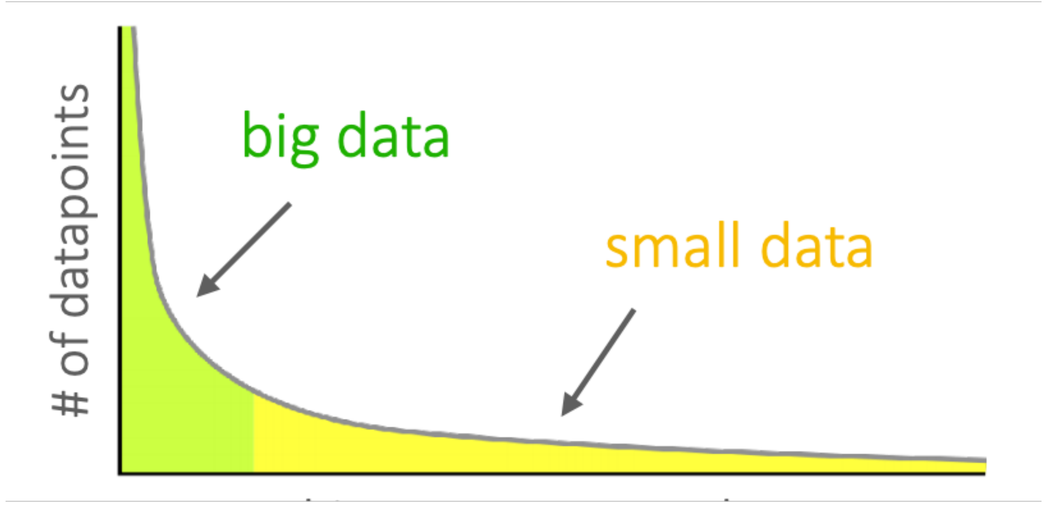
Meta / Few-shot Learning

分享人：邹笑寒

2021.08.25

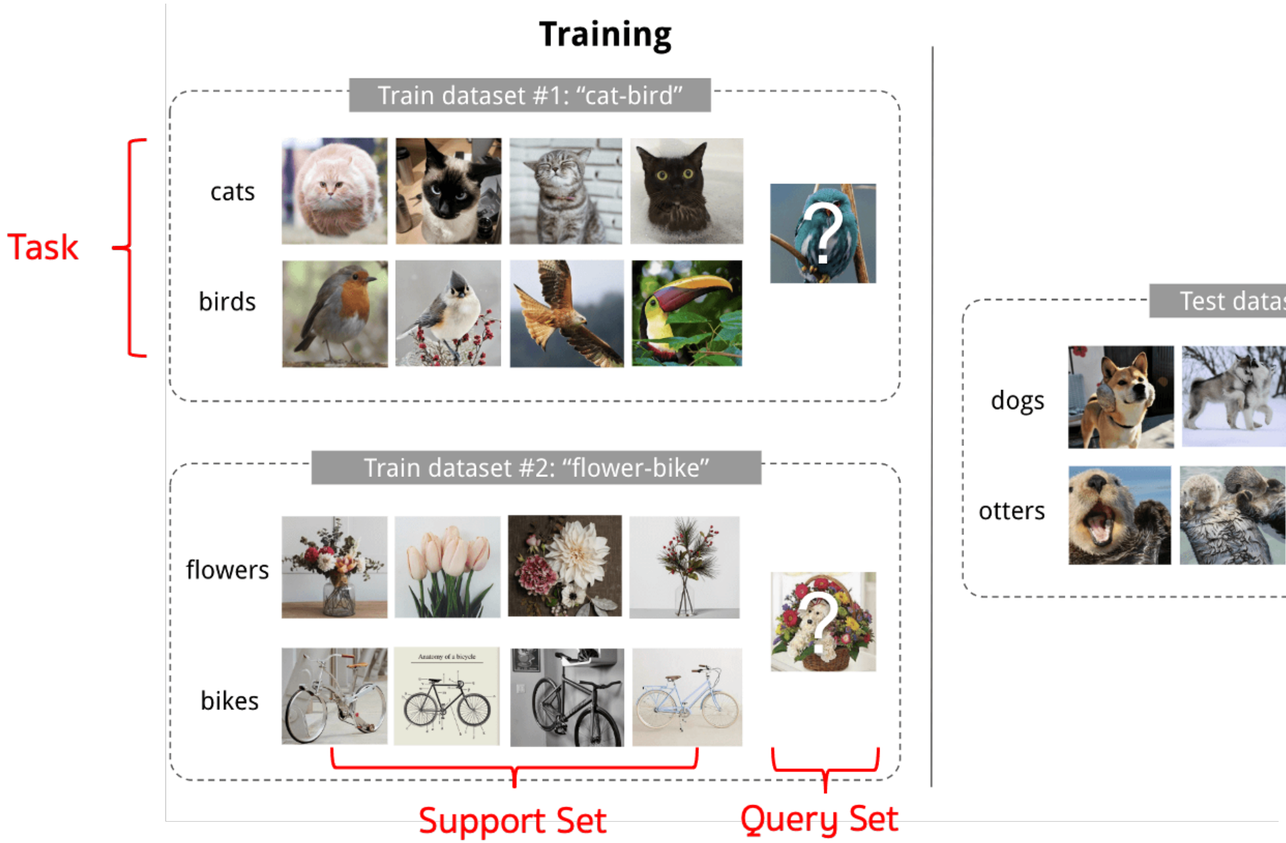
Problem Definition



- task 多
- 每个 task 内的样本少

在已经学习了大量 task 后，如何在新的 task 上快速学习?

Application on Supervised Learning: Few-shot Learning



2-way 4-shot

Supervised Learning:

Inputs: $\mathbf{x}$ Outputs: $\mathbf{y}$ Data: $\mathcal{D} = \{(\mathbf{x}, \mathbf{y})_i\}$

$$\mathbf{y} = f(\mathbf{x}; \theta)$$

Meta-Supervised Learning:

Inputs: $\mathcal{D}^{\text{tr}}$ $\mathbf{x}_{\text{test}}$ Outputs: $\mathbf{y}_{\text{test}}$ Data: $\mathcal{D}_{\text{meta-train}} = \{\mathcal{D}_i\}$

$$\{(\mathbf{x}, \mathbf{y})_{1:K}\}$$

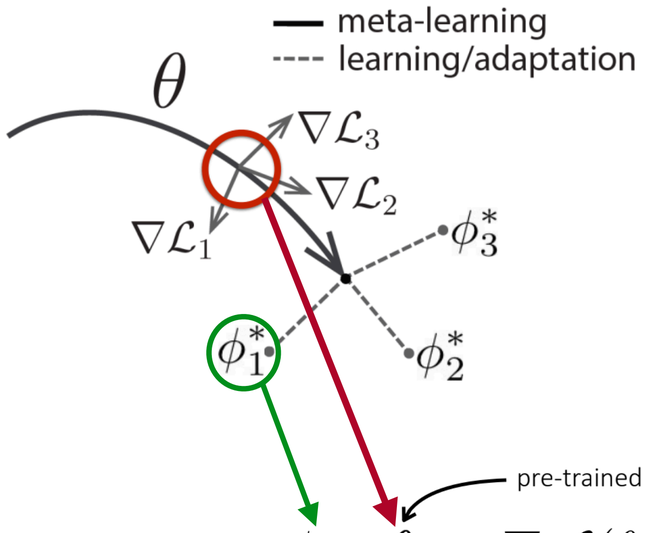
$$\mathbf{y}_{\text{test}} = f(\mathcal{D}^{\text{tr}}, \mathbf{x}_{\text{test}}; \theta)$$

$$\mathcal{D}_i : \{(\mathbf{x}, \mathbf{y})_j\}$$

Popular Approaches

Optimization-based

学习出模型的初始化参数



pre-trained parameters

Fine-tuning [test-time]

$$\phi \leftarrow \theta - \alpha\nabla_\theta\mathcal{L}(\theta, \mathcal{D}^{\text{tr}})$$

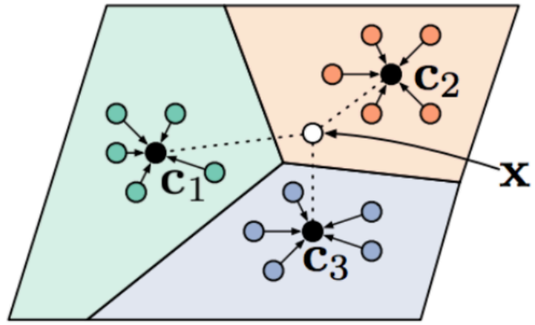
training data for new task

Meta-learning

$$\min_\theta \sum_{\text{task } i} \mathcal{L}(\theta - \alpha\nabla_\theta\mathcal{L}(\theta, \mathcal{D}_i^{\text{tr}}), \mathcal{D}_i^{\text{ts}})$$

Metric-based

学习出样本间的距离函数（聚类）

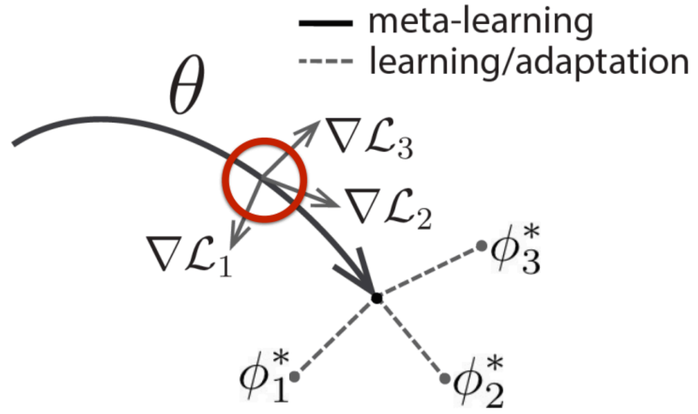


$$y^{\text{ts}} = f_{\text{PN}}(\mathcal{D}_i^{\text{tr}}, x^{\text{ts}})$$
$$= \text{softmax}\left(-d(f_\theta(x), c_k)\right)$$

Others

AutoML?

Optimization-based: MAML



Algorithm 1 Model-Agnostic Meta-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: Evaluate $\nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ with respect to K examples
6: Compute adapted parameters with gradient descent: $\theta_i' = \theta - \alpha \nabla_\theta \mathcal{L}_{\mathcal{T}_i}(f_\theta)$ Inner Loop
7: **end for**
8: Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i \sim p(\mathcal{T})} \mathcal{L}_{\mathcal{T}_i}(f_{\theta_i'})$ Outer Loop
9: **end while**

Bi-level Optimization

Algorithm 2 Generic First-order Model-agnostic Meta-learning algorithm

Require: $p(\tau)$: Distribution over tasks
Require: α, β : step hyper-parameters
Initialisation : Random θ
1: **for** $i = 1, 2...n$ **do**
2: sample tasks $\tau_i \sim p(\tau)$
3: **for** all τ_i **do**
4: Evaluate $\nabla_\theta L(\theta)$ with K examples
5: Compute adapted param with GD: $\theta_i = \theta - \alpha \nabla_\theta L(\theta)$
6: **end for**
7: update: $\theta \leftarrow \theta - \beta \nabla_{\theta_i} L(\theta_i)$
8: **end for**

First-order Version

[1] Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks. https://arxiv.org/abs/1703.03400. ICML 2017.

Optimization-based: Reptile

Algorithm 1 Reptile (serial version)

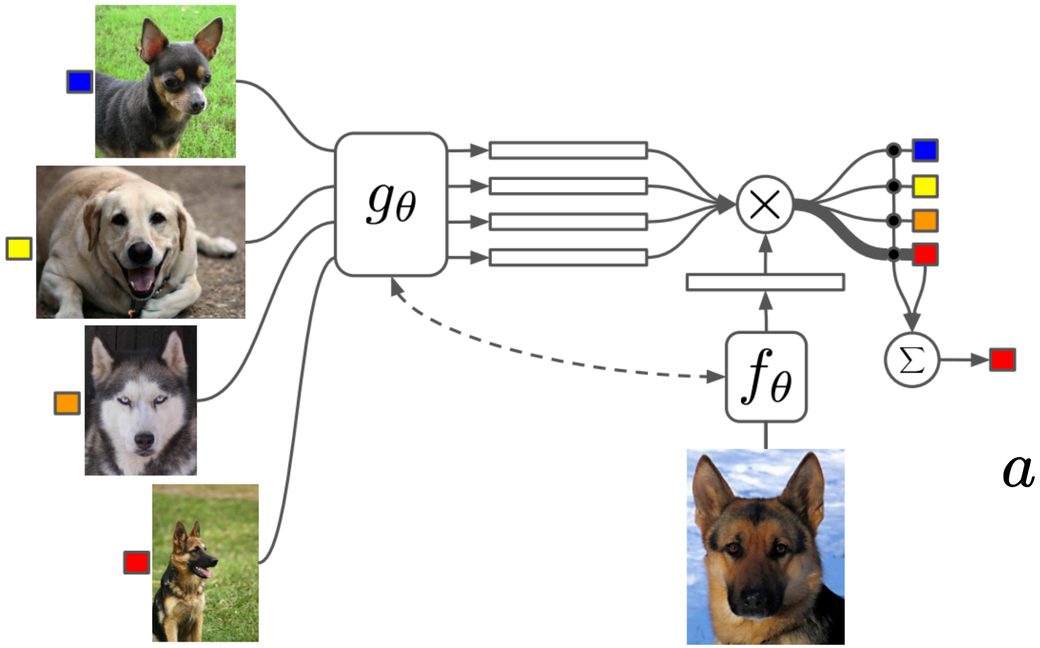
Initialize ϕ, the vector of initial parameters
for iteration $= 1, 2, \ldots$ **do**
  Sample task τ, corresponding to loss L_τ on weight vectors $\widetilde{\phi}$
  Compute $\widetilde{\phi} = U_\tau^k(\phi)$, denoting k steps of SGD or Adam
  Update $\phi \leftarrow \phi + \epsilon(\widetilde{\phi} - \phi)$
end for

$$
\begin{aligned}
g_{\text{MAML}} &= \bar{g}_2 - \alpha \overline{H}_2 \bar{g}_1 - \alpha \overline{H}_1 \bar{g}_2 + O(\alpha^2) \\
g_{\text{FOMAML}} = g_2 &= \bar{g}_2 - \alpha \overline{H}_2 \bar{g}_1 + O(\alpha^2) \\
g_{\text{Reptile}} = g_1 + g_2 &= \bar{g}_1 + \bar{g}_2 - \alpha \overline{H}_2 \bar{g}_1 + O(\alpha^2)
\end{aligned}
$$

增大不同 batch 上的梯度的内积 = 减小夹角

On First-Order Meta-Learning Algorithms. https://arxiv.org/abs/1803.02999. 2018.

Metric-based: Matching Networks

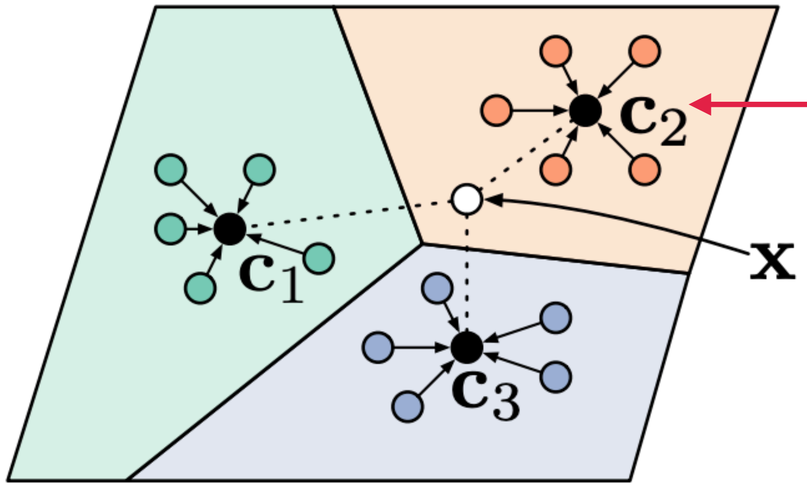


$$P(\hat{y}|\hat{x}, S) = \sum_{i=1}^{k} a(\hat{x}, x_i) y_i$$

$$a(\hat{x}, x_i) = \text{softmax}\Big(- d\big(f_\theta(\hat{x}), g_\theta(x_i)\big)\Big)$$

Matching Networks for One Shot Learning. https://arxiv.org/abs/1606.04080. NIPS 2016.

Metric-based: Prototypical Networks



$$\mathbf{c}_k = \frac{1}{|\mathcal{D}_i^{\text{tr}}|} \sum_{(x,y) \in \mathcal{D}_i^{\text{tr}}} f_\theta(x)$$

$$p_\theta(y = k|x) = \frac{\exp(-d(f_\theta(x), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_\theta(x), \mathbf{c}_{k'}))}$$

Prototypical Networks for Few-shot Learning. https://arxiv.org/abs/1703.05175. NIPS 2017.

How to Enhance Metric-based Methods

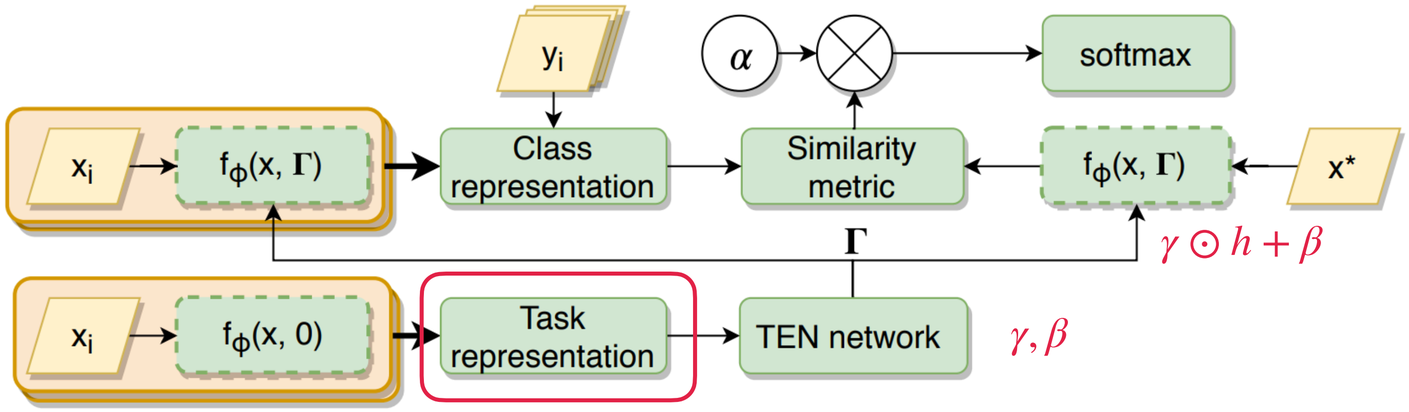
- Feature Extractor
- Prototypes
- Distance Computing
- Additional information (Cross-modal)

Metric-based: TADAM

Learnable temperature

Metric Scaling $$p_{\phi,\alpha}(y = k|\mathbf{x}) = \text{softmax}(-\alpha d(\mathbf{z}, \mathbf{c}_k))$$

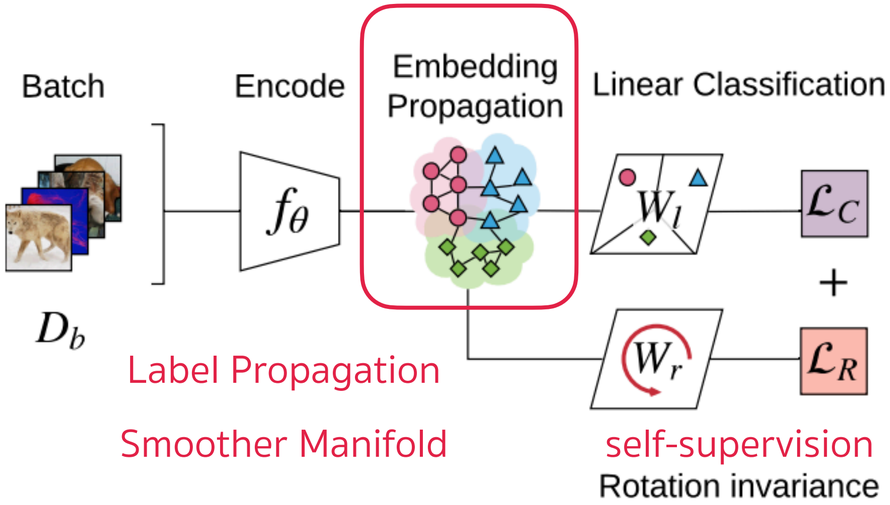
Task conditioning

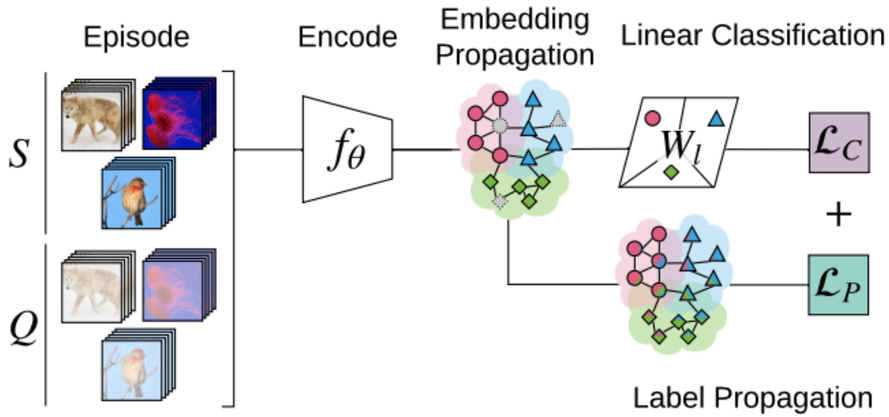


mean of the class prototypes: $\bar{\mathbf{c}} = \frac{1}{K}\sum_k \mathbf{c}_k$

TADAM: Task dependent adaptive metric for improved few-shot learning. https://arxiv.org/abs/1805.10123. NIPS 2018.

Metric-based: EPNet





$$\mathcal{L}_c(\mathbf{x}_i, y_i; W_l, \theta) = -\ln p(y_i|\widetilde{\mathbf{z}}_i, W_l)$$

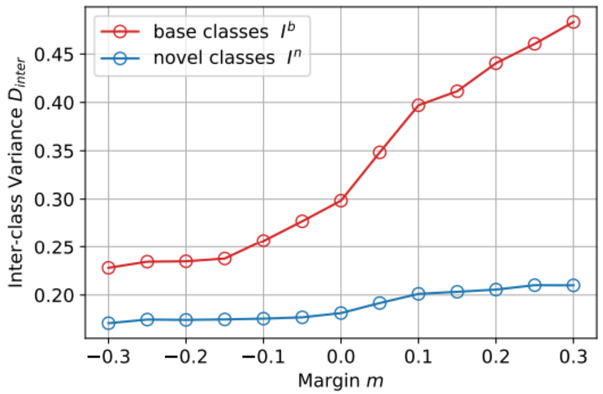
$$\mathcal{L}_r(\mathbf{x}_i, r_j; W_r, \theta) = -\ln p(r_j|\widetilde{\mathbf{z}}_i, W_r)$$

$$\underset{\theta, W_l, W_r}{\operatorname{argmin}} \sum_{i=1}^{128} \sum_{j=1}^{4} \mathcal{L}_c(\mathbf{x}_i, y_i; W_l, \theta) + \mathcal{L}_r(\mathbf{x}_i, r_j; W_r, \theta)$$

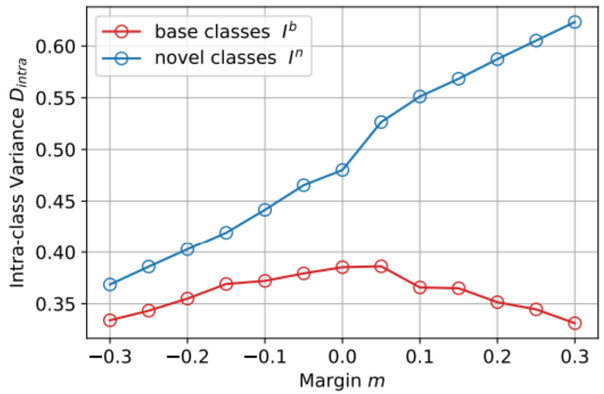
$$\underset{\theta, W_l}{\operatorname{argmin}} \left[\frac{1}{|Q|} \sum_{(\mathbf{x}_i, y_i) \in Q} \mathcal{L}_p(\mathbf{x}_i, y_i; \theta) + \frac{1}{|S \cup Q|} \sum_{(\mathbf{x}_i, y_i) \in S \cup Q} \frac{1}{2} \mathcal{L}_c(\mathbf{x}_i, y_i; W_l, \theta) \right]$$

Embedding Propagation: Smoother Manifold for Few-Shot Classification. https://arxiv.org/abs/2003.04151. ECCV 2020.

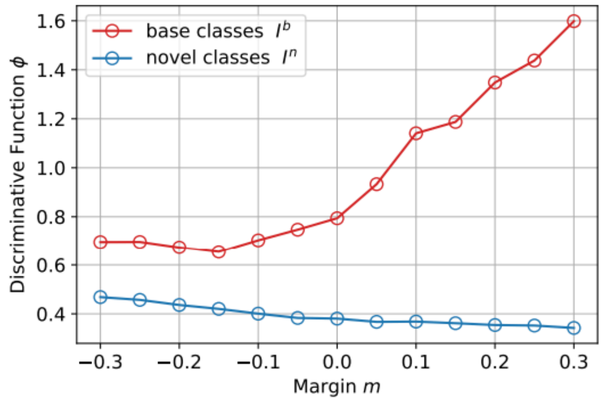
Metric-based: Negative-Margin Softmax Loss



(a) D_{inter}



(b) D_{intra}



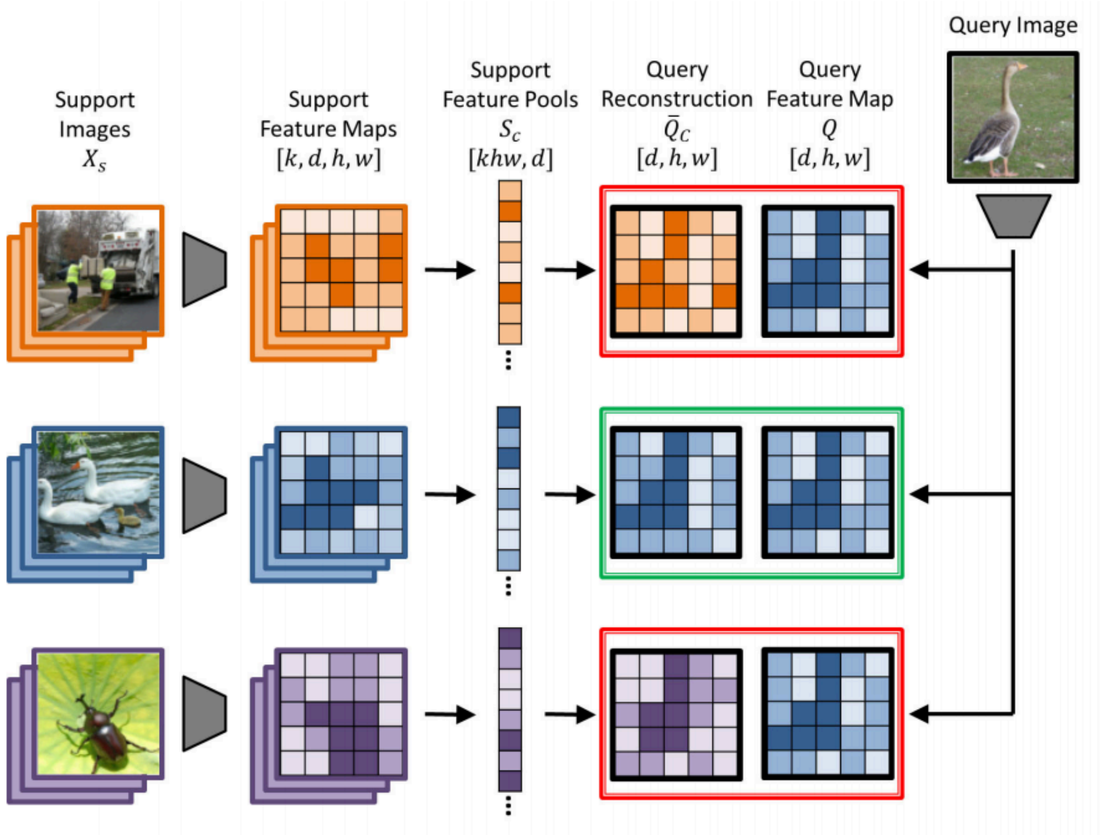
(c) ϕ $\quad \phi(I,m) = \dfrac{D_{\text{inter}}(I,m)}{D_{\text{intra}}(I,m)}$

negative margin (hyperparameter)

$$L = -\frac{1}{N}\sum_{i=1}^{N}\log\frac{e^{\beta\cdot\left(\mathbf{s}\left(z_i,W_{y_i}\right)-m\right)}}{e^{\beta\cdot\left(\mathbf{s}\left(z_i,W_{y_i}\right)-m\right)}+\sum_{j=1,j\neq y_i}^{C}e^{\beta\cdot\mathbf{s}\left(z_i,W_j\right)}}$$

Negative Margin Matters: Understanding Margin in Few-shot Classification. https://arxiv.org/abs/2003.12060. ECCV 2020.

Metric-based: FRN



Reconstruct feature maps to preserve location details

Find a matrix $W \in \mathbb{R}^{r \times kr}$ such that $WS_c \approx Q$

Closed-form solution:

$$\bar{W} = \arg\min_W ||Q - WS_c||^2 + \lambda||W||^2$$

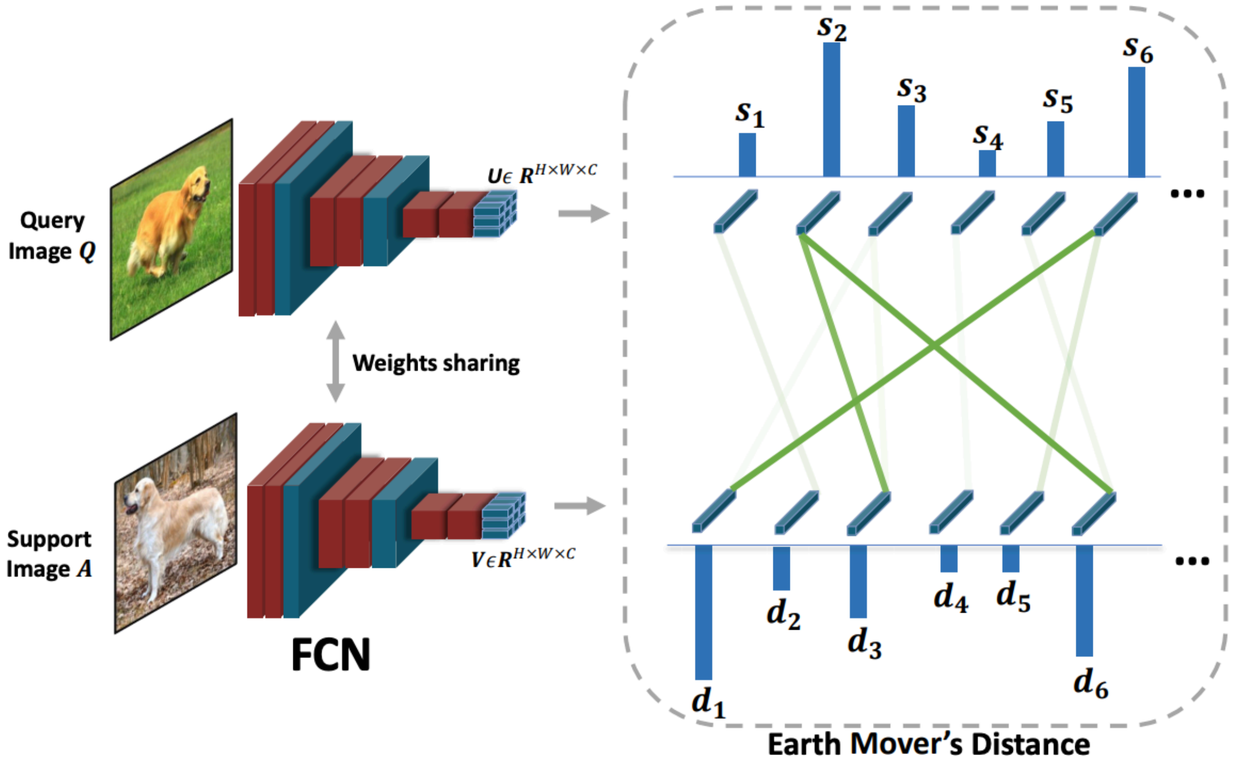
$$\bar{W} = QS_c^T(S_cS_c^T + \lambda I)^{-1}$$

$$\bar{Q}_c = \bar{W}S_c$$

$$P(y_q = c|x_q) = \frac{e^{(-\gamma\langle Q, \bar{Q}_c\rangle)}}{\sum_{c' \in C} e^{(-\gamma\langle Q, \bar{Q}_{c'}\rangle)}}$$

Few-Shot Classification with Feature Map Reconstruction Networks. https://arxiv.org/abs/2012.01506. CVPR 2021.

Metric-based: DeepEMD



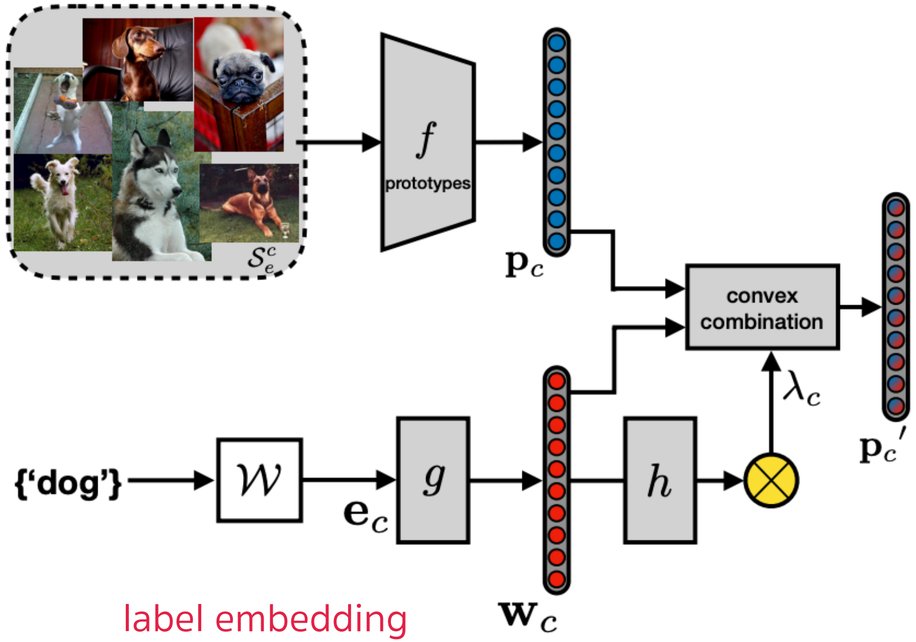
$$c_{ij} = 1 - \frac{\mathbf{u}_i{}^T \mathbf{v}_j}{\|\mathbf{u}_i\| \|\mathbf{v}_j\|}$$

$$distance(\boldsymbol{Q}, \boldsymbol{A}) = \sum_{ij} \tilde{x}_{ij}(1 - c_{ij})$$

$\boldsymbol{U} \in \boldsymbol{R}^{H \times W \times C}$: feature map of image Q
$\boldsymbol{V} \in \boldsymbol{R}^{H \times W \times C}$: feature map of image A
$\tilde{\boldsymbol{X}} \in \boldsymbol{R}^{HW \times HW}$: values of optimal matching flows
c_{ij}: matching cost between vector $\boldsymbol{u_i}$ and $\boldsymbol{v_j}$
s_i: weight of vector $\boldsymbol{u_i}$
d_j: weight of vector $\boldsymbol{v_j}$

DeepEMD: Differentiable Earth Mover's Distance for Few-Shot Learning. https://arxiv.org/abs/2003.06777. CVPR 2020.

Metric-based: AM3



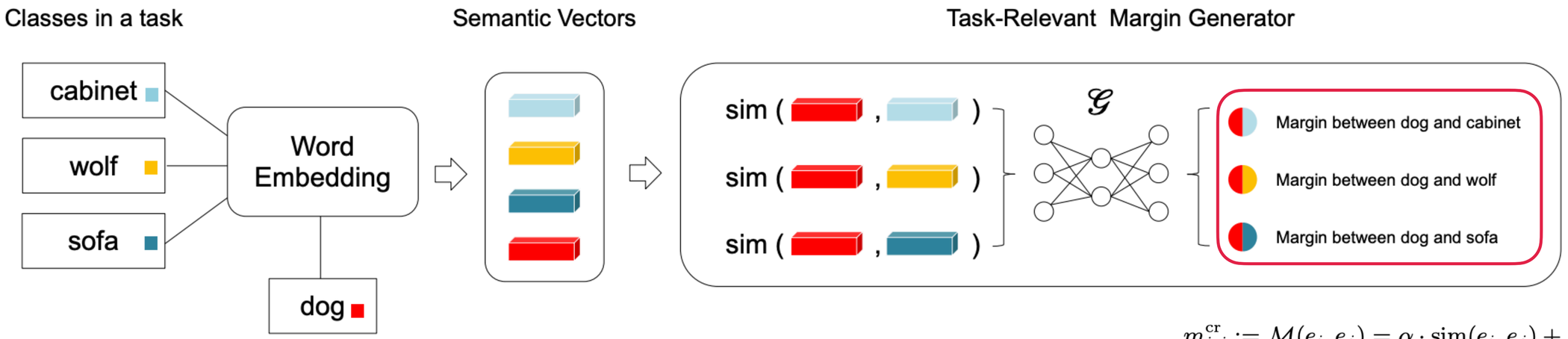
$$\mathbf{p}_c' = \lambda_c \cdot \mathbf{p}_c + (1 - \lambda_c) \cdot \mathbf{w}_c$$

$$\lambda_c = \frac{1}{1 + \exp(-h(\mathbf{w}_c))}$$

$$p_\theta(y = c | q_t, S_e, \mathcal{W}) = \frac{\exp(-d(f(q_t), \mathbf{p}_c'))}{\sum_k \exp(-d(f(q_t), \mathbf{p}_k'))}$$

Adaptive Cross-Modal Few-Shot Learning. https://arxiv.org/abs/1902.07104. NIPS 2019.

Metric-based: TRAML

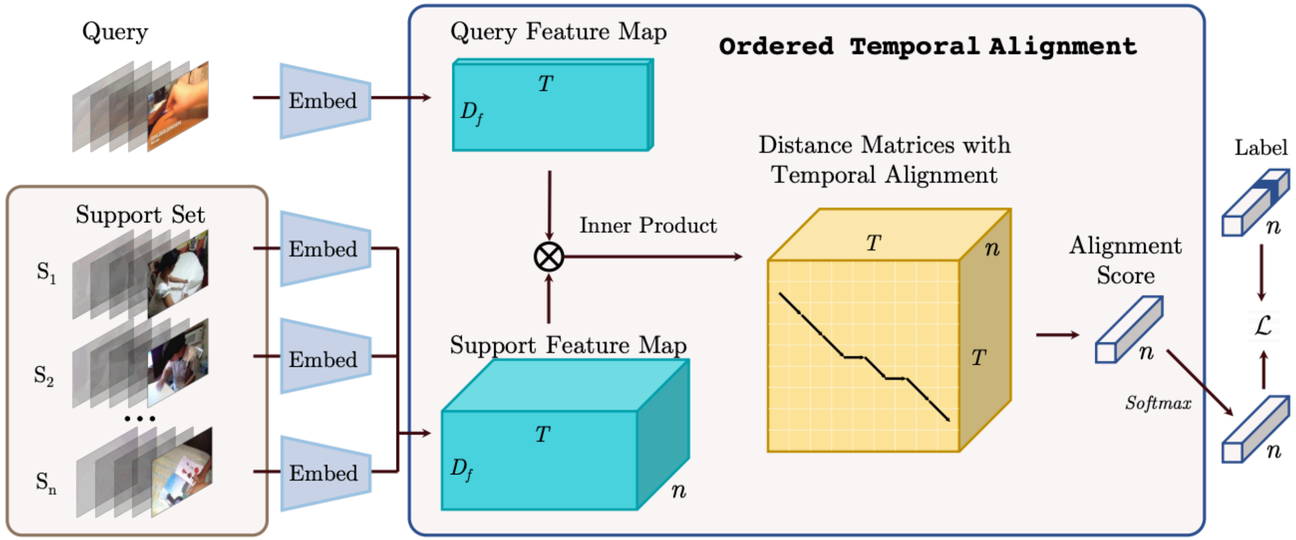


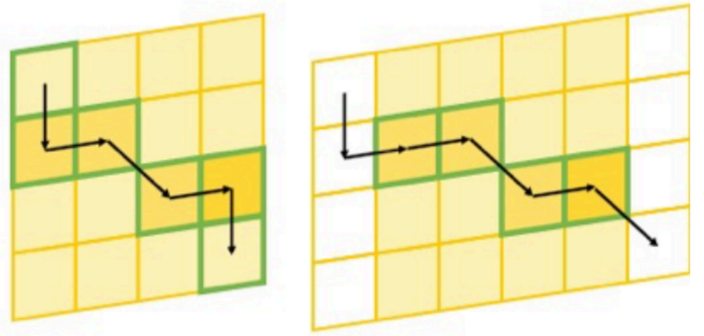
$$m_{i,j}^{\mathrm{cr}} := \mathcal{M}(e_i, e_j) = \alpha \cdot \mathrm{sim}(e_i, e_j) + \beta$$

$$p^{\mathrm{tr}}(y|x, S) = \frac{e^{\mathcal{D}(\mathcal{F}(x), r_y)}}{e^{\mathcal{D}(\mathcal{F}(x), r_y)} + \sum_{k \in C_t \setminus \{y\}} e^{\mathcal{D}(\mathcal{F}(x), r_k) + m_{y,k}^{\mathrm{tr}}}}$$

Boosting Few-Shot Learning With Adaptive Margin Loss. https://arxiv.org/abs/2005.13826. CVPR 2020.

Metric-based: OTAM (Video Classification)





$$\gamma(i,j) = D_{ij} + \begin{cases} \min\{\gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1)\}, \\ \qquad\qquad j = 0 \text{ or } j = T+1 \\ \min\{\gamma(i-1,j-1), \gamma(i,j-1), \text{ otherwise} \end{cases}$$

Few-Shot Video Classification via Temporal Alignment. https://arxiv.org/abs/1906.11415. CVPR 2020.

Thanks!