Meta-Learning for semi-supervised few-shot classification

2018.10.11

김보섭

Agenda

1. Introduction
2. Background
   - Few-shot learning
   - Prototypical networks
3. Semi-supervised few-shot learning
   - Semi-supervised prototypical networks
4. Experiments
5. Conclusion

Introduction

근래의 Deep learning 방법론들은 labeled data가 부족한 상황에서는 일반화 성능이 떨어지는 등 문제점이 많음, 그러나 사람은...

- 적은 양의 labeled data에 대해서도 특징을 잘 파악하고, 파악한 특징을 바탕으로 일반화를 잘 한다.

  → Few shot learning!!

- 본 논문은 Few-shot learning과 관련 있는 문제 중, class마다 labeled data가 적을 경우에도 classification을 잘하는 model을 만들고자함

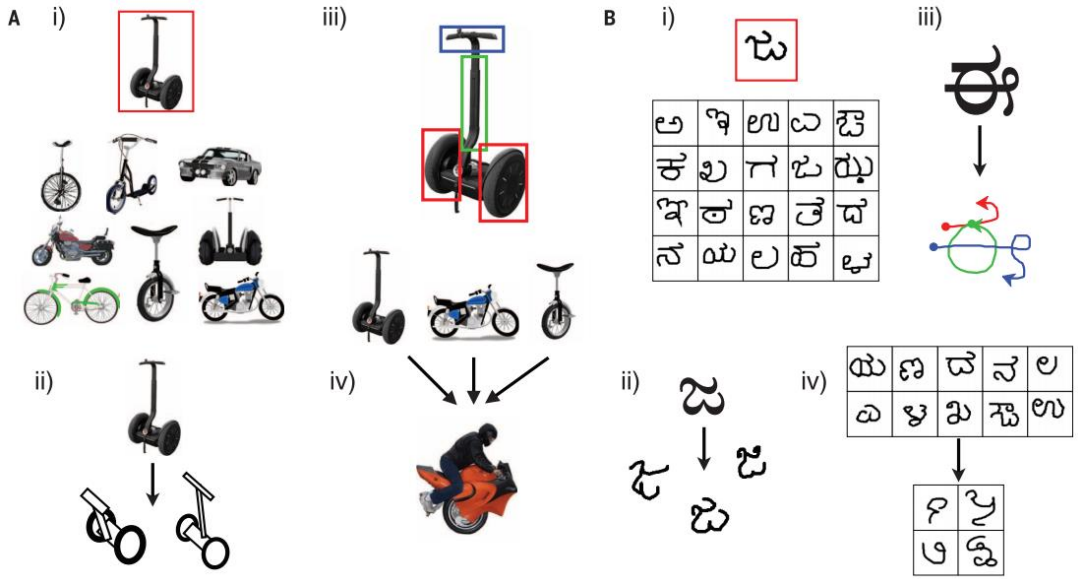


Fig. 1. People can learn rich concepts from limited data. (**A** and **B**) A single example of a new concept (red boxes) can be enough information to support the (i) classification of new examples, (ii) generation of new examples, (iii) parsing an object into parts and relations (parts segmented by color), and (iv) generation of new concepts from related concepts. [Image credit for (A), iv, bottom: With permission from Glenn Roberts and Motorcycle Mojo Magazine]

Lake, Brenden M., Ruslan Salakhutdinov, and Joshua B. Tenenbaum. "Human-level concept learning through probabilistic program induction." Science 350.6266 (2015): 1332-1338.

Introduction

기존의 관련 연구와는 다르게 본 논문에서는 Semi-supervised concept을 Few-shot learning에 차용함, 그 이유는...

- 사람은 unlabeled data에 대해서 공통적인 특징을 찾아내고 활용할 수 있음
  - ✓ Semi-supervised learning을 도입하여 비슷하게나마 활용하고자함
- 아래의 두 가지 상황을 고려
  - ✓ Scenario 1 : unlabeled data에 학습해야할 class가 존재하는 경우
  - ✓ Scenario 2 : unlabeled data에 학습해야할 class가 존재하는 것은 아니지만 학습하고자하는 class를 학습할 때 방해가 되는경우

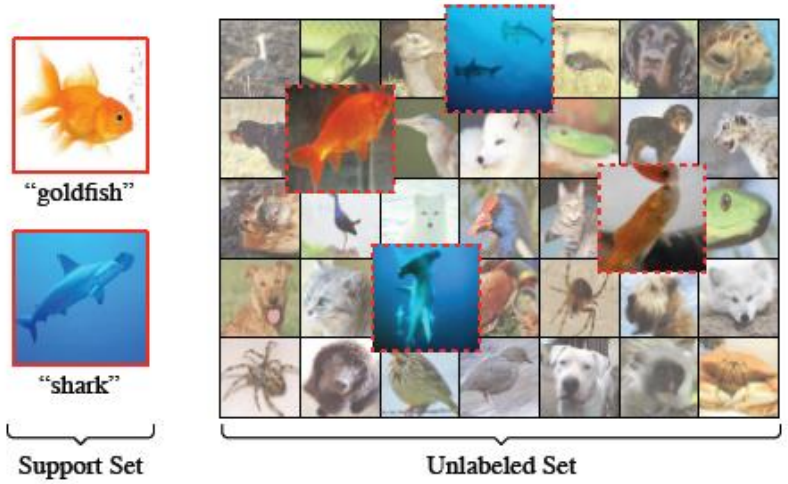


Figure 1: Consider a setup where the aim is to learn a classifier to distinguish between two previously unseen classes, goldfish and shark, given not only labeled examples of these two classes, but also a larger pool of unlabeled examples, some of which may belong to one of these two classes of interest. In this work we aim to move a step closer to this more natural learning framework by incorporating in our learning episodes unlabeled data from the classes we aim to learn representations for (shown with dashed red borders) as well as from *distractor* classes .

Background

Few-shot learning

Episodic paradigm 기반의 Few-shot learning은 대량의 labeled training data를 이용하여, Few-shot classification 상황을 모사하는 것임

- K-shot, N-Way episode training of Few-shot learning
  - ✓ Sampling a small subset of N classes from C_{train}, and sampling K examples from each of the N classes to generate support set S

$$S = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \dots, (\boldsymbol{x}_{N\times K}, y_{N\times K})\}$$

  - ✓ Sampling T examples from each of the N classes excluding sampled examples of support set S to generate Query set Q

$$Q = \{(\boldsymbol{x}_1^*, y_1^*), (\boldsymbol{x}_2^*, y_2^*), \dots, (\boldsymbol{x}_{N\times T}^*, y_{N\times T}^*)\}$$

  - ✓ Training on such episodes is done by feeding the support set S to the model and updating its parameters to minimize the loss of its predictions for the examples in the query set Q

Background

Prototypical Network (1/2)

Episode의 support set S로 각 class 별 prototype vector를 만들고, query set Q의 input과 각 class 별 prototype vector와의 거리를 기반으로 Classification

- The prototype $\boldsymbol{p}_c$ of each class c is computed by Prototypical Network h

$$\boldsymbol{p}_c = \frac{\sum_i h(\boldsymbol{x_i}) z_{i,c}}{\sum_i z_{i,c}}, where\ z_{i,c} = \mathrm{I}[y_i = c]$$

- The predictor for the class of any new example $\boldsymbol{x}^*$

$$p(c|\boldsymbol{x}^*, \{\boldsymbol{p}_c\}) = \frac{\exp(-\|h(\boldsymbol{x}^*) - \boldsymbol{p}_c\|_2^2)}{\sum_{c'} \exp(-\|h(\boldsymbol{x}^*) - \boldsymbol{p}_{c'}\|_2^2)}$$

- The loss function used to update Prototypical Networks for a given training episodes

$$-\frac{1}{NT}\sum_i \log p(y_i^*|\boldsymbol{x}_i^*, \{\boldsymbol{p}_c\})$$

Background

Prototypical Network (2/2)

Episode의 support set S로 각 class 별 prototype vector를 만들고, query set Q의 input과 각 class 별 prototype vector와의 거리를 기반으로 Classification

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.

$V \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for episode
for k in $\{1, \ldots, N_C\}$ **do**
  $S_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples
  $Q_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples
  $\mathbf{c}_k \leftarrow \frac{1}{N_C} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\phi}(\mathbf{x}_i)$ ▷ Compute prototype from support examples
end for
$J \leftarrow 0$ ▷ Initialize loss
for k in $\{1, \ldots, N_C\}$ **do**
  for $(\mathbf{x}, y)$ in Q_k **do**
    $J \leftarrow J + \frac{1}{N_C N_Q} \left[d(f_{\phi}(\mathbf{x}), \mathbf{c}_k)) + \log \sum_{k'} \exp(-d(f_{\phi}(\mathbf{x}), \mathbf{c}_k)) \right]$ ▷ Update loss
  end for
end for

Snell, Jake, Kevin Swersky, and Richard Zemel. "Prototypical networks for few-shot learning." Advances in Neural Information Processing Systems. 2017.

Semi-supervised few-shot learning

Semi-supervised setting for few-shot learning

support set S에 unlabeled set을 추가, query set Q의 구성은 동일

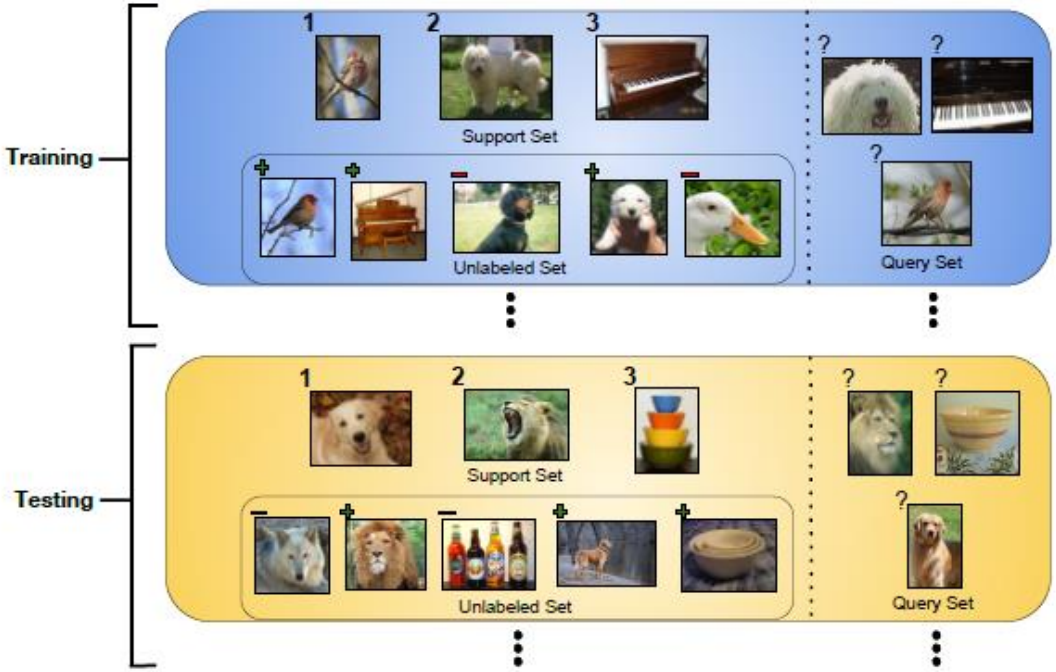
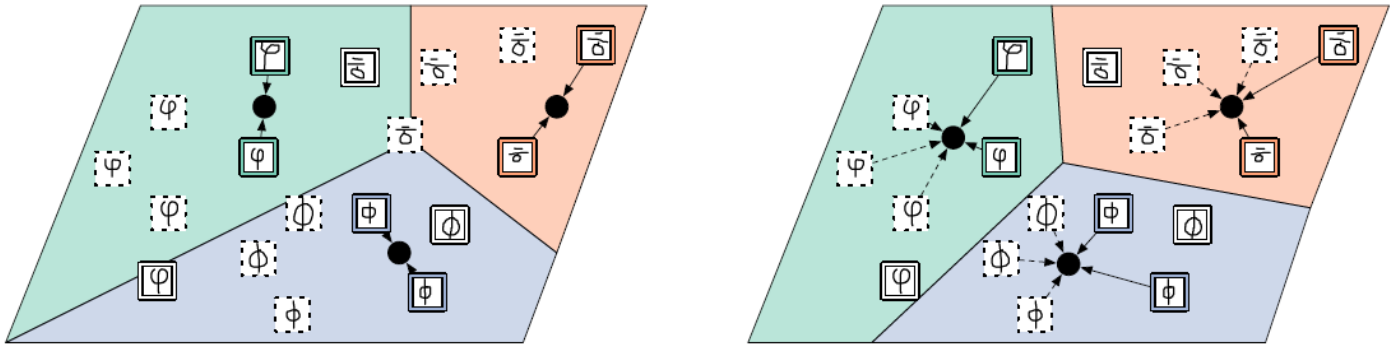


Figure 2: Example of the semi-supervised few-shot learning setup. Training involves iterating through training episodes, consisting of a support set $\mathcal{S}$, an unlabeled set $\mathcal{R}$, and a query set $\mathcal{Q}$. The goal is to use the labeled items (shown with their numeric class label) in $\mathcal{S}$ and the unlabeled items in $\mathcal{R}$ within each episode to generalize to good performance on the corresponding query set. The unlabeled items in $\mathcal{R}$ may either be pertinent to the classes we are considering (shown above with green plus signs) or they may be *distractor* items which belong to a class that is not relevant to the current episode (shown with red minus signs). However note that the model does not actually have ground truth information as to whether each unlabeled example is a distractor or not; the plus/minus signs are shown only for illustrative purposes. At test time, we are given new episodes consisting of novel classes not seen during training that we use to evaluate the meta-learning method.

Semi-supervised few-shot learning

Semi-supervised prototypical networks

unlabeled set을 이용, 기존의 prototype $\boldsymbol{p}_c$를 개선하는 아래의 세 가지 방법론을 제안, 개선된 prototype $\tilde{\boldsymbol{p}}_c$를 이용하는 것 외에는 prototypical network와 동일



Before Refinement

After Refinement

Figure 3: Left: The prototypes are initialized based on the mean location of the examples of the corresponding class, as in ordinary Prototypical Networks. Support, unlabeled, and query examples have solid, dashed, and white colored borders respectively. Right: The refined prototypes obtained by incorporating the unlabeled examples, which classifies all query examples correctly.

Semi-supervised few-shot learning

Prototypical networks with soft k-means

prototype $\boldsymbol{p}_c$을 cluster의 centroid로 간주하고, soft k-means clustering 기반으로 unlabeled set을 cluster에 할당 후 새로운 centroid $\tilde{\boldsymbol{p}}_c$를 생성

- unlabeled set의 class set은 support set, query set의 class set과 동일
- soft k-means의 방식의 assignment step은 한번만!

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.

$V \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for episode
for k in $\{1, \ldots, N_C\}$ **do**
  $S_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples
  $Q_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples
  $\mathbf{c}_k \leftarrow \frac{1}{N_C} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$ ▷ Compute prototype from support examples
end for
$J \leftarrow 0$ ▷ Initialize loss
for k in $\{1, \ldots, N_C\}$ **do**
  for $(\mathbf{x}, y)$ in Q_k **do**
    $J \leftarrow J + \frac{1}{N_C N_Q} \left[d(f_\phi(\mathbf{x}), \mathbf{c}_k)) + \log \sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_k)) \right]$ ▷ Update loss
  end for
end for

$$\tilde{\boldsymbol{p}}_c = \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c} + \sum_j h(\tilde{\boldsymbol{x}}_j) \tilde{z}_{j,c}}{\sum_i z_{i,c} + \sum_j \tilde{z}_{j,c}}$$

$$z_{i,c} = \mathrm{I}[y_i = c]$$

$$\tilde{z}_{j,c} = \frac{\exp\left(-\left\|h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_c\right\|_2^2\right)}{\sum_{c'} \exp\left(-\left\|h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_{c'}\right\|_2^2\right)}$$

Semi-supervised few-shot learning

Prototypical networks with soft k-means with a distractor cluster

unlabeled set에 support set, query set에 없는 class들이 존재, 이를 하나의 distractor class로 가정하고 하나의 cluster로 간주

- unlabeled set에 support set, query set에 없는 class들도 존재, 이들은 하나의 class로 간주 (distractor class)
- soft k-means의 방식의 assignment step은 한번만!

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.

$V \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for episode
for k in $\{1, \ldots, N_C\}$ **do**
  $S_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples
  $Q_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples
  $\mathbf{c}_k \leftarrow \frac{1}{N_C} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$ ▷ Compute prototype from support examples
end for
$J \leftarrow 0$ ▷ Initialize loss
for k in $\{1, \ldots, N_C\}$ **do**
  for $(\mathbf{x}, y)$ in Q_k **do**
    $J \leftarrow J + \frac{1}{N_C N_Q} \left[d(f_\phi(\mathbf{x}), \mathbf{c}_k)) + \log \sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_k)) \right]$ ▷ Update loss
  end for
end for

$$\boldsymbol{p}_c = \begin{cases} \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c}}{\sum_i z_{i,c}} & for\ c = 1, \ldots, N \\ \mathbf{0} & for\ c = N+1 \end{cases}$$

$$\tilde{\boldsymbol{p}}_c = \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c} + \sum_j h(\tilde{\boldsymbol{x}}_j) \tilde{z}_{j,c}}{\sum_i z_{i,c} + \sum_j \tilde{z}_{j,c}}$$

$$z_{i,c} = \mathrm{I}[y_i = c], \qquad \tilde{z}_{j,c} = \frac{\exp\left(-\frac{1}{r_c^2} \left\| h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_c \right\|_2^2 - A(r_c) \right)}{\sum_{c'} \exp(-\frac{1}{r_{c'}^2} \left\| h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_{c'} \right\|_2^2 - A(r_{c'}))}$$

$$A(r) = \frac{1}{2}\log(2\pi) + \log(r)$$

$$r_{1,\ldots N} = 1, r_{N+1}\ is\ parameter$$

Semi-supervised few-shot learning

Prototypical networks with soft k-means and masking

distractor class를 가정하는 대신 prototype을 update할 때, unlabeled example의 반영도를 계산하는 multi-layer perceptron를 prototypical network와 같이 학습

- Multi-layer perceptron은 prototype 별 cluster의 centroid update 시, unlabeled example을 반영하기위한 threshold β_c, γ_c를 계산
- Soft k-means의 방식의 assignment step은 한번만!

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.

$V \leftarrow$ RANDOMSAMPLE$(\{1, \ldots, K\}, N_C)$ ▷ Select class indices for episode
for k in $\{1, \ldots, N_C\}$ **do**
  $S_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k}, N_S)$ ▷ Select support examples
  $Q_k \leftarrow$ RANDOMSAMPLE$(\mathcal{D}_{V_k} \setminus S_k, N_Q)$ ▷ Select query examples
  $\mathbf{c}_k \leftarrow \frac{1}{N_C} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_\phi(\mathbf{x}_i)$ ▷ Compute prototype from support examples
end for
$J \leftarrow 0$ ▷ Initialize loss
for k in $\{1, \ldots, N_C\}$ **do**
  for $(\mathbf{x}, y)$ in Q_k **do**
    $J \leftarrow J + \frac{1}{N_C N_Q} \left[d(f_\phi(\mathbf{x}), \mathbf{c}_k)) + \log \sum_{k'} \exp(-d(f_\phi(\mathbf{x}), \mathbf{c}_k)) \right]$ ▷ Update loss
  end for
end for

$$\tilde{d}_{j,c} = \frac{d_{j,c}}{\frac{1}{M}\sum_j d_{j,c}}, \text{where } d_{j,c} = \left\| h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_c \right\|_2^2$$

$$[\beta_c, \gamma_c] = \text{MLP}\left(\left[\min_j \tilde{d}_{j,c}, \max_j \tilde{d}_{j,c}, \text{var}_j \tilde{d}_{j,c}, \text{skew}_j \tilde{d}_{j,c}, \text{kurt}_j \tilde{d}_{j,c} \right] \right)$$

$$\tilde{\boldsymbol{p}}_c = \frac{\sum_i h(\boldsymbol{x}_i) z_{i,c} + \sum_j h(\tilde{\boldsymbol{x}}_j) \tilde{z}_{j,c} m_{j,c}}{\sum_i z_{i,c} + \sum_j \tilde{z}_{j,c} m_{j,c}}$$

$$m_{j,c} = \sigma\left(-\gamma_c (\tilde{d}_{j,c} - \beta_c) \right)$$

$$z_{i,c} = \text{I}[y_i = c], \qquad \tilde{z}_{j,c} = \frac{\exp\left(-\left\| h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_c \right\|_2^2 \right)}{\sum_{c'} \exp(-\left\| h(\tilde{\boldsymbol{x}}_j) - \boldsymbol{p}_{c'} \right\|_2^2)}$$

Experiments

Datasets

제안하는 방법론을 세 가지 dataset을 이용하여 성능을 검증

- Omniglot
  - ✓ 50 종류의 alphabet 1,623개, 각 alphabet 별로 20명의 사람에 의하여 쓰여짐
  - ✓ 28×28의 image로 resize한 뒤, rotation을 실시하여 6,492개의 class를 가지게 만들고, 4,112개의 class에 해당하는 image를 training, 688개의 class에 해당하는 image를 validation, 1,692개의 class에 해당하는 image를 test에 활용
  - ✓ Vinyals, Oriol, et al. "Matching networks for one shot learning." *Advances in Neural Information Processing Systems*. 2016
- mini-ImageNet
  - ✓ ILSVRC-12 dataset에서 100개의 class를 random하게 뽑고, 각 class 별로 600개의 image를 sampling
  - ✓ 64개의 class에 해당하는 image를 training, 16개의 class에 해당하는 image를 validation, 20개에 해당하는 image를 test로 활용
  - ✓ Ravi, Sachin, and Hugo Larochelle. "Optimization as a model for few-shot learning." (2016)
- tiered-ImageNet
  - ✓ 본 논문에서 제안하는 dataset
  - ✓ mini-ImageNet과 유사하나 608개의 class를 활용하고, high-level category를 반영한 dataset

Experiments

Adapting the datasets for semi-supervised learning

각 dataset 별로 아래의 과정을 통해 episode를 구성하여, 제안하는 세 가지 방법론의 모형을 학습

labeled set, unlabeled set 구성

- Omniglot, tiered-ImageNet의 경우 각 class 별로 10%씩 image를 sampling하여 labeled set을 구성하고, 나머지 90%는 unlabeled set으로 활용
- mini-ImageNet의 경우는 각 class 별로 40%씩 image를 sampling하여 labeled set을 구성하고, 나머지 60%는 unlabeled set으로 활용

episode 구성

- support set, unlabeled set in episode
  - ✓ Training에 활용하는 class set C_{train}에서 N개의 class를 sampling
  - ✓ N개의 class에 해당하는 labeled set에서 각 class 별 K개의 image를 sampling, (NK개)
  - ✓ N개의 class에 해당하는 unlabeled set에서 각 class 별 M개의 image를 sampling, (MN개)
  - ✓ distractor class가 존재한다고 가정할 경우, C_{train}에서 이미 뽑힌 N개의 class를 제외한 class set에서 H개의 class를 sampling하고, H개의 class에 해당하는 unlabeled set에서 M개의 image를 sampling (MH개)
- query set in episode
  - ✓ class set C_{train}에서 sampling한 N개의 class에 해당하는 labeled set에서 sampling한 image로 구성

Experiments

Results (1/2)

Benchmark dataset에 대해서 제안한 방법론 중 적어도 한가지는 baseline인 Supervised, Semi-supervised inference보다 좋은 성능을 보임

Models	Acc.	Acc. w/ D
Supervised	94.62 ± 0.09	94.62 ± 0.09
Semi-Supervised Inference	97.45 ± 0.05	95.08 ± 0.09
Soft k-Means	97.25 ± 0.10	95.01 ± 0.09
Soft k-Means+Cluster	**97.68 ± 0.07**	97.17 ± 0.04
Masked Soft k-Means	97.52 ± 0.07	**97.30 ± 0.08**

Table 1: Omniglot 1-shot classification results. In this table as well as those below "w/ D" denotes "with distractors", where the unlabeled images contain irrelevant classes.

Models	1-shot Acc.	5-shot Acc.	1-shot Acc w/ D	5-shot Acc. w/ D
Supervised	43.61 ± 0.27	59.08 ± 0.22	43.61 ± 0.27	59.08 ± 0.22
Semi-Supervised Inference	48.98 ± 0.34	63.77 ± 0.20	47.42 ± 0.33	62.62 ± 0.24
Soft k-Means	**50.09 ± 0.45**	**64.59 ± 0.28**	**48.70 ± 0.32**	**63.55 ± 0.28**
Soft k-Means+Cluster	49.03 ± 0.24	63.08 ± 0.18	**48.86 ± 0.32**	61.27 ± 0.24
Masked Soft k-Means	**50.41 ± 0.31**	**64.39 ± 0.24**	**49.04 ± 0.31**	62.96 ± 0.14

Table 2: *mini*ImageNet 1/5-shot classification results.

Models	1-shot Acc.	5-shot Acc.	1-shot Acc. w/ D	5-shot Acc. w/ D
Supervised	46.52 ± 0.52	66.15 ± 0.22	46.52 ± 0.52	66.15 ± 0.22
Semi-Supervised Inference	50.74 ± 0.75	69.37 ± 0.26	48.67 ± 0.60	67.46 ± 0.24
Soft k-Means	51.52 ± 0.36	**70.25 ± 0.31**	49.88 ± 0.52	68.32 ± 0.22
Soft k-Means+Cluster	**51.85 ± 0.25**	69.42 ± 0.17	**51.36 ± 0.31**	67.56 ± 0.10
Masked Soft k-Means	**52.39 ± 0.44**	**69.88 ± 0.20**	**51.38 ± 0.38**	**69.08 ± 0.25**

Table 3: *tiered*ImageNet 1/5-shot classification results.

Baseline

Supervised : Prototypical network

Semi-supervised Inference : Soft k-means is only used to refine prototypes, not to train embedding function

Experiments

Results (2/2)

unlabeled example의 개수를 늘릴수록 model performance가 증가함을 보임

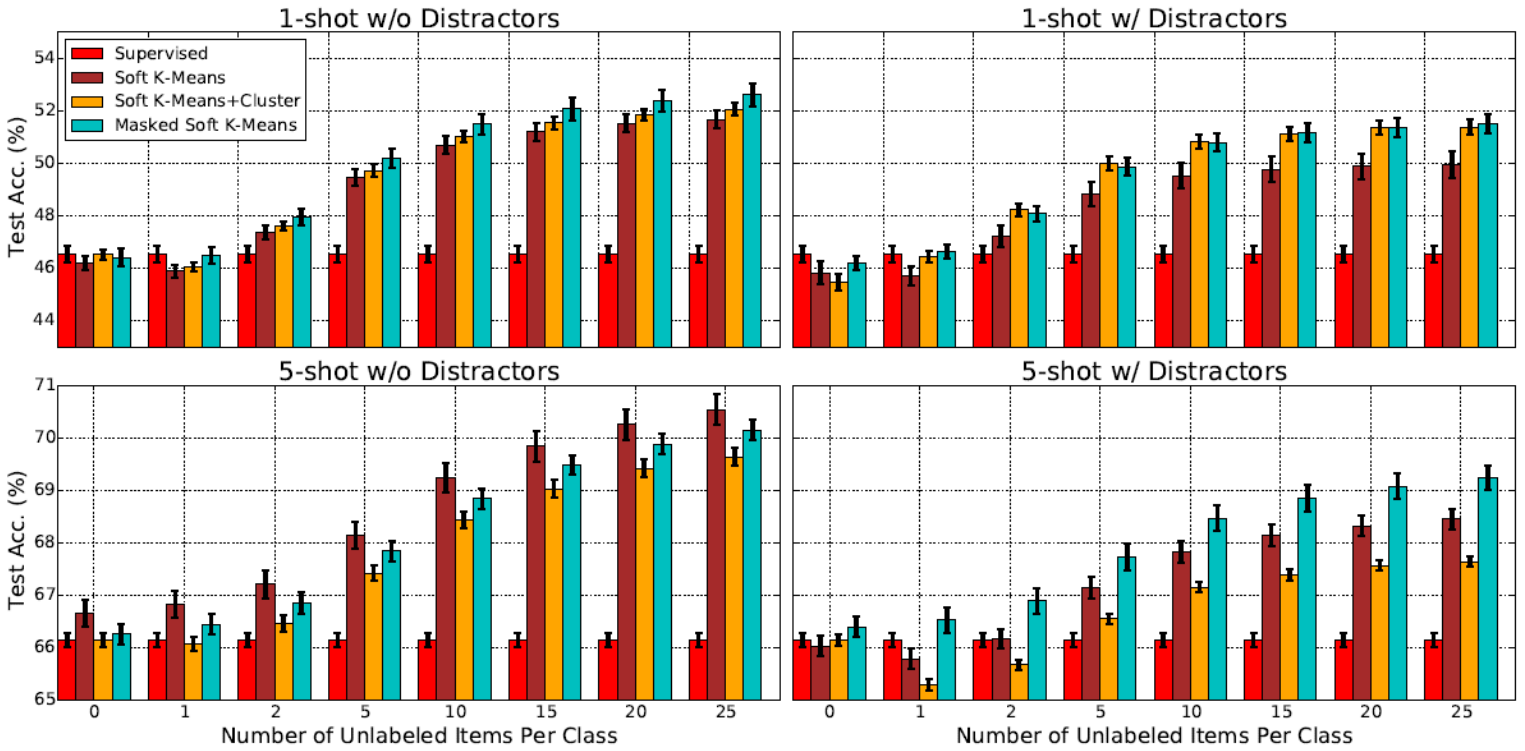


Figure 4: Model Performance on *tiered*ImageNet with different numbers of unlabeled items during test time.

Conclusion

본 논문에서는 기존의 episode 방식의 few-shot learning paradigm에 unlabeled example을 덧붙여 활용하는 semi-supervised few-shot learning paradigm을 제안

Future work

- 아래의 두 가지 연구를 통합 및 이 연구에 적용하여, episode 구성에 따라 example이 다양한 embedding representation을 갖도록 하는 것이 목표
  - ✓ Ba, Jimmy, et al. "Using fast weights to attend to the recent past." *Advances in Neural Information Processing Systems*. 2016.
  - ✓ Finn, Chelsea, Pieter Abbeel, and Sergey Levine. "Model-agnostic meta-learning for fast adaptation of deep networks." *arXiv preprint arXiv:1703.03400* (2017).

Q & A

감사합니다.