Adaptive Cross-Modal Few-shot Learning

Element AI, Montreal, Canada

NIPS 2019

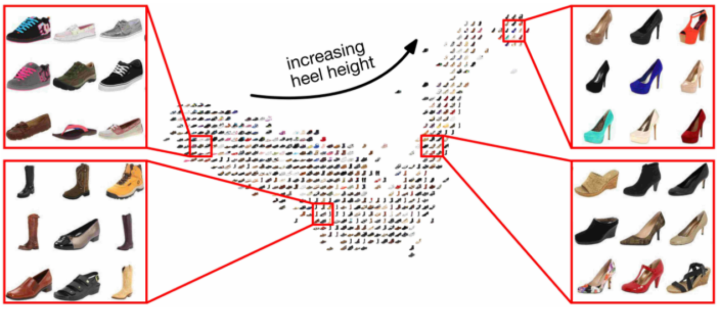
1. Introduction

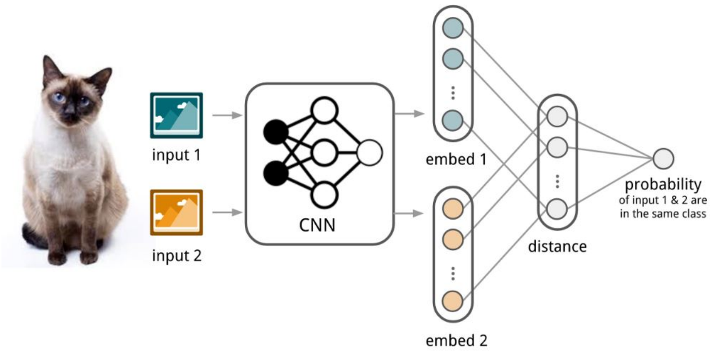
- **Deep learning requires a large amount of labeled data ,but impractical or expensive to acquire.**
- Limited labeled data lead to over-fitting and generalization issues
- Existing evidence: humans can learn new concepts from a very few samples
- **Few Shot Learning (FSL): learning new concepts with small number of labeled data points**
- Propose Adaptive Modality Mixture Mechanism (AM3), an approach that adaptively and selectively **combines information from two modalities, visual and semantic**, for few-shot learning

2. Related work

Meta Learning & Few-shot Learning

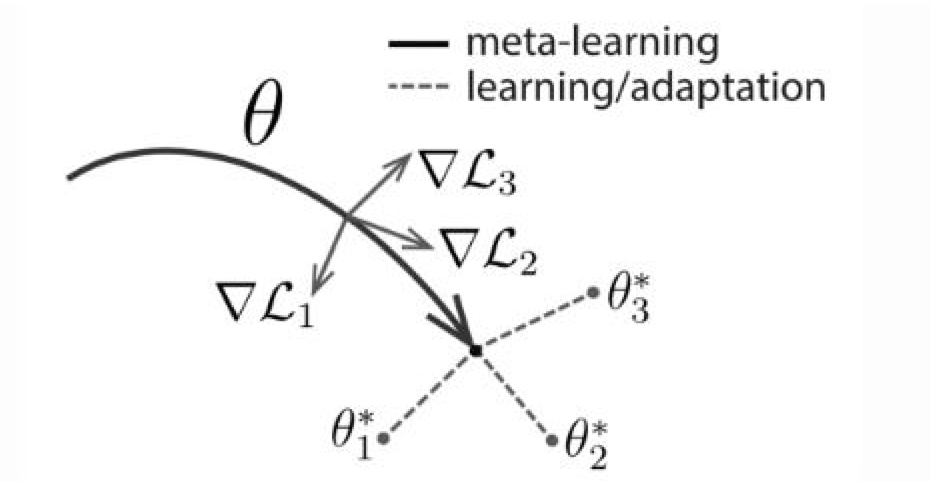
1. metric-based approaches: aim at learning representations that minimize intra-class distances while maximizing the distance between different classes





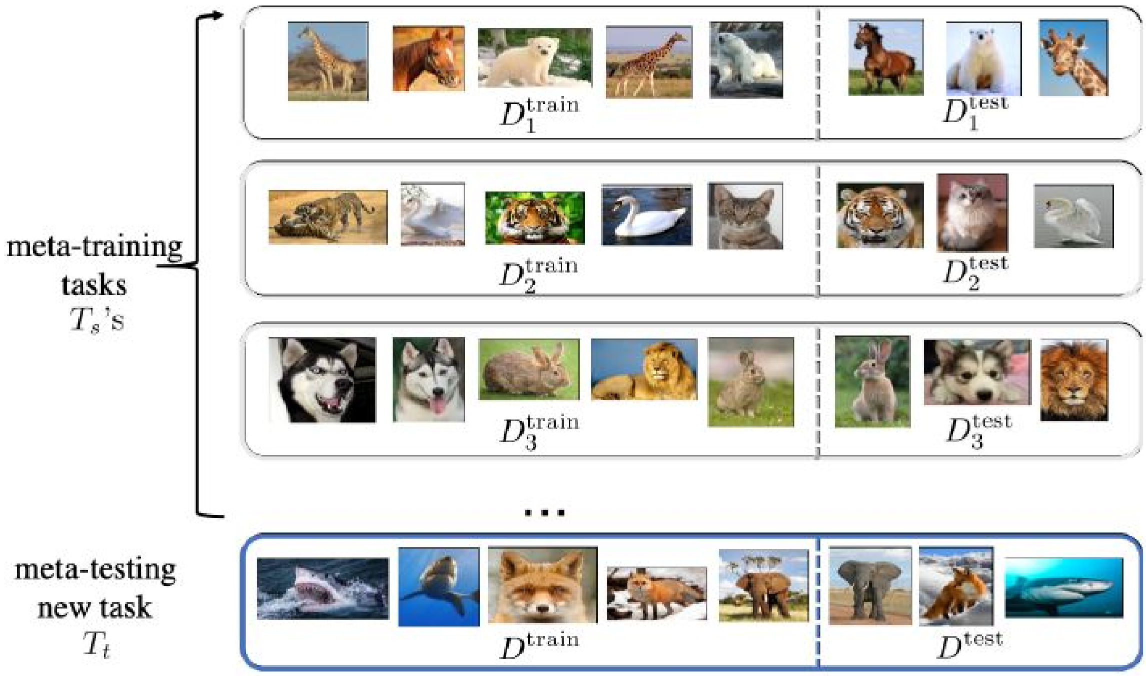
2. gradient-based approaches : aim at training models that can generalize well to new tasks with only a few fine-tuning updates

- Model-agnostic meta-learning for fast adaptation of deep networks, ICML 2017



3. Method

Episodic Training



Data set:

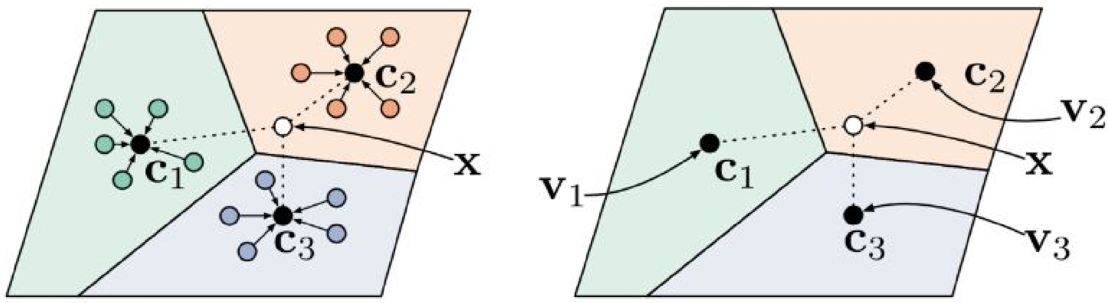
$$D_{=}\{D_{train}, D_{test}\}$$

Meta-learning : meta-training task Ts 와 meta-testing task Tt로 분리되고, 여러 개의 episode들이 합쳐서 task를 이룹니다. 하나의 episode에는 전체 데이터셋 D로부터 random으로 N개의 클래스를 선택하고 각 클래스에서 K 개의 이미지를 사용하여 총 N * K 개의 이미지가 포함됩니다. 이 데이터를 support set이라고 하며, 위 그림에서 D_train으로 되어 있는 episode를 보면 5개의 클래스가 있고 각 이미지는 1장씩 포함되어 있습니다. 이때 5-way 1-shot이라고 하며, 일반화하여 N-way K-shot이라고 합니다. Support set과의 중복없이 각 N개 클래스에서 Q개의 이미지를 더 추출하게 되는데 이를 query set이라고 합니다. Support set이 주어졌을 때, 모델에 query set을 이용하여 loss를 계산하고 이를 update하는 방식으로 학습이 진행됩니다.

meta-testing task에 속하는 data들의 클래스는 training task에 사용된 클래스들과 서로 중복되지 않습니다. meta-testing task에서는 meta-training 과정에서 한번도 학습이 되지 않은 새로운 데이터(unseen data)로 task를 testing해서 그 learner가 데이터들을 같은 클래스끼리 잘 분류를 하는지 확인합니다. 즉, Few shot learning 환경에서는 labeling 데이터가 많지 않기 때문에 Imagenet과 같은 대량의 label이 있는 데이터로 training을 한 뒤 meta-testing 단계에서 few shot data를 가지고 learner가 학습이 잘 되었는지 확인할 수 있게 하는 원리입니다.

정리해보자면, 기존의 classification model이 주어진 데이터셋으로 학습을 하고 새로운 이미지가 어떤 클래스인지 잘 예측하는 것이라면, meta-learning의 목적은 task 자체에 목적을 두고 training을 잘 해서 새로운 task가 있을 때, 그 task가 기존 training 한 과정 처럼 그 task를 잘 수행할 수 있도록 learn-to-learn 하는 것이라고 생각할 수 있습니다(**task를 잘 학습할 수 있도록 학습하는 것**).

Prototypical Network



(a) Few-shot (b) Zero-shot

Prototypical Network는 Metric 기반으로 학습되며, 기존에 우리가 알고 있는 nearest neighbor 알고리즘과 유사하다고 생각할 수 있습니다. Figure (a)에서 c1, c2, c3는 각 class에 대한 prototype을 나타내고 있습니다. 즉, 그 class를 대표 할 수 있는 값입니다. 이 그림에서 보이는 데이터들이 하나의 episode를 구성한다고 가정한다면 5 shot 3 way로 생각할 수 있겠습니다. 즉, c1에 대한 5개의 이미지를 이용하여 feature vector를 구하고 이를 평균내서 대표 값 한가지를 만드는 것입니다.

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\phi}(\mathbf{x}_i)$$

Test 단계에서는 입력 값 x의 feature vector와 위 3개의 class prototype과 distance를 계산하여 어느 class에 속하는지 분류합니다.

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

Loss function :

$$J(\phi) = -\log p_{\phi}(y = k \mid \mathbf{x})$$

Algorithm

Algorithm 1 Training episode loss computation for prototypical networks. N is the number of examples in the training set, K is the number of classes in the training set, $N_C \leq K$ is the number of classes per episode, N_S is the number of support examples per class, N_Q is the number of query examples per class. RANDOMSAMPLE(S, N) denotes a set of N elements chosen uniformly at random from set S, without replacement.

Input: Training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where each $y_i \in \{1, \ldots, K\}$. $\mathcal{D}_k$ denotes the subset of $\mathcal{D}$ containing all elements $(\mathbf{x}_i, y_i)$ such that $y_i = k$.

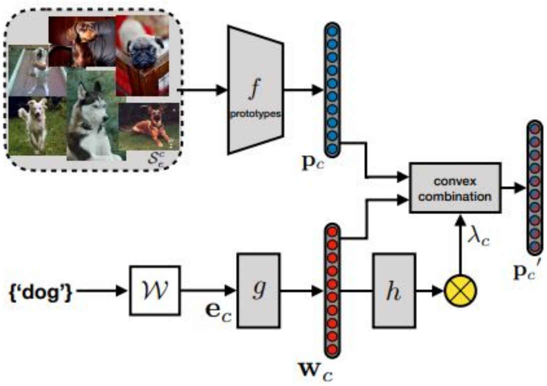
Output: The loss J for a randomly generated training episode.

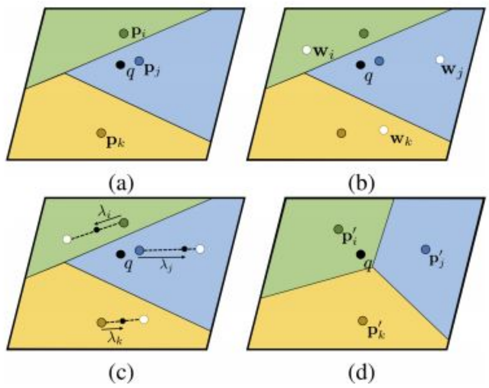
```
V ← RANDOMSAMPLE({1, ..., K}, N_C)                    ▷ Select class indices for episode
for k in {1, ..., N_C} do
    S_k ← RANDOMSAMPLE(D_{V_k}, N_S)                   ▷ Select support examples
    Q_k ← RANDOMSAMPLE(D_{V_k} \ S_k, N_Q)             ▷ Select query examples
    c_k ← (1/N_C) Σ_{(x_i, y_i) ∈ S_k} f_φ(x_i)        ▷ Compute prototype from support examples
end for
J ← 0                                                  ▷ Initialize loss
for k in {1, ..., N_C} do
    for (x, y) in Q_k do
        J ← J + (1/(N_C N_Q)) [d(f_φ(x), c_k)) + log Σ_{k'} exp(−d(f_φ(x), c_k))]   ▷ Update loss
    end for
end for
```

Adaptive Modality Mixture Mechanism (AM3)

- Augment metric-based FSL methods to incorporate language structure

Figure 1: Concepts have different visual and semantic feature space. *(Left)* Some categories may have similar visual features and dissimilar semantic features. (*Right*) Other can possess same semantic label but very distinct visual features. Our method adaptively exploits both modalities to improve classification performance in low-shot regime.





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

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

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

A Algorithm for Episode Loss

Algorithm 1: Training episode loss computation for adaptive cross-modality few-shot learning. M is the total number of classes in the training set, N is the number of classes in every episode, K is the number of supports for each class, K_Q is the number of queries for each class, $\mathcal{W}$ is the pretrained label embedding dictionary.

Input: Training set $\mathcal{D}_{\text{train}} = \{(\mathbf{x}_i, y_i)\}_i, y_i \in \{1, ..., M\}$. $\mathcal{D}^c_{\text{train}} = \{(\mathbf{x}_i, y_i) \in \mathcal{D}_{\text{train}} \mid y_i = c\}$.
Output: Episodic loss $\mathcal{L}(\theta)$ for sampled episode e.

```
{Select N classes for episode e}
C ← RandomSample({1, ..., M}, N)
{Compute cross-modal prototypes}
for c in C do
  S_e^c ← RandomSample(D_train^c, K)
  Q_e^c ← RandomSample(D_train^c \ S_e^c, K_Q)
  p_c ← (1/|S_e^c|) Σ_{(s_i,y_i)∈S_e^c} f(s_i)
  e_c ← LookUp(c, W)
  w_c ← g(e_c)
  λ_c ← 1 / (1+exp(−h(w_c)))
  p'_c ← λ_c · p_c + (1 − λ_c) · w_c
end for
{Compute loss}
L(θ) ← 0
for c in C do
  for (q_t, y_t) in Q_e^c do
    L(θ) ← L(θ) + (1/(N·K)) [d(f(q_t), p'_c)) + log Σ_k exp(−d(f(q_t), p'_k))]
  end for
end for
```

$$\{\texttt{Select } N \texttt{ classes for episode } e\}$$
$$C \leftarrow RandomSample(\{1, ..., M\}, N)$$
$$\{\texttt{Compute cross-modal prototypes}\}$$
for c in C **do**
$$\mathcal{S}^c_e \leftarrow RandomSample(\mathcal{D}^c_{\text{train}}, K)$$
$$\mathcal{Q}^c_e \leftarrow RandomSample(\mathcal{D}^c_{\text{train}} \setminus \mathcal{S}^c_e, K_Q)$$
$$\mathbf{p}_c \leftarrow \frac{1}{|\mathcal{S}^c_e|} \sum_{(s_i, y_i) \in \mathcal{S}^c_e} f(s_i)$$
$$\mathbf{e}_c \leftarrow LookUp(c, \mathcal{W})$$
$$\mathbf{w}_c \leftarrow g(\mathbf{e}_c)$$
$$\lambda_c \leftarrow \frac{1}{1+\exp(-h(\mathbf{w}_c))}$$
$$\mathbf{p}'_c \leftarrow \lambda_c \cdot \mathbf{p}_c + (1 - \lambda_c) \cdot \mathbf{w}_c$$
end for
$$\{\texttt{Compute loss}\}$$
$$\mathcal{L}(\theta) \leftarrow 0$$
for c in C **do**
for (q_t, y_t) in Q^c_e **do**
$$\mathcal{L}(\theta) \leftarrow \mathcal{L}(\theta) + \frac{1}{N \cdot K}[d(f(q_t), \mathbf{p}'_c)) \ + \ \log \textstyle\sum_k \exp(-d(f(q_t), \mathbf{p}'_k))]$$
end for
end for

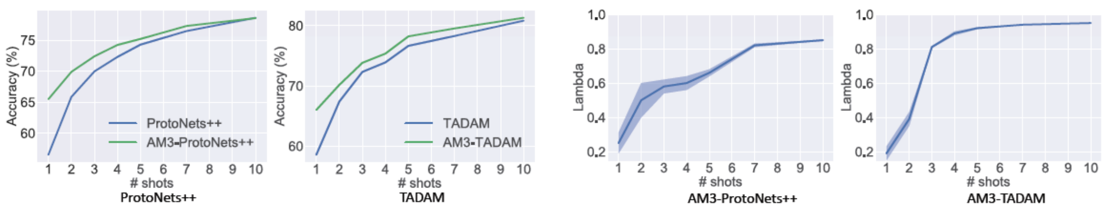
4. Experiments & Results

Model	**Test Accuracy**		
	5-way 1-shot	5-way 5-shot	5-way 10-shot
Uni-modality few-shot learning baselines			
Matching Network [53]	$43.56 \pm 0.84\%$	$55.31 \pm 0.73\%$	-
Prototypical Network [47]	$49.42 \pm 0.78\%$	$68.20 \pm 0.66\%$	$74.30 \pm 0.52\%$
Discriminative k-shot [2]	$56.30 \pm 0.40\%$	$73.90 \pm 0.30\%$	$78.50 \pm 0.00\%$
Meta-Learner LSTM [38]	$43.44 \pm 0.77\%$	$60.60 \pm 0.71\%$	-
MAML [7]	$48.70 \pm 1.84\%$	$63.11 \pm 0.92\%$	-
ProtoNets w Soft k-Means [39]	$50.41 \pm 0.31\%$	$69.88 \pm 0.20\%$	-
SNAIL [32]	$55.71 \pm 0.99\%$	$68.80 \pm 0.92\%$	-
CAML [16]	$59.23 \pm 0.99\%$	$72.35 \pm 0.71\%$	-
LEO [41]	$61.76 \pm 0.08\%$	$77.59 \pm 0.12\%$	-
Modality alignment baselines			
DeViSE [9]	$37.43 \pm 0.42\%$	$59.82 \pm 0.39\%$	$66.50 \pm 0.28\%$
ReViSE [14]	$43.20 \pm 0.87\%$	$66.53 \pm 0.68\%$	$72.60 \pm 0.66\%$
CBPL [29]	$58.50 \pm 0.82\%$	$75.62 \pm 0.61\%$	-
f-CLSWGAN [57]	$53.29 \pm 0.82\%$	$72.58 \pm 0.27\%$	$73.49 \pm 0.29\%$
CADA-VAE [44]	$58.92 \pm 1.36\%$	$73.46 \pm 1.08\%$	$76.83 \pm 0.98\%$
Modality alignment baselines extended to metric-based FSL framework			
DeViSE-FSL	$56.99 \pm 1.33\%$	$72.63 \pm 0.72\%$	$76.70 \pm 0.53\%$
ReViSE-FSL	$57.23 \pm 0.76\%$	$73.85 \pm 0.63\%$	$77.21 \pm 0.31\%$
f-CLSWGAN-FSL	$58.47 \pm 0.71\%$	$72.23 \pm 0.45\%$	$76.90 \pm 0.38\%$
CADA-VAE-FSL	$61.59 \pm 0.84\%$	$75.63 \pm 0.52\%$	$79.57 \pm 0.28\%$
AM3 and its backbones			
ProtoNets++	$56.52 \pm 0.45\%$	$74.28 \pm 0.20\%$	$78.31 \pm 0.44\%$
AM3-ProtoNets++	$65.21 \pm 0.30\%$	$75.20 \pm 0.27\%$	$78.52 \pm 0.28\%$
TADAM [35]	$58.56 \pm 0.39\%$	$76.65 \pm 0.38\%$	$80.83 \pm 0.37\%$
AM3-TADAM	$\mathbf{65.30 \pm 0.49\%}$	$\mathbf{78.10 \pm 0.36\%}$	$\mathbf{81.57 \pm 0.47\ \%}$

Table 1: Few-shot classification accuracy on *test* split of *mini*ImageNet. Results in the top use only visual features. Modality alignment baselines are shown on the middle and our results (and their backbones) on the bottom part.

Model	Test Accuracy	
	5-way 1-shot	5-way 5-shot
Uni-modality few-shot learning baselines		
MAML† [7]	51.67 ± 1.81%	70.30 ± 0.08%
Proto. Nets with Soft k-Means [39]	53.31 ± 0.89%	72.69 ± 0.74%
Relation Net† [50]	54.48 ± 0.93%	71.32 ± 0.78%
Transductive Prop. Nets [28]	54.48 ± 0.93%	71.32 ± 0.78%
LEO [41]	66.33 ± 0.05%	81.44 ± 0.09%
Modality alignment baselines		
DeViSE [9]	49.05±0.92%	68.27±0.73%
ReViSE [14]	52.40±0.46%	69.92±0.59%
CADA-VAE [44]	58.92±1.36%	73.46±1.08%
Modality alignment baselines extended to metric-based FSL framework		
DeViSE-FSL	61.78 ± 0.43%	77.17 ± 0.81%
ReViSE-FSL	62.77 ± 0.31%	77.27 ± 0.42%
CADA-VAE-FSL	63.16 ± 0.93%	78.86 ± 0.31%
AM3 and its backbones		
ProtoNets++	58.47 ± 0.64%	78.41 ± 0.41%
AM3-ProtoNets++	67.23 ± 0.34%	78.95 ± 0.22%
TADAM [35]	62.13 ± 0.31%	81.92 ± 0.30%
AM3-TADAM	**69.08 ± 0.47%**	**82.58 ± 0.31%**

Table 2: Few-shot classification accuracy on *test* split of *tiered*ImageNet. Results in the top use only visual features. Modality alignment baselines are shown in the middle and our results (and their backbones) in the bottom part. †deeper net, evaluated in [28].



(a) Accuracy vs. # shots (b) λ vs. # shots

Figure 3: (a) Comparison of AM3 and its corresponding backbone for different number of shots (b) Average value of λ (over whole validation set) for different number of shot, considering both backbones.