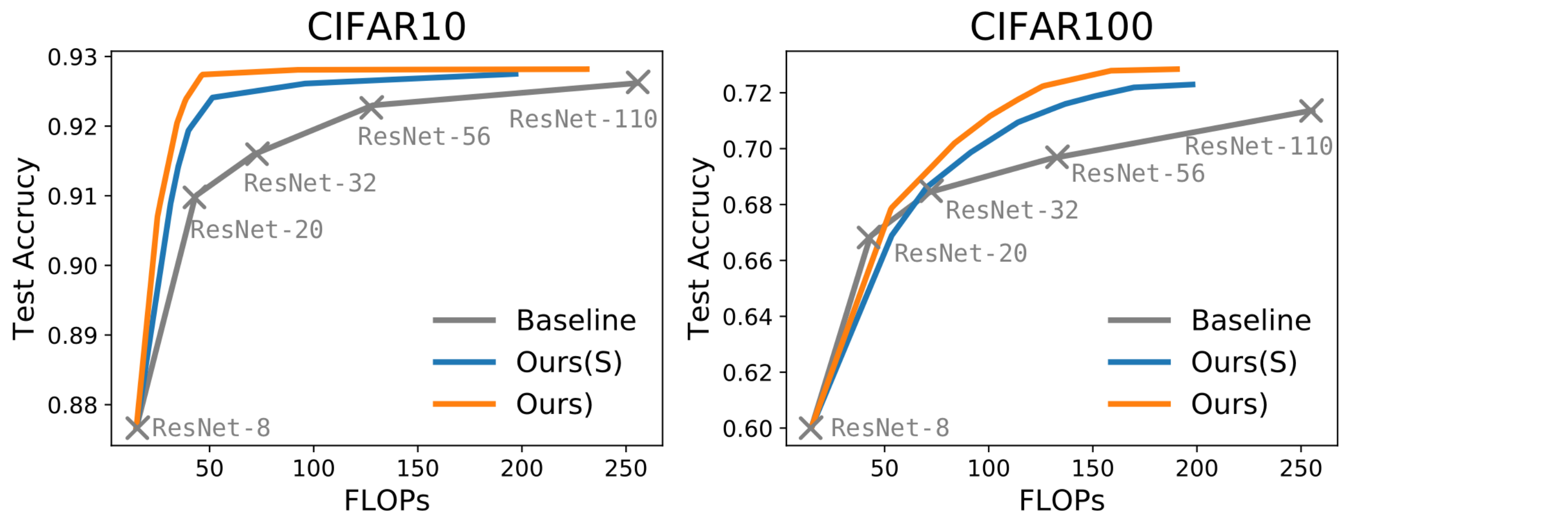
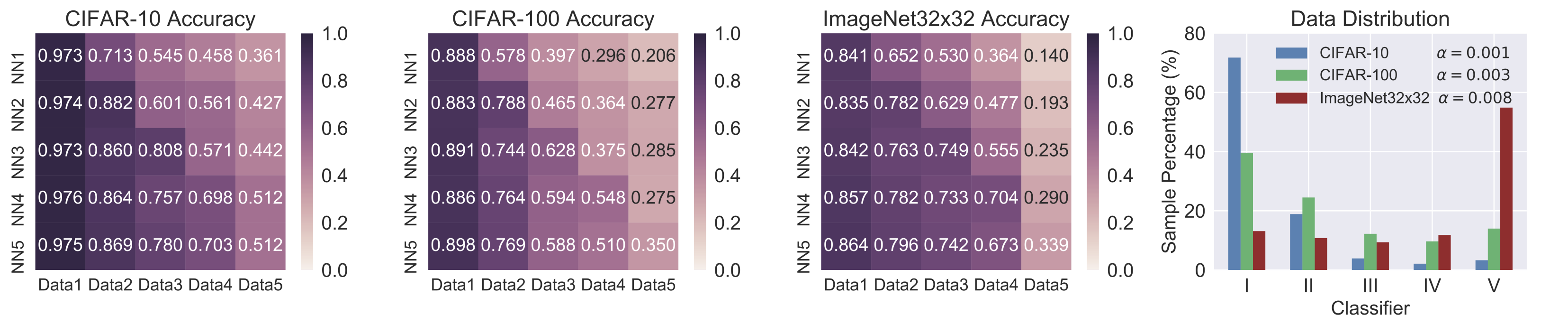


Figure 2: **Results on CIFAR-10/100:** The x-axis denotes the millions of FLOPs and y-axis denotes the corresponding accuracy obtained by the static ResNet (gray), our model (orange), and the simplified version of our model (blue, in which we only train the policy module), respectively.

- **Accuracy and Sample distribution:**



CIFAR-10 Accuracy

	Data1	Data2	Data3	Data4	Data5
NN1	0.973	0.713	0.545	0.458	0.361
NN2	0.974	0.882	0.601	0.561	0.427
NN3	0.973	0.860	0.808	0.571	0.442
NN4	0.976	0.864	0.757	0.698	0.512
NN5	0.975	0.869	0.780	0.703	0.512

CIFAR-100 Accuracy

	Data1	Data2	Data3	Data4	Data5
NN1	0.888	0.578	0.397	0.296	0.206
NN2	0.883	0.788	0.465	0.364	0.277
NN3	0.891	0.744	0.628	0.375	0.285
NN4	0.886	0.764	0.594	0.548	0.275
NN5	0.898	0.769	0.588	0.510	0.350

ImageNet32x32 Accuracy

	Data1	Data2	Data3	Data4	Data5
NN1	0.841	0.652	0.530	0.364	0.140
NN2	0.835	0.782	0.629	0.477	0.193
NN3	0.842	0.763	0.749	0.555	0.235
NN4	0.857	0.782	0.733	0.704	0.290
NN5	0.864	0.796	0.742	0.673	0.339

Figure 3: Accuracy distribution of classifiers in the cascade: The first three figures plot accuracy distributions on the CIFAR-10, CIFAR-100 and ImageNet32x32 datasets. The *i*-th row and *j*-th column denotes the average accuracy predicted by the *i*-th classifier of samples assigned to the *j*-th classifier. The fourth figure is the distribution of test images on individual classifiers, where x-axis indexes five classifiers and y-axis denotes the proportion of samples eventually assigned to the corresponding classifier.

- **Conclusion:** Tested on image classification, our model assigns a large portion of images to the smaller networks and remaining difficult images to the deeper models when necessary. In this way, our model is able to achieve both high accuracy and amortized efficiency during test time.

References

[1] T. Bolukbasi, J. Wang, O. Dekel, and V. Saligrama.
Adaptive neural networks for efficient inference.
In *International Conference on Machine Learning*, pages 527–536, 2017.

[2] K. He, X. Zhang, S. Ren, and J. Sun.
Deep residual learning for image recognition.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

guanjq14@mails.tsinghua.edu.cn

Jul 17th, 2018

jianpeng@illinois.edu