Learning with Data (Annotations) Absent

An Opinionated Introduction

Liangzu Peng

School of Information Science and Technology
ShanghaiTech University
penglz@shanghaitech.edu.cn

July 24, 2018

Learning with Data (Annotations) Absent

Outline

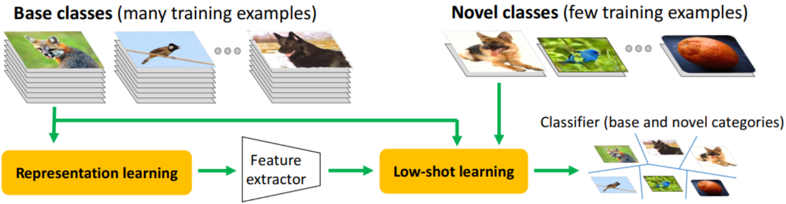
- Low-shot learning
  - a problem setting motivated by real-world scenario
- Related concepts and/or settings
  - Meta-learning as a low-shot training strategy
- Pseudo-data generation for low-shot learning[1,2]
- SGM loss for low-shot learning[2]

[1] Wang, Y. X., Girshick, R., Hebert, M., & Hariharan, B. (2018). Low-Shot Learning from Imaginary Data. arXiv preprint arXiv:1801.05401.

[2] Hariharan, B., & Girshick, R. B. (2017, October). Low-Shot Visual Recognition by Shrinking and Hallucinating Features. In ICCV (pp. 3037-3046).

Learning from data with annotations (partially) absent

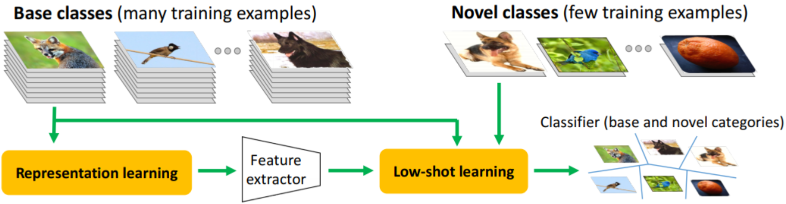
Low-shot Learning: a problem setting motivated by real-world scenario



1. People collect data (base classes) and label them.
   - labeling is expensive.
   - base classes with many training examples.
   - enable the training.

Learning from data with annotations (partially) absent

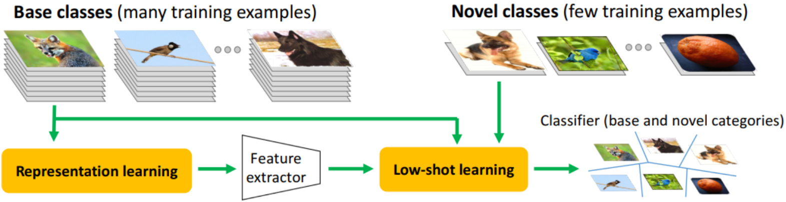
Low-shot Learning: a problem setting motivated by real-world scenario



1. People collect data (base classes) and label them.
   - labeling is expensive.
   - base classes with many training examples.
   - enable the training.
2. New data is coming (containing novel classes) without labels.
   - only able to label little to none of them.
   - novel classes with few training examples.

Learning from data with annotations (partially) absent

Low-shot Learning: a problem setting motivated by real-world scenario



1. People collect data (base classes) and label them.
   - labeling is expensive.
   - base classes with many training examples.
   - enable the training.
2. New data is coming (containing novel classes) without labels.
   - only able to label little to none of them.
   - novel classes with few training examples.
   - How to learn in a way (Low-shot Learning) such that
     - accuracy can be achieved as better as possible.
     - labeling is required as less as possible.

Learning from data with annotations

Related Settings.

Let $\mathcal{Q} = \{S_i, T_i\}_{i=1,\dots,n+1}$ be a list of training-test set pairs.

Learning from data with annotations

Related Settings.

Let $\mathcal{Q} = \{S_i, T_i\}_{i=1,\dots,n+1}$ be a list of training-test set pairs.

- Transfer Learning (naive)
  - TRAIN: train on S_1 and fine-tune on S_2.
  - TEST: test on T_2.
  - EXAMPLE: train on ImageNet, fine-tune and test on Cifar-10.
    - final layers replaced by a new classifier for new classes.

Learning from data with annotations

Related Settings.

Let $\mathcal{Q} = \{S_i, T_i\}_{i=1,\ldots,n+1}$ be a list of training-test set pairs.

- Transfer Learning (naive)
  - TRAIN: train on S_1 and fine-tune on S_2.
  - TEST: test on T_2.
  - EXAMPLE: train on ImageNet, fine-tune and test on Cifar-10.
    - final layers replaced by a new classifier for new classes.
- Meta-Learning (informal)
  - TRAIN: train on S_i and test on T_i ($i = 1, 2, \ldots, n$).
    - S_i, T_i randomly sampled from $\mathcal{Q}$.
    - test accuracy as a loss provides error signal via BP.
  - TEST: train on S_{n+1} and test on T_{n+1}.

Learning from data with annotations

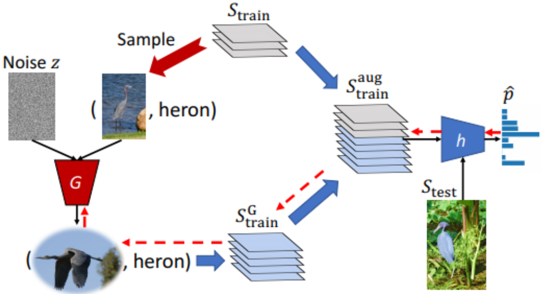
Related Settings.

Let $\mathcal{Q} = \{S_i, T_i\}_{i=1,\ldots,n+1}$ be a list of training-test set pairs.

- Transfer Learning (naive)
  - TRAIN: train on S_1 and fine-tune on S_2.
  - TEST: test on T_2.
  - EXAMPLE: train on ImageNet, fine-tune and test on Cifar-10.
    - final layers replaced by a new classifier for new classes.
- Meta-Learning (informal)
  - TRAIN: train on S_i and test on T_i $(i = 1, 2, \ldots, n)$.
    - S_i, T_i randomly sampled from $\mathcal{Q}$.
    - test accuracy as a loss provides error signal via BP.
  - TEST: train on S_{n+1} and test on T_{n+1}.
  - REMARK: meta-learning as a low-shot training strategy
    - TRAIN $\iff$ the first stage in low-shot setting
    - TEST $\iff$ the second stage in low-shot setting

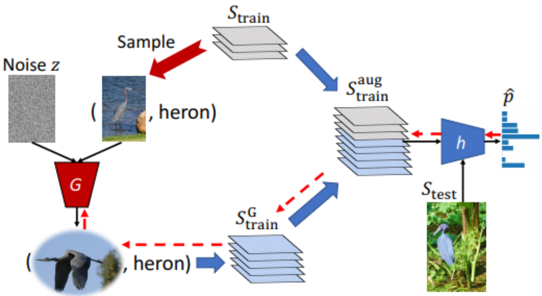
Pseudo-data generation for low-shot learning

Wang et al., CVPR' 2018



Pseudo-data generation for low-shot learning

Wang et al., CVPR' 2018



1. noise z + labeled image from $S_{\text{train}} \overset{G}{\Rightarrow}$ pseudo-labeled image.
   - G is learned
2. TRAIN with augmented dataset $S_{\text{train}}^{\text{aug}}$ and S_{test}.

Pseudo-data generation for low-shot learning (cont.)

Hariharan and Girshick, ICCV' 2017

Figure 1: assume that there is a transformation T that sends the image (feature) c_1^a to the image (feature) c_2^a in category a, then T can also send c_1^b to c_2^b for category b ($c_1^a\!:\!c_2^a\!::\!c_1^b\!:\!c_2^b$).

Pseudo-data generation for low-shot learning (cont.)

Hariharan and Girshick, ICCV' 2017

Figure 1: assume that there is a transformation T that sends the image (feature) c_1^a to the image (feature) c_2^a in category a, then T can also send c_1^b to c_2^b for category b (c_1^a:c_2^a::c_1^b:c_2^b).

- generate quadruplet $(c_1^a, c_2^a, c_1^b, c_2^b)$ for training
- train G such that $G(c_1^a, c_2^a, c_1^b) = \hat{c}_2^b$

Pseudo-data generation for low-shot learning (cont.)

Hariharan and Girshick, ICCV' 2017

Figure 1: assume that there is a transformation T that sends the image (feature) c_1^a to the image (feature) c_2^a in category a, then T can also send c_1^b to c_2^b for category b ($c_1^a{:}c_2^a{::}c_1^b{:}c_2^b$).

- generate quadruplet $(c_1^a, c_2^a, c_1^b, c_2^b)$ for training as follows:
  - for each c_1^a, c_2^a in a, find c_1^b, c_2^b such that
    - cosine distance of $c_1^a - c_2^a$ and $c_1^b - c_2^b$ is minimized.
- train G such that $G(c_1^a, c_2^a, c_1^b) = \hat{c}_2^b$

Pseudo-data generation for low-shot learning (cont.)

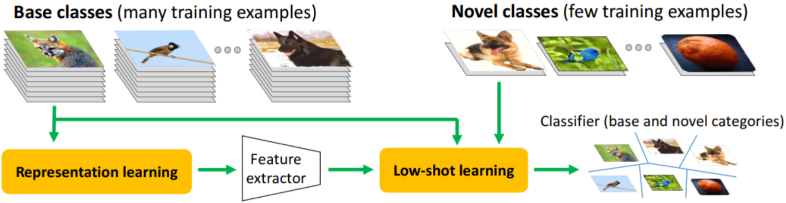
Hariharan and Girshick, ICCV' 2017

Figure 1: assume that there is a transformation T that sends the image (feature) c_1^a to the image (feature) c_2^a in category a, then T can also send c_1^b to c_2^b for category b (c_1^a:c_2^a::c_1^b:c_2^b).

- generate quadruplet $(c_1^a, c_2^a, c_1^b, c_2^b)$ for training as follows:
  - for each c_1^a, c_2^a in a, find c_1^b, c_2^b such that
    - cosine distance of $c_1^a - c_2^a$ and $c_1^b - c_2^b$ is minimized.
- train G such that $G(c_1^a, c_2^a, c_1^b) = \hat{c}_2^b$, with losses
  - $L_{\text{MSE}}(c_2^b, \hat{c}_2^b)$ and
  - $L_{\text{cls}}(\hat{c}_2^b, b)$ (a fixed classifier is given).

Squared Gradient Magnitude Loss for Low-shot Learning

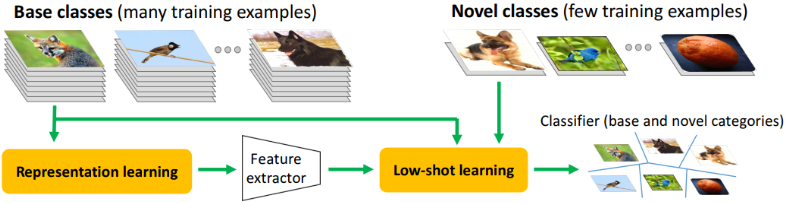
Hariharan and Girshick, ICCV' 2017



- train a feature extractor ϕ and a classifier W on dataset D (i.e., labeled base classes).
  - $L_D(\phi, W) = \frac{1}{|D|}\Sigma_{(x,y)\in D} L_{cls}(W, \phi(x), y)$.
    - $L_{cls}(W, x', y) = -\log p_y(W, x)$.
    - $p_k(W, x) = \frac{exp(w_k^T x)}{\Sigma_j exp(w_j^T x)}$.

Squared Gradient Magnitude Loss for Low-shot Learning

Hariharan and Girshick, ICCV' 2017

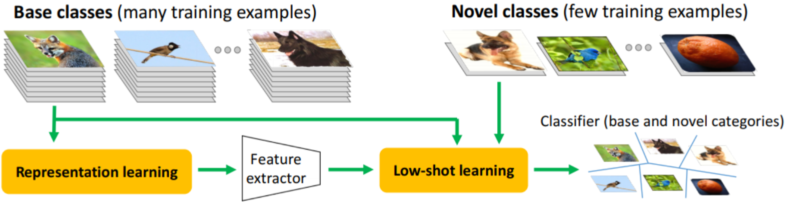


- train a feature extractor ϕ and a classifier W on dataset D (i.e., labeled base classes).
  - $L_D(\phi, W) = \frac{1}{|D|}\Sigma_{(x,y)\in D} L_{cls}(W, \phi(x), y)$.

- feed new data into the trained model will give a result
  - accuracy decreases (novel classes are unseen).

Squared Gradient Magnitude Loss for Low-shot Learning

Hariharan and Girshick, ICCV' 2017



- train a feature extractor ϕ and a classifier W on dataset D (i.e., labeled base classes).
  - $L_D(\phi, W) = \frac{1}{|D|}\Sigma_{(x,y)\in D} L_{cls}(W, \phi(x), y)$.

- feed new data into the trained model will give a result
  - accuracy decreases (novel classes are unseen).
- How to modify the TRAIN process to improve accuracy?
  - simulate low-shot learning experiments on the base classes.

Squared Gradient Magnitude Loss for Low-shot Learning

Hariharan and Girshick, ICCV' 2017

simulate low-shot learning experiments on the base classes:

Squared Gradient Magnitude Loss for Low-shot Learning

Hariharan and Girshick, ICCV' 2017

simulate low-shot learning experiments on the base classes:

- train a feature extractor ϕ and a classifier W on dataset D (i.e., labeled base classes).
  - $L_D(\phi, W) = \frac{1}{|D|}\Sigma_{(x,y)\in D} L_{cls}(W, x', y)$, where $x' = \phi(x)$.
    - $L_{cls}(W, x', y) = -\log p_y(W, x)$.
    - $p_k(W, x) = \frac{exp(w_k^T x)}{\Sigma_j exp(w_j^T x)}$.

Squared Gradient Magnitude Loss for Low-shot Learning

Hariharan and Girshick, ICCV' 2017

simulate low-shot learning experiments on the base classes:

- train a feature extractor ϕ and a classifier W on dataset D (i.e., labeled base classes).
  - $L_D(\phi, W) = \frac{1}{|D|}\Sigma_{(x,y)\in D} L_{cls}(W, x', y)$, where $x' = \phi(x)$.

- Let $S \subset D$ be a tiny training set.

Squared Gradient Magnitude Loss for Low-shot Learning

Hariharan and Girshick, ICCV' 2017

simulate low-shot learning experiments on the base classes:

- train a feature extractor ϕ and a classifier W on dataset D (i.e., labeled base classes).
  - $L_D(\phi, W) = \frac{1}{|D|}\Sigma_{(x,y)\in D} L_{cls}(W, x', y)$, where $x' = \phi(x)$.

- Let $S \subset D$ be a tiny training set.
  - loss for training a classifier V on S:

$$L_S(\phi, V) = \frac{1}{|S|}\Sigma_{(x,y)\in S} L_{cls}(V, \phi(x), y).$$

Squared Gradient Magnitude Loss for Low-shot Learning

Hariharan and Girshick, ICCV' 2017

simulate low-shot learning experiments on the base classes:

- train a feature extractor ϕ and a classifier W on dataset D (i.e., labeled base classes).
  - $L_D(\phi, W) = \frac{1}{|D|}\Sigma_{(x,y)\in D} L_{cls}(W, x', y)$, where $x' = \phi(x)$.
- Let $S \subset D$ be a tiny training set.
  - loss for training a classifier V on S:

$$L_S(\phi, V) = \frac{1}{|S|}\Sigma_{(x,y)\in S} L_{cls}(V, \phi(x), y).$$

  - want the minimizer of $L_S(\phi, V)$ to match W, i.e., (note that $L_S(\phi, V)$ is convex in V):

$$\nabla_V L_S(\phi, V)|_{V=W} = 0$$

Squared Gradient Magnitude Loss for Low-shot Learning

Hariharan and Girshick, ICCV' 2017

simulate low-shot learning experiments on the base classes:

- train a feature extractor ϕ and a classifier W on dataset D (i.e., labeled base classes).
  - $L_D(\phi, W) = \frac{1}{|D|} \Sigma_{(x,y) \in D} L_{cls}(W, x', y)$, where $x' = \phi(x)$.

- Let $S \subset D$ be a tiny training set.
  - loss for training a classifier V on S:

$$L_S(\phi, V) = \frac{1}{|S|} \Sigma_{(x,y) \in S} L_{cls}(V, \phi(x), y).$$

  - want the minimizer of $L_S(\phi, V)$ to match W, i.e., (note that $L_S(\phi, V)$ is convex in V):

$$\nabla_V L_S(\phi, V)|_{V=W} = 0 \Rightarrow L_D^{SGM} = \frac{1}{|D|} \Sigma_{(x,y) \in D} ||\nabla_V L_S(\phi, V)|_{V=W}||_2^2.$$

Put It Together

Hariharan and Girshick, ICCV' 2017

1. we have trained a feature extractor ϕ and a classifier W.
2. new data coming (few labeled dataset S_{n+1} and unlabeled dataset T_{n+1}).
   - generate new pseudo-data for S_{n+1}, and finetune.