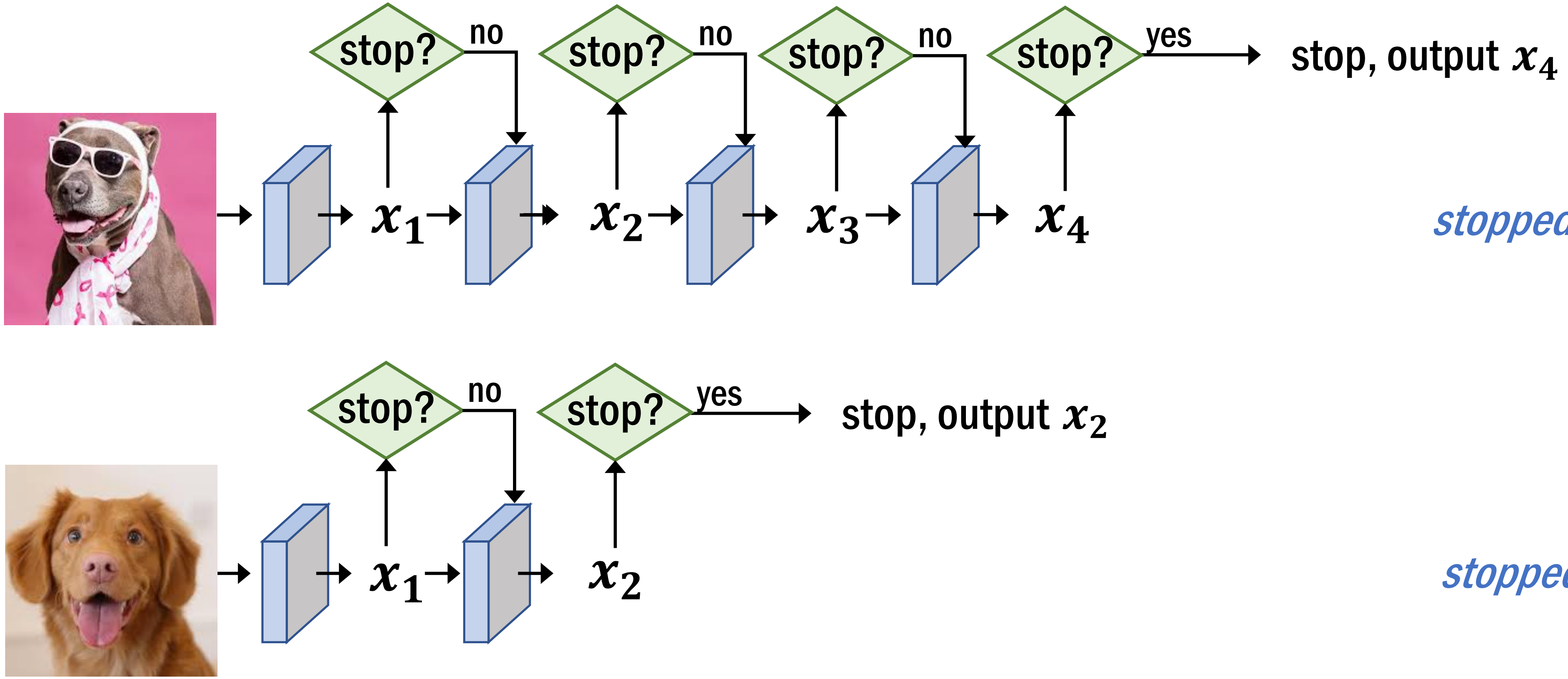
Learning To Stop While Learning To Predict

Xinshi Chen[1], Hanjun Dai[2], Yu Li[3], Xin Gao[3], Le Song[1,4]
[1]Georgia Tech, [2]Google Brain, [3]KAUST, [4]Ant Financial

ICML 2020

Dynamic Depth

stop at different depths for different input samples.



stopped depth=4

stopped depth=2

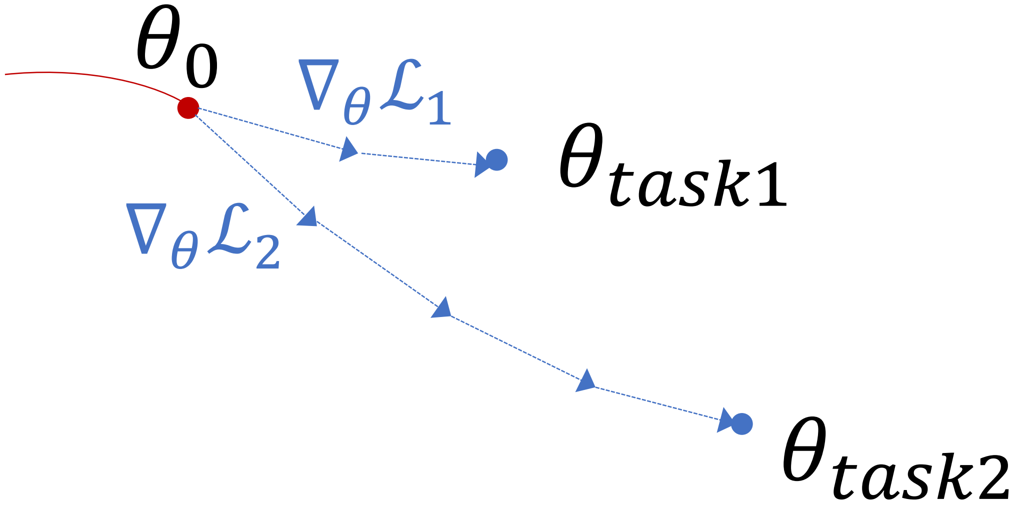
Motivation

1. Task-imbalanced Meta Learning

Task 1: fewer samples

Task 2: more samples

Need **different numbers of gradient steps** **for adaptation**

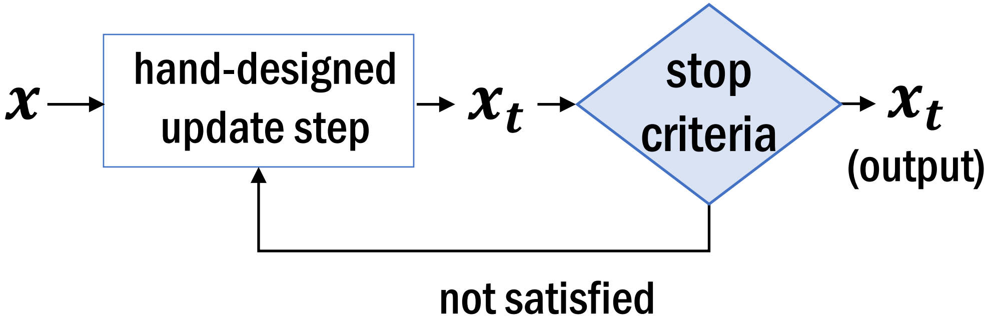


Motivation

2. Data-driven Algorithm Design

Traditional algorithms have certain stop criteria to determine the number of iterations for each problem.
E.g.,

- iterate until convergence
- early stopping to avoid over-fitting



Deep learning based algorithms usually have a **fixed number of iterations** in the architecture.

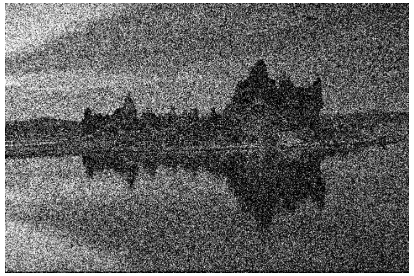
Motivation

3. Others

Image Denoising

- Images with different noise levels may need different number of denoising steps.

noisy



less noisy

Image Recognition

- ‘early exits’ is proposed to improve the computation efficiency and avoid ‘over-thinking’.

[Teerapittayanon et al., 2016; Zamir et al., 2017; Huang et al., 2018, Kaya et al. (2019)]

Predictive Model with Stopping Policy

5-minute Core Message

Predictive model $\mathcal{F}_\theta$

- Transforms the input $\boldsymbol{x}$ to generate a path of states $\boldsymbol{x}_1, \dots, \boldsymbol{x}_T$

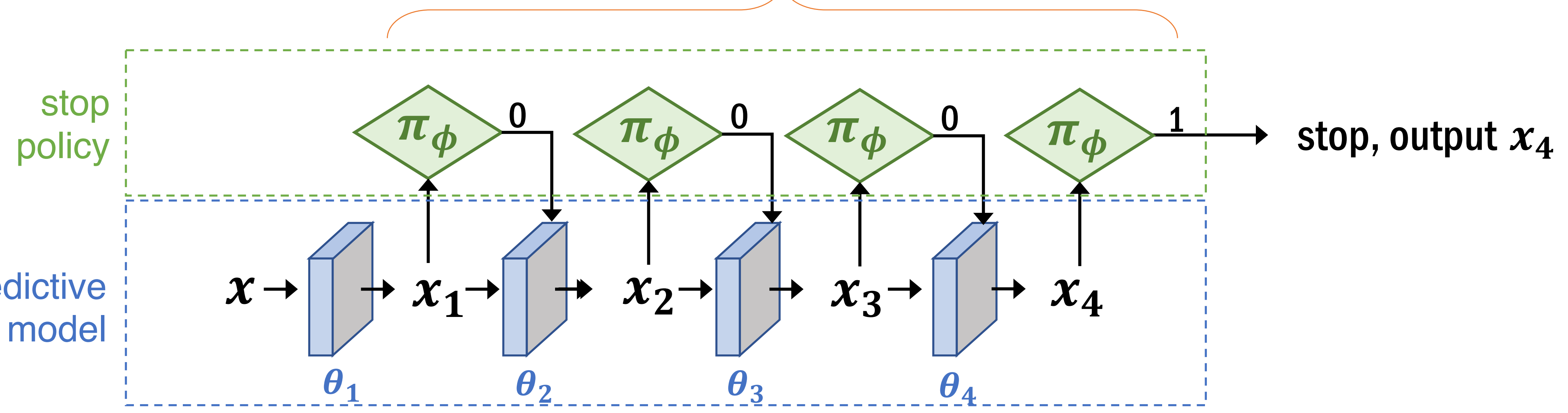
Stopping Policy π_ϕ

- Sequentially observes the states $\boldsymbol{x}_t$ and determines the probability of stop at layer t

Variational stop time distribution q_ϕ

- Stop time distribution induced by stopping policy $\boldsymbol{\pi}_\phi$

variational stop time distribution $q_\phi(t) = \pi_\phi(x_t) \prod_{\tau=1}^{t-1} (1 - \pi_\phi(x_\tau))$



How to learn the optimal $(\mathcal{F}_{\theta}, \pi_{\phi})$ efficiently?

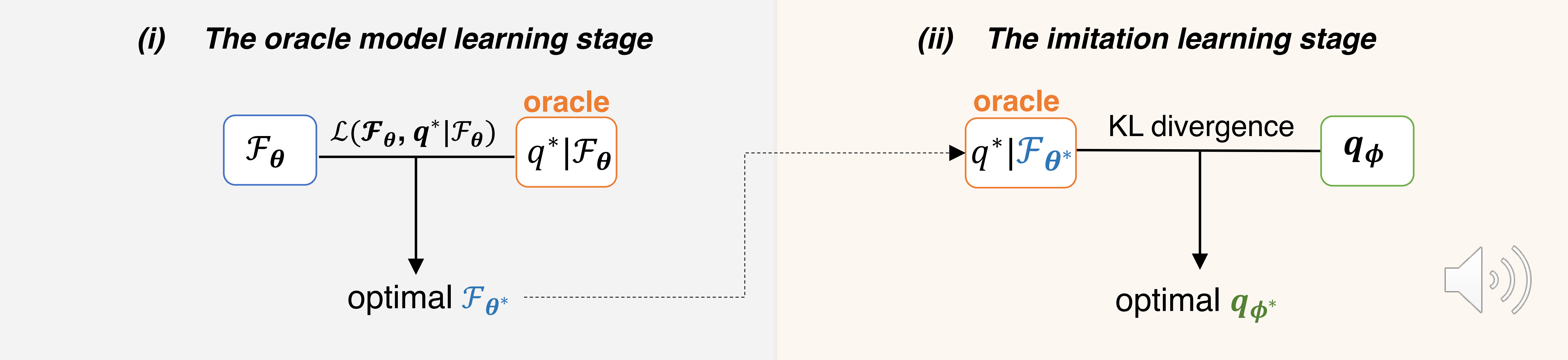
- Design a **joint training objective**:

$$\mathcal{L}(\mathcal{F}_{\theta}, q_{\phi})$$

- Introduce an **oracle stop time distribution**:

$$q^*|\mathcal{F}_{\theta} := \mathrm{argmin}_{q \in \Delta^{T-1}} \mathcal{L}(\mathcal{F}_{\theta}, q)$$

- Then we decompose the learning procedure into **two stages**:



Advantages of our training procedure

✓ **Principled**

- Two components are optimized towards a joint objective.

✓ **Tuning-free**

- Weights of different layers in the loss are given by the oracle distribution automatically.
- For different input samples, the weights on the layers can be different.

✓ **Efficient**

- Instead of updating θ and ϕ alternatively, $\boldsymbol{\theta}$ is optimized in 1st stage, and then ϕ is optimized in 2nd stage.

✓ **Generic**

- can be applied to a diverse range of applications.

✓ **Better understanding**

- A **variational Bayes** perspective, for better understanding the proposed model and joint training.
- A **reinforcement learning** perspective, for better understanding the learning of the stop policy.

Experiments

- Learning to optimize: sparse recovery
- Task-imbalanced meta learning: few-shot learning
- Image denoising
- Some observations on image recognition tasks.

Problem Formulation - Models

Predictive model $\mathcal{F}_\theta$

- $\boldsymbol{x}_t = f_{\theta_t}(\boldsymbol{x}_{t-1})$, for $t = 1,2,\ldots,T$

Stopping Policy π_ϕ

- $\pi_t = \pi_\phi(\boldsymbol{x}, \boldsymbol{x}_t)$, for $t = 1,2,\ldots,T$

Variational stop time distribution q_ϕ (induced by π_ϕ)

- $q_\phi(t) = \pi_t \underbrace{\prod_{\tau=1}^{t-1}(1-\pi_\tau)}_{\text{Pr[not stopped before t]}}$ for $t < T$
- Help design the training objective and the algorithm.

Problem Formulation – Optimization Objective

$$\mathcal{L}(\mathcal{F}_\theta, q_\phi; x, y) = \underbrace{\mathbb{E}_{t\sim q_\phi} l(y, x_t; \theta)}_{\text{loss in expectation over } t} - \beta \underbrace{H(q_\phi)}_{\text{entropy}}$$

- **Variational Bayes Perspective**

stop time t	latent variable
label $\boldsymbol{y}$	observation
loss $\ell(\boldsymbol{y}, \boldsymbol{x}_t; \theta)$	likelihood $p_\theta(\boldsymbol{y} \mid t, \boldsymbol{x})$
stop time distribution q_ϕ	posterior $p_\theta(t \mid \boldsymbol{y}, \boldsymbol{x})$
regularization	prior $p(t \mid \boldsymbol{x})$

$$\min_{\theta,\phi} \mathcal{L}(\mathcal{F}_\theta, q_\phi; x, y) \iff \max_{\theta,\phi} \mathcal{J}_{\beta-VAE}(\mathcal{F}_\theta, q_\phi; x, y)$$

equivalent

(i.e., β-VAE, ELBO)

Training Algorithm – Stage I

Oracle stop time distribution:

$$q_\theta^*(\cdot\,|y,x) \coloneqq \underset{\boldsymbol{q}\in\Delta^{T-1}}{\operatorname{argmax}}\, \mathcal{J}_{\beta-VAE}(\mathcal{F}_\theta, \boldsymbol{q}; x, y)$$

$$= \frac{p_\theta(y|t,x)^{1/\beta}}{\sum_{t=1}^{T} p_\theta(y|t,x)^{1/\beta}}$$

Interpretation:

- It is the optimal stop time distribution given a predictive model $\mathcal{F}_\theta$
- When $\beta = 1$, the oracle is the true posterior, $q_\theta^*(t|y,x) = p_\theta(t|y,x)$
- This posterior is computationally tractable, but it requires the knowledge of the true label y.

Stage I. Oracle model learning

$$\max_\theta \frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}} \mathcal{J}_{\beta-VAE}(\boldsymbol{\mathcal{F}_\theta}, \boldsymbol{q_\theta^*}; x, y) = \max_\theta \frac{1}{|\mathcal{D}|}\sum_{(x,y)\in\mathcal{D}}\sum_{t=1}^{T} q_\theta^*(t|y,x)\log \underbrace{\boldsymbol{p_\theta(y|t,x)}}_{\text{likelihood of the output at } t\text{-th layer}}$$

likelihood of the output at t-th layer

Training Algorithm – Stage II

Recall: Variational stop time distribution $\boldsymbol{q_\phi}(t|x)$ induced by the sequential policy $\boldsymbol{\pi_\phi}$

Hope: $\boldsymbol{q_\phi}(t|x)$ can mimic the oracle distribution $\boldsymbol{q^*_{\theta^*}}(t|y,x)$, by optimizing the **forward KL divergence**:

Stage II. Imitation With Sequential Policy

forward KL divergence

$$\mathrm{KL}(q^*_{\theta^*}||q_\phi) = -\sum_{t=1}^{T} q^*_{\theta^*}(t|y,x)\log q_\phi(t|x) - H(q^*_{\theta^*})$$

Note: If we use **reverse KL divergence**, then it is equivalent to solving **maximum-entropy RL**.

Experiment I - Learning To Optimize: Sparse Recovery

- *Task:* Recover x^* from its noisy measurements $b = Ax^* + \epsilon$
- *Traditional Approach:*
  - LASSO formulation $\min_x ½||b - Ax||_2^2 + \rho||x||_1$
  - Solved by iterative algorithms such as ISTA
- *Learning-based Algorithm:*
  - Learned ISTA (LISTA) is a deep architecture designed based on ISTA update steps
- *Ablation study:* Whether LISTA with adaptive depth (**LISTA-stop**) is better than **LISTA**.

Table 2. Recovery performances of different algorithms/models.

SNR	mixed	20	30	40
FISTA ($T = 100$)	-18.96	-16.75	-20.46	-20.97
ISTA ($T = 100$)	-14.66	-13.99	-14.99	-15.07
ISTA ($T = 20$)	-9.17	-9.12	-9.24	-9.16
FISTA ($T = 20$)	-11.12	-10.98	-11.19	-11.19
LISTA ($T = 20$)	-17.53	-16.53	-18.07	-18.20
LISTA-stop ($T \leqslant 20$)	**-22.41**	**-20.29**	**-23.90**	**-24.21**

Experiment II – Task-imbalanced Meta Learning

- *Task:* Task-imbalanced few-shot learning. Each task contains *k*-shots for each class where *k* can vary.
- *Our variant,* ***MAML-stop****:*
  - Built on top of MAML, but MAML-stop learns how many adaptation gradient descent steps are needed for each task.

Task-imbalanced setting:

Table 4. Task-imbalanced few-shot image classification.

	Omniglot 20-way, 1-5 shot	MiniImagenet 5-way, 1-10 shot
MAML	$97.96 \pm 0.3\%$	$57.20 \pm 1.1\%$
MAML-stop	$\mathbf{98.45 \pm 0.2\%}$	$\mathbf{60.67 \pm 1.0\%}$

Vanilla setting:

Table 5. Few-shot classification in vanilla meta learning setting (Finn et al., 2017) where all tasks have the same number of data points.

	Omniglot 5-way		Omniglot 20-way		MiniImagenet 5-way	
	1-shot	5-shot	1-shot	5-shot	1-shot	5-shot
MAML	$98.7 \pm 0.4\%$	$99.1 \pm 0.1\%$	$95.8 \pm 0.3\%$	$98.9 \pm 0.2\%$	$48.70 \pm 1.84\%$	$63.11 \pm 0.92\%$
MAML-stop	$\mathbf{99.62 \pm 0.22\%}$	$\mathbf{99.68 \pm 0.12\%}$	$\mathbf{96.05 \pm 0.35\%}$	$\mathbf{98.94 \pm 0.10\%}$	$\mathbf{49.56 \pm 0.82\%}$	$\mathbf{63.41 \pm 0.80\%}$

Experiment III – Image Denoising

- *Our variant, **DnCNN-stop**:*
  - Built on top of one of the most popular models, DnCNN, for the denoising task.

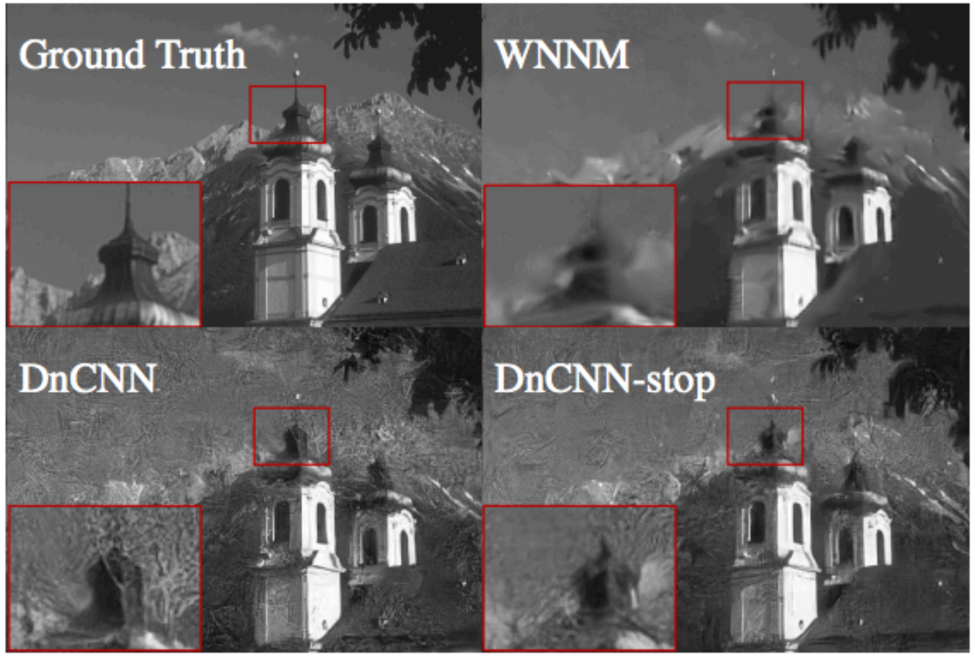


Figure 5. Denoising results of an image with noise level 65. (See Appendix B.3.2 for more visualization results.)

**Noise-level 65, 75 are not observed during training.*

σ	DnCNN-stop	DnCNN	UNLNet_5	BM3D	WNNM
35	**27.61**	27.60	27.50	26.81	27.36
45	**26.59**	26.56	26.48	25.97	26.31
55	**25.79**	25.71	25.64	25.21	25.50
*65	**23.56**	22.19	-	24.60	**24.92**
*75	**18.62**	17.90	-	24.08	**24.39**