Meta Learning for Few-shot Generation and Classification Tasks

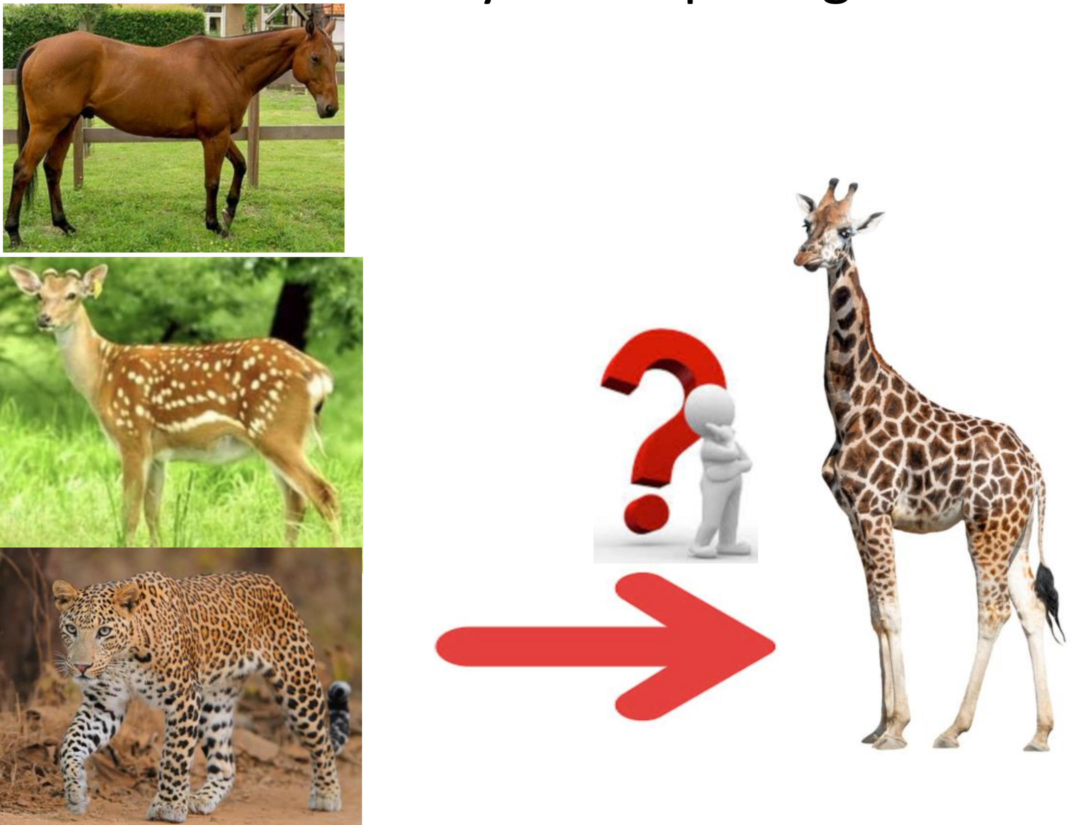
Yiping Song

2020.8.7

北京大學
PEKING UNIVERSITY

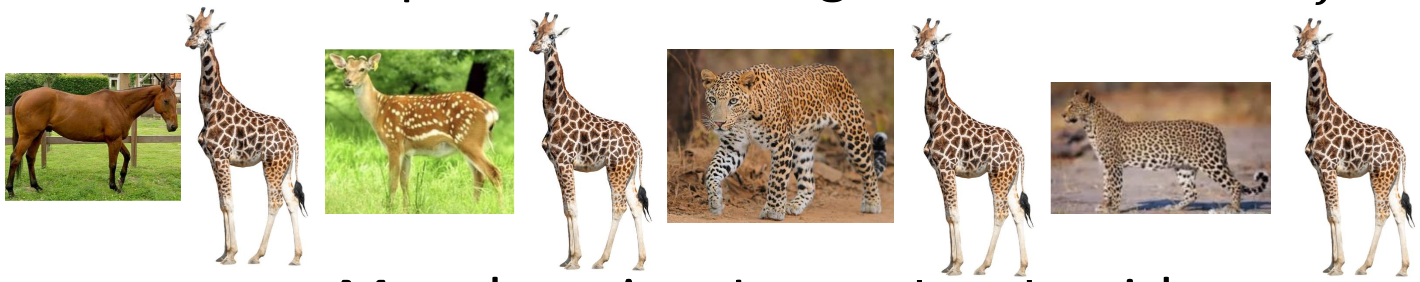
Introduction of Meta Learning

- Humans learn new concepts or skills faster than machines
  - Humans can recognize new species with few photos
  - Humans learn to ride a motorcycle fast if they can ride a bicycle.
- Reason:
  - Ability to adapt or generalize to new tasks



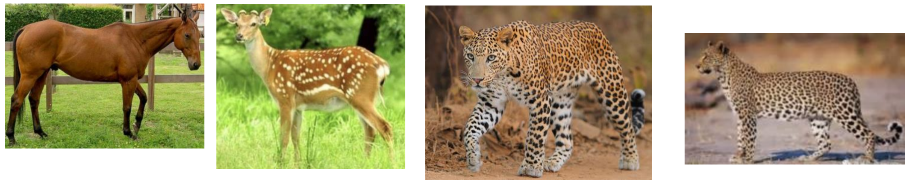
Introduction of Meta Learning

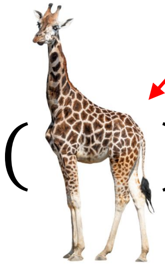
- Meta-learning (learn to learn)
  - Supervised learning: Learn a function f



$\longrightarrow f \longrightarrow f($  $) =$ "giraffe"

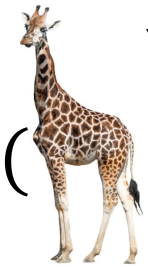
  - Meta learning: Learn the algorithms
    - Learn a function F to find a function f^* for new task



$\longrightarrow F \longrightarrow F($  $) = f^*$

Requires few examples

$\downarrow$

$f^*($  $) =$ "giraffe"

Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks

PMLR 2017

Chelsea Finn, Pieter Abbeel, Sergey Levine

Berkeley, OpenAI

Introduction

- Application
  - Model-agnostic
    - Any supervised (classification/regression) & reinforcement learning models learnt by gradient descent.
  - Few-shot scenarios
    - Many tasks
    - Only few examples for the new task
- Idea
  - Find a good global parameter θ that can be adapted to all tasks with few examples
    - θ is a task independent parameter, serving as the initial parameters for all tasks.

Model

- MAML for supervise learning (classification & regression)

Algorithm 2 MAML for Few-Shot Supervised Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters
1: randomly initialize θ
2: **while** not done **do**
3: Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
4: **for all** $\mathcal{T}_i$ **do**
5: Sample K datapoints $\mathcal{D} = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$
6: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ using $\mathcal{D}$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation (2) or (3)
7: Compute adapted parameters with gradient descent: $\theta'_i = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$
8: Sample datapoints $\mathcal{D}'_i = \{\mathbf{x}^{(j)}, \mathbf{y}^{(j)}\}$ from $\mathcal{T}_i$ for the meta-update
9: **end for**
10: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta'_i})$ using each $\mathcal{D}'_i$ and $\mathcal{L}_{\mathcal{T}_i}$ in Equation 2 or 3
11: **end while**

A batch of tasks $\mathcal{T}_0, \ldots, \mathcal{T}_i, \ldots, \mathcal{T}_N$

K: Shot num

Feed forward of base model f_θ on support set, get gradients

Update θ for one step

Obtain samples for query set

Update θ based on the performance of θ'_i on query set $\mathcal{D}'_i$

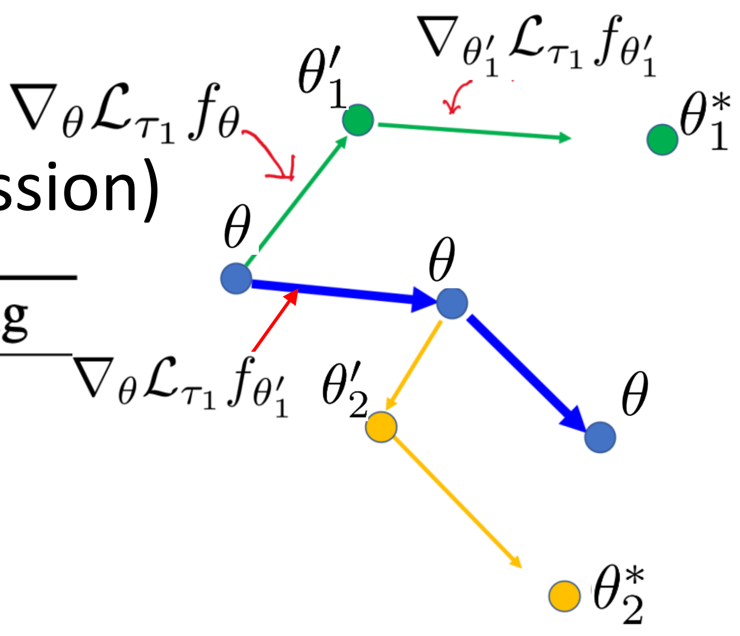
$\mathcal{D}$: Support set

$\mathcal{T}_i$ is a task

Equ2: mean sqaure loss for regression;
Equ3: cross entropy loss for classification

Model

- MAML for supervise learning (classification & regression)



Algorithm 2 MAML for Few-Shot Supervised Learning

Require: $p(\mathcal{T})$: distribution over tasks
Require: α, β: step size hyperparameters

```
randomly initialize θ
while not done do
    Sample batch of tasks T_i ~ p(T)
    for all T_i do
        Sample K datapoints D = {x^(j), y^(j)} from T_i
        Evaluate ∇_θ L_{T_i}(f_θ) using D and L_{T_i} in Equation (2) or (3)
        Compute adapted parameters with gradient descent: θ'_i = θ − α∇_θ L_{T_i}(f_θ)
        Sample datapoints D'_i = {x^(j), y^(j)} from T_i for the meta-update
    end for
    Update θ ← θ − β∇_θ Σ_{T_i~p(T)} L_{T_i}(f_{θ'_i}) using each D'_i and L_{T_i} in Equation 2 or 3
end while
```

A batch of tasks $\mathcal{T}_0, \ldots, \mathcal{T}_i, \ldots, \mathcal{T}_N$

K: Shot num

$\mathcal{D}$: Support set

$\mathcal{T}_i$ is a task

Feed forward of base model f_θ on support set, get gradients

Update θ for one step

Obtain samples for query set

Update θ based on the performance of θ'_i on query set $\mathcal{D}'_i$

Equ2: mean sqaure loss for regression;
Equ3: cross entropy loss for classification

Learning to Customize Model Structures for Few-shot Dialogue Generation Tasks

Yiping Song, Zequn Liu, Wei Bi, Rui Yan, Ming Zhang
Department of Computer Science, School of EECS, Peking University
Tencent AI Lab, Shenzhen, China
Wangxuan Institute of Computer Technology, Peking University

北京大學 PEKING UNIVERSITY

Few-shot Text Generation

- Cold start in text generation tasks
- Multi-language machine translation
- Personalized dialogue generation
- Emotional dialogue generation

Methods for Few-shot Dialogue Generation

- W/ task description
  - explicit: user profile
  - implicit: user description
- Pre-train
  - from non-target domain to target domain
  - require sufficient data for fine-tuning

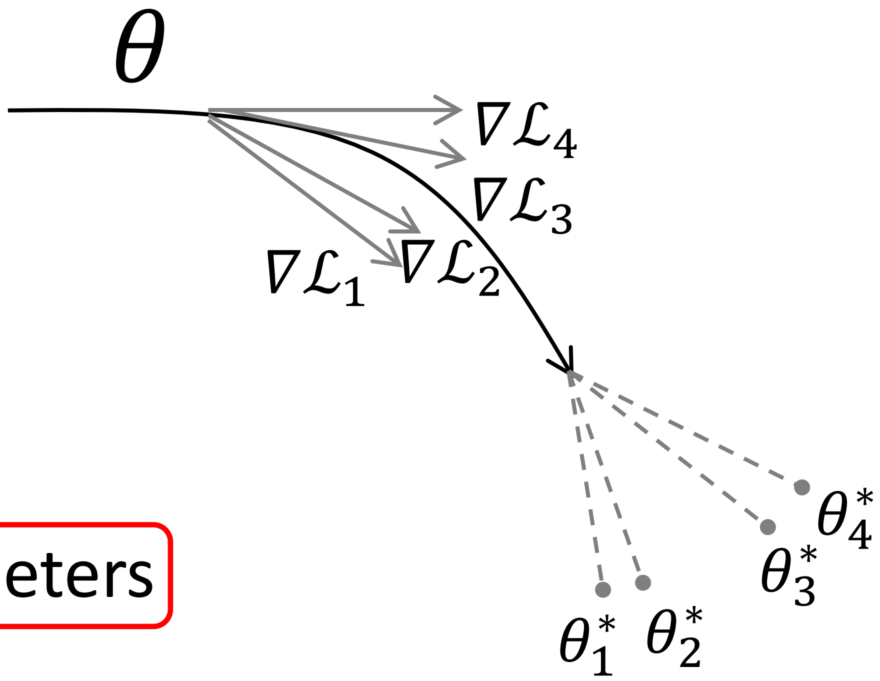
Methods for Few-shot Dialogue Generation

- W/O task description
  - meta-learning

- Meta-learning
  - metric-based methods -> classification
  - model-based methods -> classification
  - optimization-based methods -> model agnostic
  - MAML (model agnostic meta-learning)

MAML

- Training
  - find an initialization of all tasks
- Testing:
  - fine-tuning
- Model = model structure + $\boxed{\text{model parameters}}$



How about the model structure?

Adjust MAML for Larger Model Diversity

- Customize models
  - different network structures + parameters

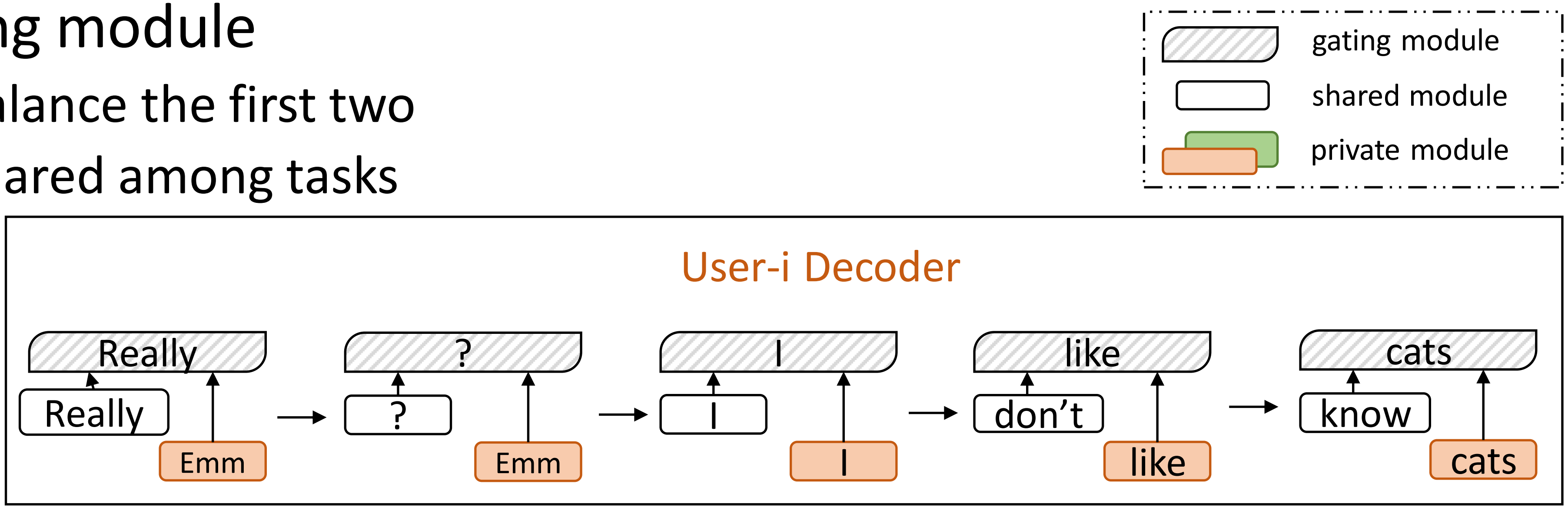
- Unique model structure
  - memorize task characteristics

- Few-shot setting
  - do not require extra data

Customized Model Agnostic Meta-Learning algorithm (CMAML)

The Dialogue Model

- Shared module
  - general generation ability
  - seq2seq
  - shared among tasks
- Gating module
  - balance the first two
  - shared among tasks
- Private module
  - memorize the characteristics of the task
  - multi-layer perception
  - unique for each task



Seq2SPG

Training Overview

- Pre-training
  - MAML: meta-training & meta-testing
  - models are the same

- Customized Model Training
  - private network pruning
    - differentiate the MLP structure
  - joint meta learning
    - re-train 3 modules of each task together

Customized Model Training

- Private network pruning
  - only on private module
- L-1 regularization
  - make parameters sparse
- Up-to-bottom pruning
  - upper layers have been pruned
  - keep edges of current layer whose weight > threshold
  - a node is pruned, nodes connected to it are pruned

Algorithm 1: Private Network Pruning

Input: All parameters θ^p in the private MLP module, the sparsity threshold γ, the total number of layers L in the private MLP module.

Output: The pruned parameters θ_i^p in private module for task T_i.

Finetune θ^p on the training data of T_i with L-1 regularization to otain θ_i^p.

for $j \in \{1, \dots, L\}$ **do**
- $E_j \leftarrow$ All edges (i.e. parameters w.r.t. each edge) in the j-th layer in θ_i^p
- $N_j \leftarrow$ All nodes in the j-th layer in θ_i^p

$E_{keep} \leftarrow E_{|L|} \cup E_1$;

$k \leftarrow |L| - 1; N_{keep} \leftarrow N_{|L|} \cup N_1$.

while $k > 1$ **do**
- **for** *each edge e in E_k* **do**
  - **if** $e > \gamma$ *and the node connected with e in N_{k+1} is in N_{keep}* **then**
    - $E_{keep} \leftarrow E_{keep} \cup \{e\}$.
- **for** *each node n in N_k* **do**
  - **for** *each edge e in E_k connected with n* **do**
    - **if** *e in E_{keep}* **then**
      - $N_{keep} \leftarrow N_{keep} \cup \{n\}$;
      - break.
- $k \leftarrow k - 1$.

return E_{keep} *as* θ_i^p

Customized Model Training

- Joint meta-training
  - joint training for all modules of tasks
  - start from the pre-trained MAML initialization
- Shared & Gating module
  - trained with all training data
- Private module
  - trained with task-specific data
  - no enough data for training

Algorithm 2: Customized Model Training

Input: The distribution over the task set $p(\mathcal{T})$, the step size α and β.

Output: The customized dialogue models $\theta^s \cup \theta_i^p \cup \theta^g$ for every task T_i.

for *each* T_i *in* T **do**

  $\theta_i^p \leftarrow$ Private_Network_Pruning(T_i).

while *not converge* **do**

  Sample a batch of tasks $T_i \sim p(\mathcal{T})$.

  for *each sampled task* T_i **do**

    Adapt θ^s / θ^g to $\theta_i'^s / \theta_i'^g$ with D_i^{train} using Eq. 2;

    Adapt θ_i^p to $\theta_i'^p$ with D_i^{train} using Eq. 4.

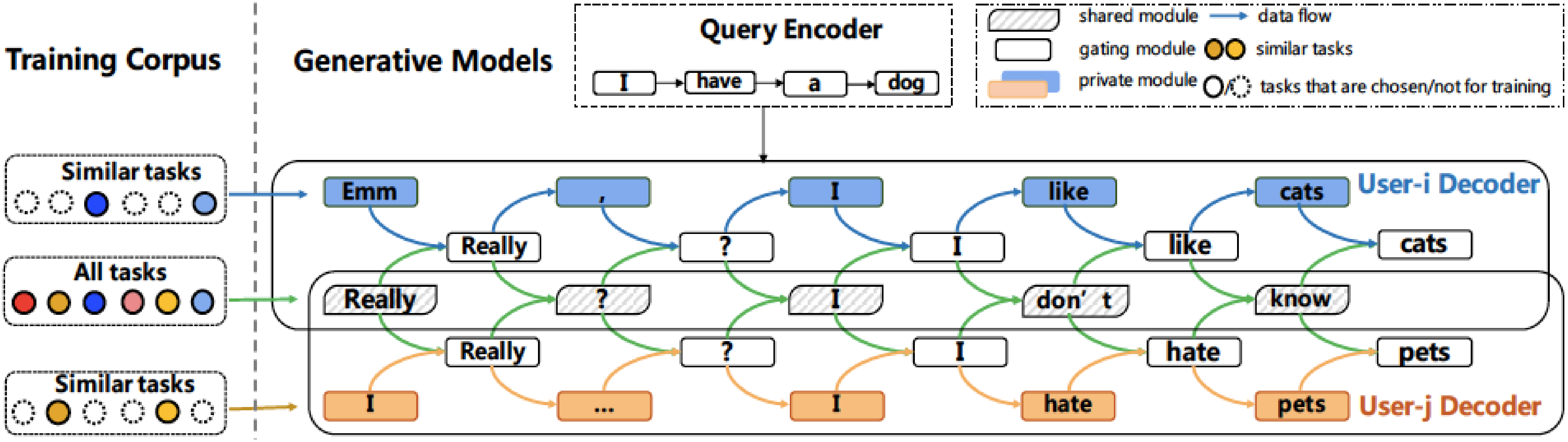
  Update θ^s, θ^g with D_i^{valid} using Eq. 3.

  Update θ_i^p with D_i^{valid} using Eq. 5.

return $\theta^s \cup \theta_i^p \cup \theta^g$

Customized Model Training

- Similar tasks share partial networks
- Dissimilar tasks have no overlaps



CMAML

Experiments

- Persona-chat
  - 1137/99/100 users for training/validation/evaluation
  - each user has 121 utterances

- MojiTalk
  - 50/6/8 emojis for training/validation/evaluation
  - each emoji has 1000 training samples

Competing Methods

- Pretrain-Only
  - Seq2seq
  - Speaker
  - Seq2SPG
- Fine-tune
  - Seq2seq-F
  - Speaker-F
  - Seq2SPG-F
- MAML
  - MAML-Seq2seq
  - MAML-Seq2SPG
- CMAML
  - CMAML-Seq2SP’G
  - CMAML-Seq2SPG

Evaluation Metrics

- Response quality/diversity
  - BLEU
  - PPL
  - Distinct-1
- Task consistency
  - C score
  - E-acc
- Model difference
  - Diff Score
  - Δ Score
  - $D(T_i, T_j) = \frac{||\theta_i - \theta_j||^2}{M}$
- Human Evaluation
  - quality
  - task consistency

Results

Fine-tune > Pre-train

Method	Human Evaluation		Automatic Metrics				Model Difference	
	Quality	Task Consistency	PPL	BLEU	Dist-1	C score/E-acc	Diff Score	Δ Score
Persona-Chat								
Seq2seq	0.67	0.10	37.91	1.27	0.0019	-0.16	0.00	0.00
Speaker	0.85	0.10	40.17	1.25	0.0037	-0.14	0.00	0.00
Seq2SPG	0.67	0.03	36.46	1.41	0.0023	-0.14	0.00	0.00
Seq2seq-F	0.78	0.11	33.65	1.56	0.0046	-0.05	17.97	9.19
Speaker-F	0.87	0.25	35.61	1.52	0.0059	0.03	285.11	143.90
Seq2SPG-F	0.7	0.07	**32.68**	1.54	0.0045	-0.05	292.85	156.30
MAML-Seq2seq	0.97	0.37	37.43	1.54	0.0087	0.14	134.01	67.79
MAML-Seq2SPG	0.85	0.36	35.89	**1.70**	0.0074	0.16	401.28	198.90
CMAML-Seq2SP′G	0.98	0.58	37.32	1.43	0.0089	0.15	479.21	238.64
CMAML-Seq2SPG	**1.15**	**0.69**	36.30	**1.70**	**0.0097**	**0.18**	**514.44**	**263.82**
MojiTalk								
Seq2seq	0.56	0.39	218.95	0.36	0.0342	0.73	0.00	0.00
Speaker	0.38	0.26	418.96	0.19	**0.0530**	0.70	0.00	0.00
Seq2SPG	0.77	0.46	158.74	0.64	0.0239	0.74	0.00	0.00
Seq2seq-F	0.50	0.35	217.60	0.40	0.0326	0.72	15.96	8.88
Speaker-F	0.39	0.25	403.92	0.21	0.0528	0.72	39.08	29.10
Seq2SPG-F	0.76	0.47	**157.92**	0.65	0.0228	0.74	72.43	40.94
MAML-Seq2seq	0.66	0.29	179.02	0.54	0.0109	0.70	183.05	117.09
MAML-Seq2SPG	0.71	0.40	181.56	0.73	0.0246	0.74	306.40	176.31
CMAML-Seq2SP′G	0.64	0.32	172.92	0.76	0.0102	0.75	142.90	81.15
CMAML-Seq2SPG	**0.78**	**0.49**	185.97	**0.85**	0.0210	**0.77**	**345.42**	**190.64**

Overall performance

Results

MAML > Fine-tune

Method	Human Evaluation		Automatic Metrics				Model Difference	
	Quality	Task Consistency	PPL	BLEU	Dist-1	C score/E-acc	Diff Score	Δ Score
Persona-Chat								
Seq2seq	0.67	0.10	37.91	1.27	0.0019	-0.16	0.00	0.00
Speaker	0.85	0.10	40.17	1.25	0.0037	-0.14	0.00	0.00
Seq2SPG	0.67	0.03	36.46	1.41	0.0023	-0.14	0.00	0.00
Seq2seq-F	0.78	0.11	33.65	1.56	0.0046	-0.05	17.97	9.19
Speaker-F	0.87	0.25	35.61	1.52	0.0059	0.03	285.11	143.90
Seq2SPG-F	0.7	0.07	**32.68**	1.54	0.0045	-0.05	292.85	156.30
MAML-Seq2seq	0.97	0.37	37.43	1.54	0.0087	0.14	134.01	67.79
MAML-Seq2SPG	0.85	0.36	35.89	**1.70**	0.0074	0.16	401.28	198.90
CMAML-Seq2SP'G	0.98	0.58	37.32	1.43	0.0089	0.15	479.21	238.64
CMAML-Seq2SPG	**1.15**	**0.69**	36.30	**1.70**	**0.0097**	**0.18**	**514.44**	**263.82**
MojiTalk								
Seq2seq	0.56	0.39	218.95	0.36	0.0342	0.73	0.00	0.00
Speaker	0.38	0.26	418.96	0.19	**0.0530**	0.70	0.00	0.00
Seq2SPG	0.77	0.46	158.74	0.64	0.0239	0.74	0.00	0.00
Seq2seq-F	0.50	0.35	217.60	0.40	0.0326	0.72	15.96	8.88
Speaker-F	0.39	0.25	403.92	0.21	0.0528	0.72	39.08	29.10
Seq2SPG-F	0.76	0.47	**157.92**	0.65	0.0228	0.74	72.43	40.94
MAML-Seq2seq	0.66	0.29	179.02	0.54	0.0109	0.70	183.05	117.09
MAML-Seq2SPG	0.71	0.40	181.56	0.73	0.0246	0.74	306.40	176.31
CMAML-Seq2SP'G	0.64	0.32	172.92	0.76	0.0102	0.75	142.90	81.15
CMAML-Seq2SPG	**0.78**	**0.49**	185.97	**0.85**	0.0210	**0.77**	**345.42**	**190.64**

Overall performance

Results

CMAML > MAML

Method	Human Evaluation		Automatic Metrics				Model Difference	
	Quality	Task Consistency	PPL	BLEU	Dist-1	C score/E-acc	Diff Score	Δ Score
Persona-Chat								
Seq2seq	0.67	0.10	37.91	1.27	0.0019	-0.16	0.00	0.00
Speaker	0.85	0.10	40.17	1.25	0.0037	-0.14	0.00	0.00
Seq2SPG	0.67	0.03	36.46	1.41	0.0023	-0.14	0.00	0.00
Seq2seq-F	0.78	0.11	33.65	1.56	0.0046	-0.05	17.97	9.19
Speaker-F	0.87	0.25	35.61	1.52	0.0059	0.03	285.11	143.90
Seq2SPG-F	0.7	0.07	**32.68**	1.54	0.0045	-0.05	292.85	156.30
MAML-Seq2seq	0.97	0.37	37.43	1.54	0.0087	0.14	134.01	67.79
MAML-Seq2SPG	0.85	0.36	35.89	**1.70**	0.0074	0.16	401.28	198.90
CMAML-Seq2SP′G	0.98	0.58	37.32	1.43	0.0089	0.15	479.21	238.64
CMAML-Seq2SPG	**1.15**	**0.69**	36.30	**1.70**	**0.0097**	**0.18**	**514.44**	**263.82**
MojiTalk								
Seq2seq	0.56	0.39	218.95	0.36	0.0342	0.73	0.00	0.00
Speaker	0.38	0.26	418.96	0.19	**0.0530**	0.70	0.00	0.00
Seq2SPG	0.77	0.46	158.74	0.64	0.0239	0.74	0.00	0.00
Seq2seq-F	0.50	0.35	217.60	0.40	0.0326	0.72	15.96	8.88
Speaker-F	0.39	0.25	403.92	0.21	0.0528	0.72	39.08	29.10
Seq2SPG-F	0.76	0.47	**157.92**	0.65	0.0228	0.74	72.43	40.94
MAML-Seq2seq	0.66	0.29	179.02	0.54	0.0109	0.70	183.05	117.09
MAML-Seq2SPG	0.71	0.40	181.56	0.73	0.0246	0.74	306.40	176.31
CMAML-Seq2SP′G	0.64	0.32	172.92	0.76	0.0102	0.75	142.90	81.15
CMAML-Seq2SPG	**0.78**	**0.49**	185.97	**0.85**	0.0210	**0.77**	**345.42**	**190.64**

Overall performance

Results

Method	100-shot			110-shot		
	PPL	BLEU	C score	PPL	BLEU	C score
Seq2seq	38.13	1.19	-0.11	37.58	1.29	-0.15
Speaker	40.95	1.02	-0.25	42.59	1.27	-0.06
Seq2SPG	39.75	1.27	-0.10	37.71	1.30	-0.15
Seq2seq-F	**34.86**	1.39	-0.03	**34.14**	1.52	-0.10
Speaker-F	37.11	1.30	-0.16	39.10	1.36	-0.06
Seq2SPG-F	37.19	1.31	0.00	37.00	1.33	-0.15
MAML-Seq2seq	36.94	1.47	0.03	37.20	1.53	0.07
MAML-Seq2SPG	36.50	1.52	0.11	35.98	1.47	0.13
CMAML-Seq2SP′G	37.18	1.46	0.11	37.08	1.44	0.09
CMAML-Seq2SPG	36.52	**1.52**	**0.14**	36.44	**1.57**	**0.15**

Different few-shot settings

Pre-train & Fine-tune: data ↑ task consistency =

MAML & CMAML: data ↑ task consistency ↑

Results

Method	Similar Users			Dissimilar Users		
	PPL	BLEU	C score	PPL	BLEU	C score
Seq2seq	76.54	1.49	-0.03	42.87	1.10	-0.10
Speaker	162.44	0.65	-0.09	46.86	1.11	-0.13
Seq2SPG	73.58	1.32	-0.04	42.21	1.14	-0.22
Seq2seq-F	74.53	1.53	-0.07	42.33	1.33	-0.06
Speaker-F	103.81	1.04	**0.04**	40.47	1.40	0.01
Seq2SPG-F	**70.15**	1.44	-0.04	**36.22**	1.35	-0.05
MAML-Seq2seq	83.17	1.52	-0.08	39.67	1.34	0.06
MAML-Seq2SPG	82.37	1.52	-0.06	39.41	1.41	0.12
CMAML-Seq2SP′G	82.56	1.50	0.00	40.50	1.40	0.13
CMAML-Seq2SPG	82.78	**1.56**	-0.07	39.55	**1.43**	**0.16**

Different task consistency settings

Pre-train & Fine-tune: similar tasks

MAML & CMAML: dissimilar tasks

Summary

- Customize models
  - different network structures & parameters
  - hundreds of training samples in each task
- Unique structure
  - memorize characteristics of each task
  - similar tasks are sharing data in the view of model structure
- Generation tasks
  - applicable to all few-shot generation tasks
- Code is available at: https://github.com/zequnl/CMAML

Hierarchically Structured Meta-learning

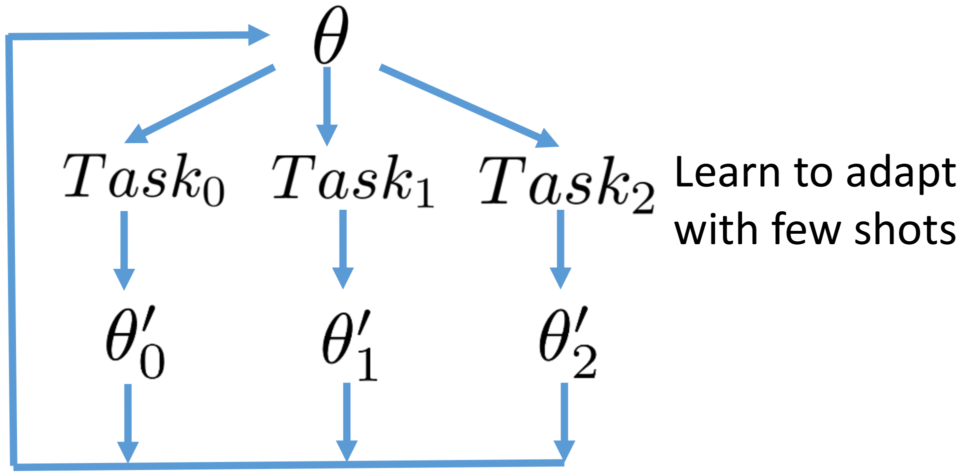
ICML 2019

Huaxiu Yao, Ying Wei, Junzhou Huang, Zhenhui Li

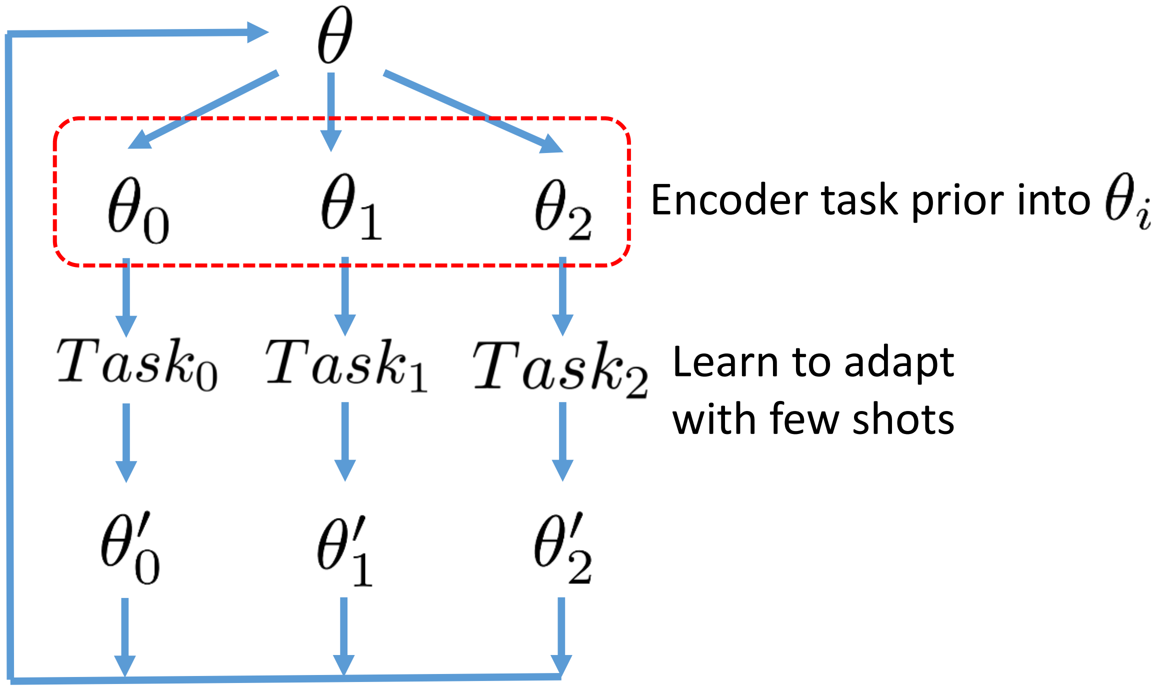
Pennsylvania, Tencent AI lab

Introduction

- Idea
  - Import task priors for MAML
  - Learn to represent tasks
  - Encode the task representation to the initial parameter θ of MAML



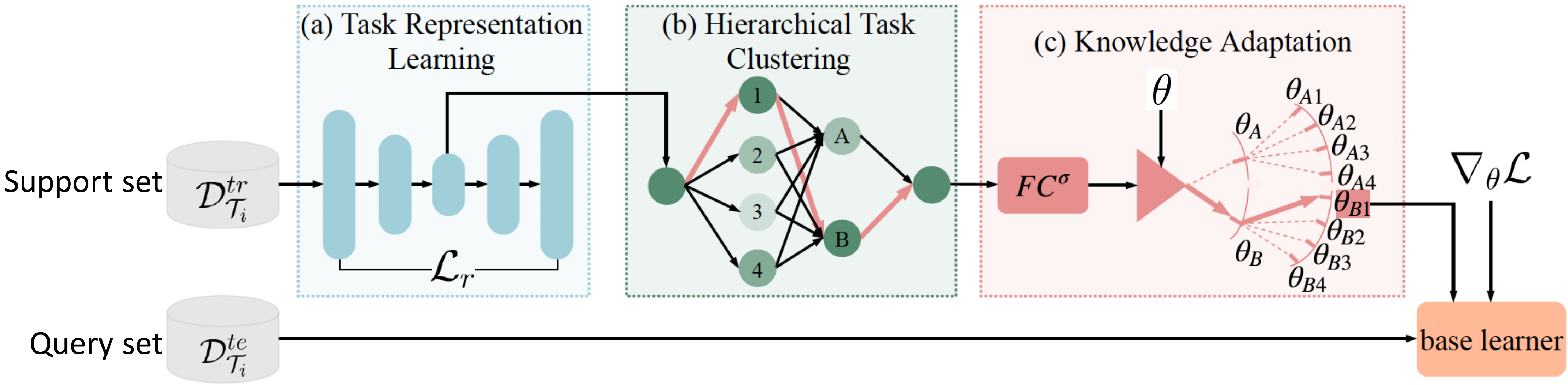
MAML



HSML

Model

- Step1 Task representation learning
- Step2 Hierarchical soft task clustering
- Step3 Adaptation on tasks



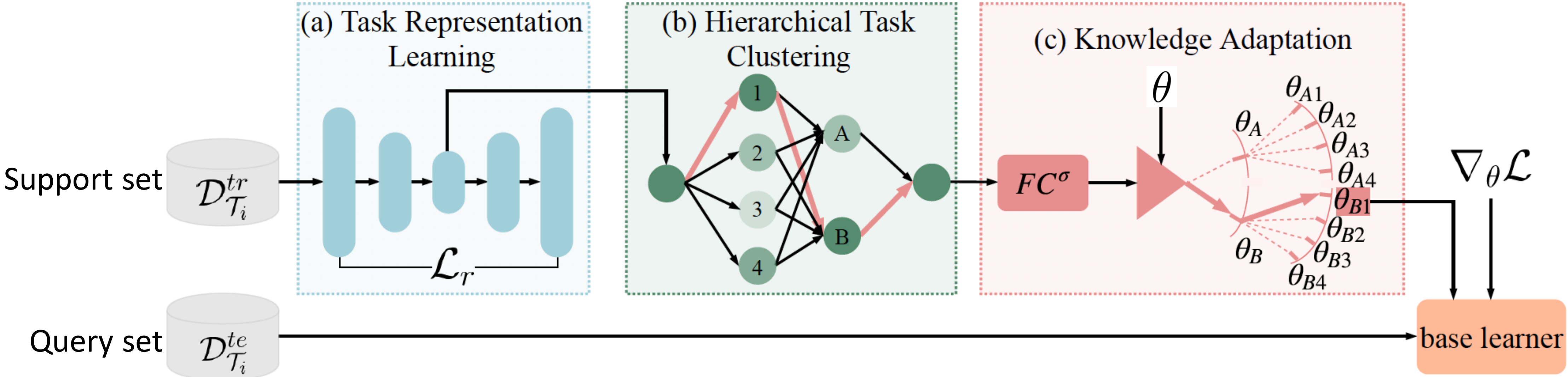
Model

- Step1 Task representation learning
  - Step1.1 train sample-level representation with autoencoder
    - Embed input x and output y (with CNN) $\mathcal{F}(\mathbf{x}_{i,j}^{tr}, \mathbf{y}_{i,j}^{tr})$
    - Reconstruct $\mathcal{F}(\mathbf{x}_{i,j}^{tr}, \mathbf{y}_{i,j}^{tr})$ with autoencoder models
      - FC- based (Fully Connected Layer)
      - RNN-based: treat all samples in a task as a sequence
      - Train by square loss

$$\mathbf{g}_{i,j} = \mathrm{FC}_{enc}(\mathcal{F}(\mathbf{x}_{i,j}^{tr}, \mathbf{y}_{i,j}^{tr}))$$

$$\mathcal{L}_r(\mathcal{D}_{\mathcal{T}_i}^{tr}) = \sum_{j=1}^{n^{tr}} \|\mathrm{FC}_{dec}(\mathbf{g}_{i,j}) - \mathcal{F}(\mathbf{x}_{i,j}^{tr}, \mathbf{y}_{i,j}^{tr})\|_2^2$$

  - Step1.2 aggregate sample-level representation to task representation
    - Merge by max/mean Pooling $\mathbf{g}_i = \mathrm{Pool}_{j=1}^{n^{tr}}(\mathbf{g}_{i,j})$



Model

- Step2 Hierarchical soft clustering on tasks
  - Step2.1 Assignment (Soft)
    - Prob of task i transferring from cluster k^l (at layer l) to cluster k^{l+1} (at layer $l+1$)

$$p_i^{k^l \to k^{l+1}} = \frac{\exp\left(-\|(\mathbf{h}_i^{k^l} - \mathbf{c}_{k^{l+1}})/\sigma^l\|_2^2/2\right)}{\sum_{k^{l+1}=1}^{K^{l+1}} \exp\left(-\|(\mathbf{h}_i^{k^l} - \mathbf{c}_{k^{l+1}})/\sigma^l\|_2^2/2\right)}$$

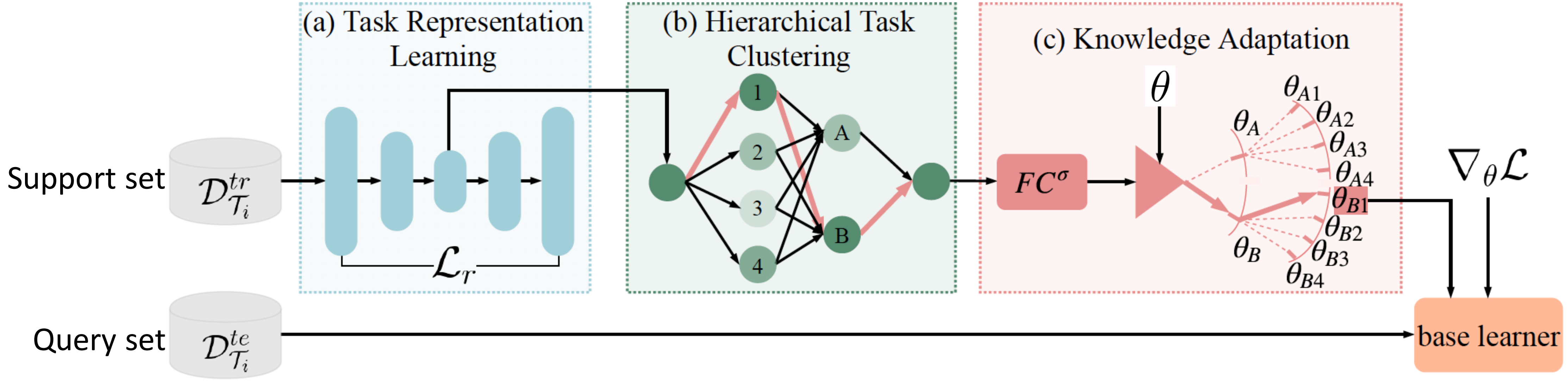
$\mathbf{h}_i^{k^l}$ Task representation of task i in cluster k^l at layer l

$\mathbf{c}_{k^{l+1}}$ Representation of cluster center k^{l+1} at layer $l+1$ (learnable)

  - Step2.2 Update
    - Update task representations
      - Weighted aggregate vectors from lower layers

$$\mathbf{h}_i^{k^{l+1}} = \sum_{k^l=1}^{K^l} p_i^{k^l \to k^{l+1}} \tanh\left(\mathbf{W}^{k^{l+1}} \mathbf{h}_i^{k^l} + \mathbf{b}^{k^{l+1}}\right)$$

$$\mathbf{h}_i^{k^0} = \mathbf{g}_i$$



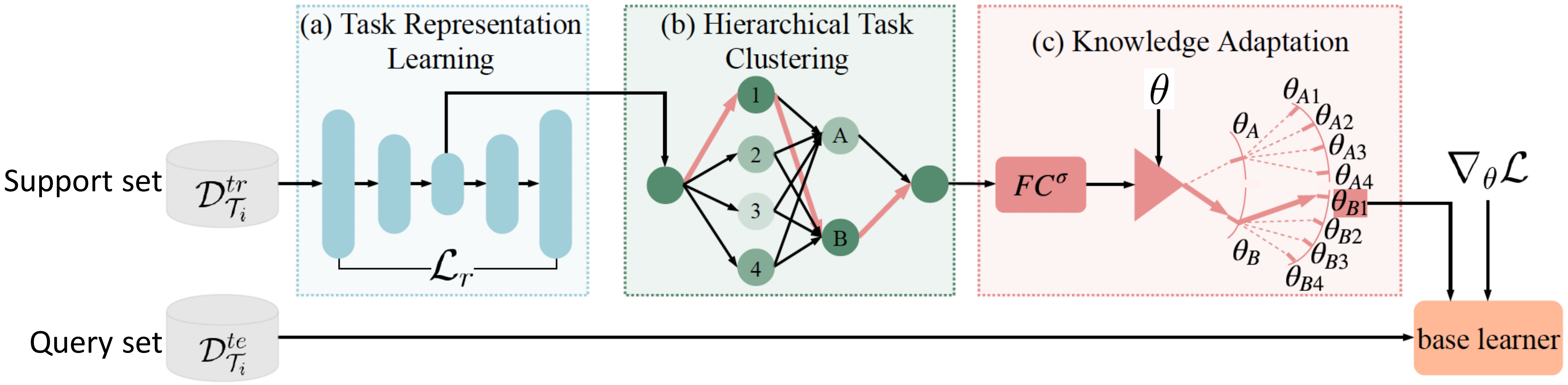
Model

- Step3 Adaptation on tasks
  - FC (Task embedding $\mathbf{g}_i$ + task representation $\mathbf{h}_i^L$ at top clustering layer)

$$\mathbf{o}_i = \mathrm{FC}^{\sigma}_{\mathbf{W}_g}(\mathbf{g}_i \oplus \mathbf{h}_i^L), \qquad \theta_{0i} = \theta \circ \mathbf{o}_i$$

- Loss
  - Meta-learning Loss as MAML + Reconstruction loss at step1

$$\min_{\Theta} \sum_{i=1}^{N_t} \mathcal{L}(f_{\theta_{0i} - \alpha \nabla_\theta \mathcal{L}(\theta, \mathcal{D}_{\mathcal{T}_i}^{tr})}, \mathcal{D}_{\mathcal{T}_i}^{te}) + \xi \mathcal{L}_r(\mathcal{D}_{\mathcal{T}_i}^{tr})$$



Optimization as a model for few-shot learning

ICLR 2017

Sachin Raviand, Hugo Larochelle

Twitter

Introduction

- Motivation (Similar to MAML based)
  - Obtain task-specific model parameters θ_i for new task with few examples
- Idea
  - Treats samples in a task as a sequence
  - LSTM learns to generate model parameters from a sequence of samples

Introduction

- Motivation
  - Obtain task-specific model parameters θ_i for new task with few examples
- Idea
  - Treats samples in a task as a sequence
  - LSTM learns to generate model parameters from a sequence of samples
  - Gradient descent

    $$\theta_t = \theta_{t-1} - \alpha_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

    - Few shot: make full use of few samples
  - LSTM (only use input and forget gates)

    $$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$$

    - LSTM balances previous information and current input
  - LSTM+GD
    - LSTM balances previous parameters and current gradients

Introduction

- Motivation
  - Obtain task-specific model parameters θ_i for new task with few examples
- Idea
  - Treats samples in a task as a sequence
  - LSTM learns to generate model parameters from a sequence of samples
  - Gradient descent

    $$\theta_t = \theta_{t-1} - \alpha_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

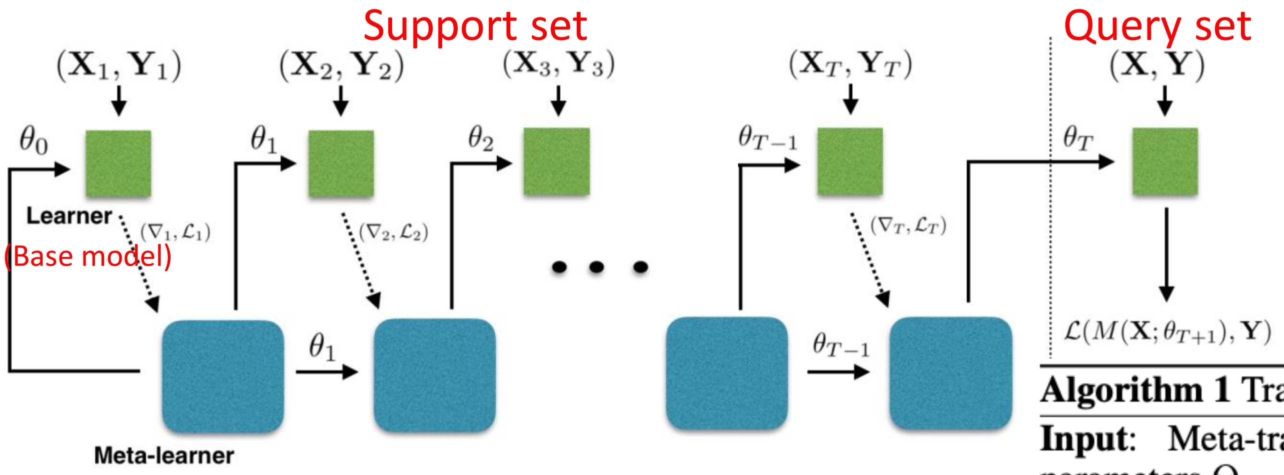
    - Few shot: make full use of few samples
  - LSTM (only use input and forget gates)

    $$c_t = \overset{=1}{f_t} \odot c_{t-1} + i_t \odot \tilde{c}_t$$

    - LSTM balances previous information and current input
  - LSTM+GD
    - LSTM balances previous parameters and current gradients
    - learn to generate θ_t from gradient $\nabla_{\theta_{t-1}} \mathcal{L}_t$ and previous θ_{t-1}

    $$\theta_t = f_t \theta_{t-1} + i_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

Model



Meta-learner

$$\theta_t = f_t \theta_{t-1} + i_t \nabla_{\theta_{t-1}} \mathcal{L}_t$$

$$i_t = \sigma\left(\mathbf{W}_I \cdot \left[\nabla_{\theta_{t-1}} \mathcal{L}_t, \mathcal{L}_t, \theta_{t-1}, i_{t-1}\right] + \mathbf{b}_I\right)$$

$$f_t = \sigma\left(\mathbf{W}_F \cdot \left[\nabla_{\theta_{t-1}} \mathcal{L}_t, \mathcal{L}_t, \theta_{t-1}, f_{t-1}\right] + \mathbf{b}_F\right)$$

Algorithm 1 Train Meta-Learner

Input: Meta-training set $\mathscr{D}_{meta-train}$, Learner M with parameters θ, Meta-Learner R with parameters Θ.

```
Θ_0 ← random initialization

for d = 1, n do
    D_train, D_test ← random dataset from 𝒟_meta−train
    θ_0 ← c_0                                          ▷ Intialize learner parameters

    for t = 1, T do
        X_t, Y_t ← random batch from D_train
        ℒ_t ← ℒ(M(X_t; θ_{t−1}), Y_t)                  ▷ Get loss of learner on train batch
        c_t ← R((∇_{θ_{t−1}} ℒ_t, ℒ_t); Θ_{d−1})       ▷ Get output of meta-learner using Equation 2
        θ_t ← c_t                                      ▷ Update learner parameters
    end for

    X, Y ← D_test
    ℒ_test ← ℒ(M(X; θ_T), Y)                           ▷ Get loss of learner on test batch
    Update Θ_d using ∇_{Θ_{d−1}} ℒ_test                ▷ Update meta-learner parameters

end for
```

Support set
Query set
Task
Training set
Sample a batch from support set of this task
Base model
Meta-learner
All samples in query set compose one batch
Evaluate on query set
Update meta-learner according to the performance on query set

Thanks for listening!