细粒度自监督学习研究进展
Fine-Grained Self-Supervised Learning

分享人：高逸凡

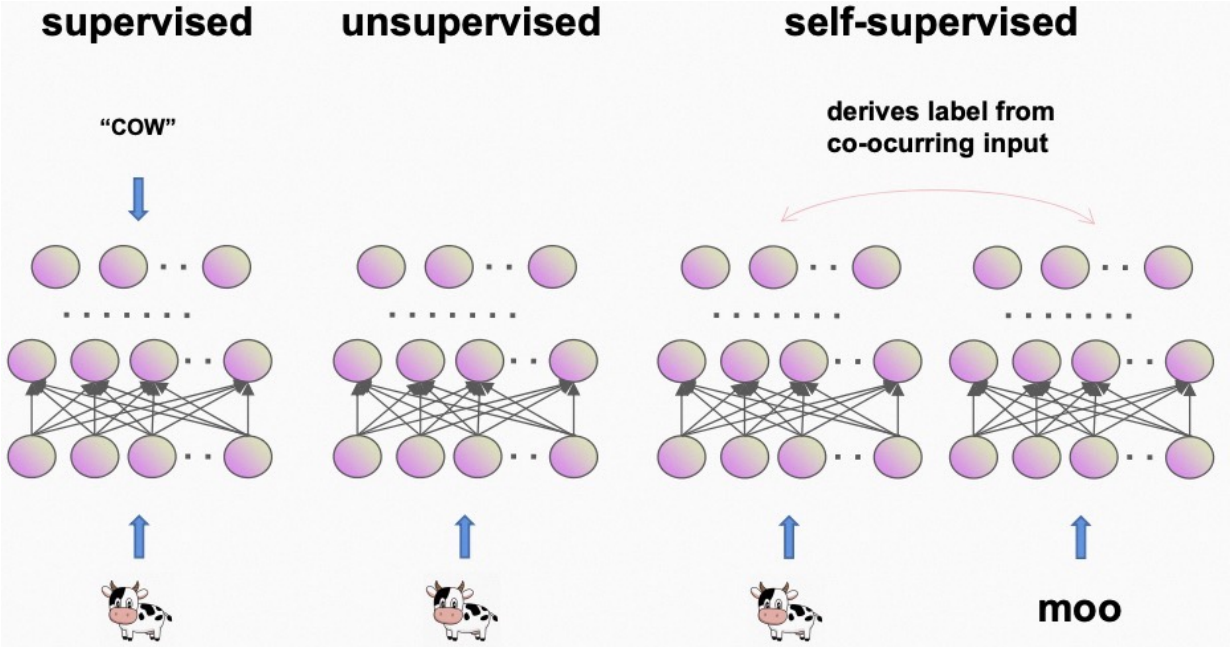
2023.05.08

目录

- 作者介绍
- 研究背景
- 研究动机
- 本文方法
- 实验效果
- 总结反思

研究背景

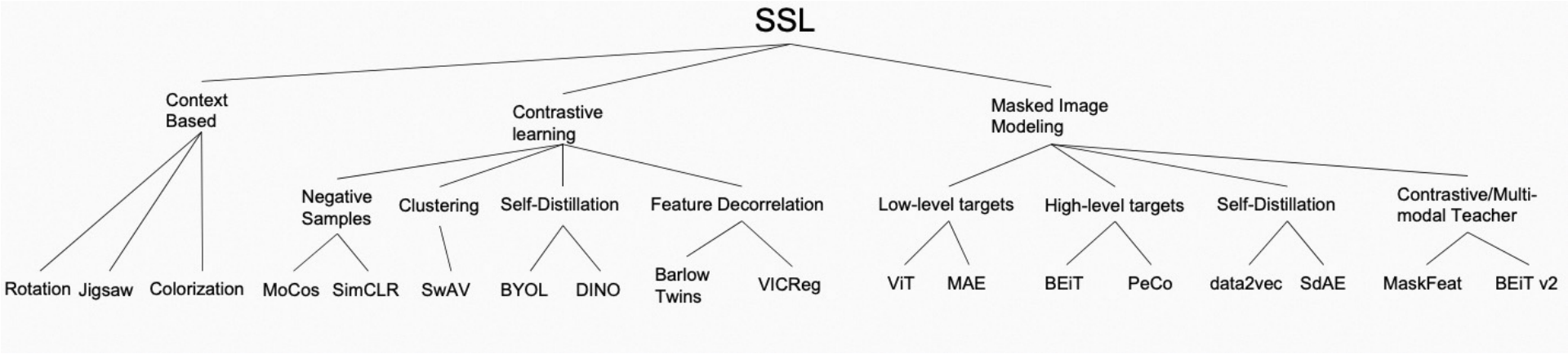
- 自监督学习
  - 一种基于pretext task的无监督学习范式



[1] de Sa V R. Learning classification with unlabeled data[J]. Advances in neural information processing systems, 1994: 112-112.

研究背景

- 自监督学习
  - 依据pretext task划分自监督学习的种类
    - 基于
    - 基于对上下文比学习
    - 基于掩码图像建模（生成式）

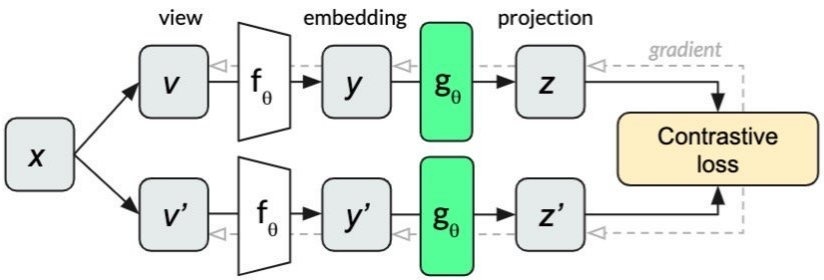


[2] Gui J, Chen T, Cao Q, et al. A Survey of Self-Supervised Learning from Multiple Perspectives: Algorithms, Theory, Applications and Future Trends[J]. arXiv preprint arXiv:2301.05712, 2023.

研究背景

- 对比学习(Contrastive Learning)

一、SimCLR



1. augmentation $x = t(I)$ $x' = t'(I)$

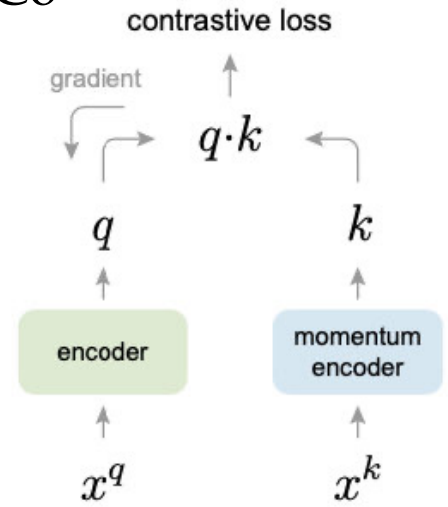
2.encode $u = f_\theta(x)$ $u' = f_\theta(x')$

3.project $z = g_\theta(u)$ $z' = g_\theta(u')$

(z, z') Positive (z, z_k) Negative

4.Loss $$\mathcal{L}_{CL} = -\log \frac{\exp(z \cdot z'/t)}{\sum_{i=0}^{Q} \exp(z \cdot z_i/t)},$$

二、MoCo



Difference：

1.用queue 保存 Image embedding

2.其中一个encoder 用EMA去update

研究背景 、

- 数据集和验证
  - Pretrain
    - ImageNet-1k、CIFAR-10/CIFAR-100
  - Evaluation
    - Linear probe：fix the backbone，train a new classifier.
    - Semi-supervised：fine-tuning in few labels
    - Transfer learning：fine-tuning in PASCAL、COCO

研究背景

- 细粒度自监督学习
  - 任务设置
    - Pretrained and Evaluated both in the fine-grained dataset
  - 研究动机
    - 细粒度图像需要专家标注 (SimCore)
    - 细粒度任务需要通用模型用于下游任务 (SimCore)
    - 对比学习存在 “coarse-grained bias” (GFB)
  - 论文
    - Self-Supervised Learning for Fine-Grained Image Classification (Arxiv 2021)
    - 1.Exploring Localization for Self-supervised Fine-grained Contrastive Learning (BMVC 2022)
    - 2.Coreset Sampling from Open-Set for Fine-Grained Self-Supervised Learning (CVPR 2023)
    - 3.Learning Common Rationale to Improve Self-Supervised Representation for Fine-Grained Visual Recognition Problems (CVPR2023)

- 作者介绍
- 研究动机
- GFB
- SimCore
- 实验效果
- 总结反思

GFB

Learning Common Rationale to Improve Self-Supervised Representation for Fine-Grained Visual Recognition Problems

Yangyang Shu Anton van den Hengel Lingqiao Liu*

School of Computer Science, The University of Adelaide

{yangyang.shu,anton.vandenhengel,lingqiao.liu}@adelaide.edu.au

GFB

- 作者

Lingqiao

Senior Lecturer in the University of Adelaide

Adelaide, Australia

Email

Google Scholar

Welcome to Lingqiao Liu's academic homepage!

I am currently a Senior Lecturer at the School of Computer Science, The University of Adelaide, Australia. I am also an Academic Member of the Australian Institute for Machine Learning. He is a recipient of ARC DECRA (Discovery Early Career Researcher Award) award in 2016 and the University of Adelaide Research Fellowship award in 2016.

I have broad interests in machine learning, computer vision and natural language processing. My research objective is to build practical machine learning systems that can be more data efficient and generalisable for real-world applications. My current major research topics are about: low-supervision machine learning, including semi-supervised learning, unsupervised learning, few-shot/zero-shot learning etc; generalisable machine learning systems, including domain generalization, compositional generalization,etc.; applications in computer vision, e.g., dense prediction, fine-grained recognition, content generation; applications in natural language processing, e.g., low-resource NLP, generalization of NLP systems.

News

1. (February 2023) 5 papers accepted to appear at CVPR 2023

https://lingqiao-adelaide.github.io/lingqiaoliu.github.io//

GFB

- 作者

Rationale-guided Machine Learning

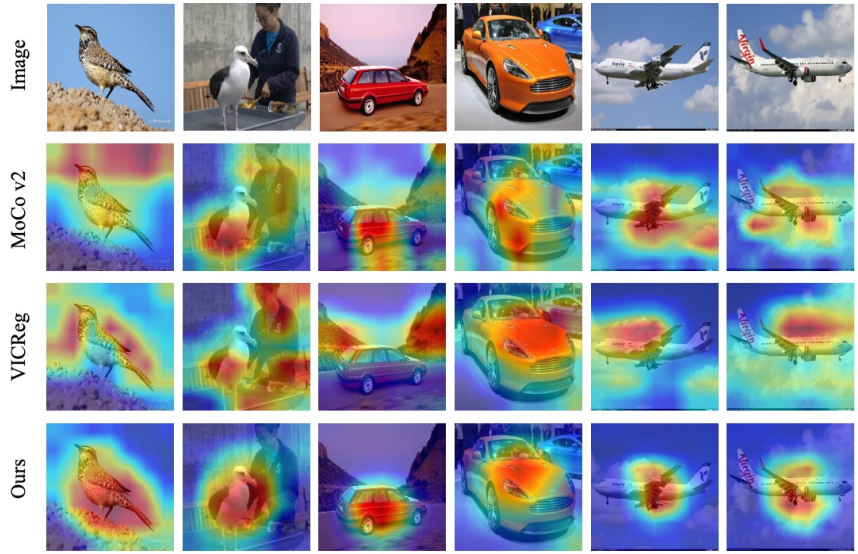
The most machine learning system is based on the principle of Empirical Risk Minimization. Any features and classifiers that contribute to risk minimization will be acquired from the learning process. In this research theme, we consider prediction rationale" – clues about why a certain decision is made in the learning process. We are investigating how to represent rationale and how to put forward various regularizations on the rationale clues. This is expected to lead to more generalizable or more data-efficient machine learning systems.

Related Publications

- Yangyang Shu, Baosheng Yu, Haiming Xu, Lingqiao Liu*: Improving Fine-grained Visual Recognition in Low Data Regimes via Self-boosting Attention Mechanism. ECCV 2022
- Yangyang Shu, Anton van den Hengel, Lingqiao Liu*: Learning Common Rationale to Improve Self-Supervised Representation for Fine-Grained Visual Recognition Problems. CVPR 2023

GFB

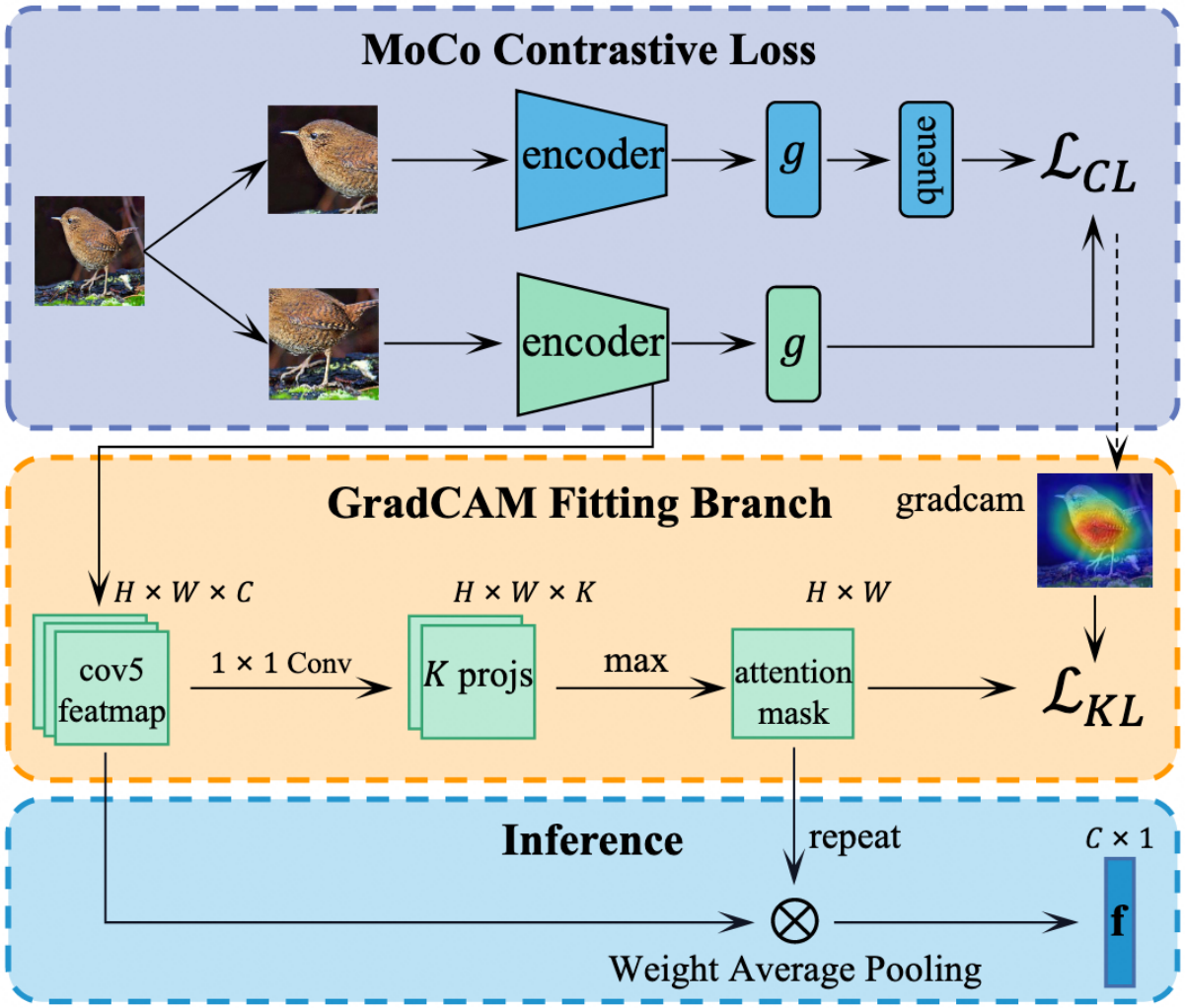
- 方法动机
  - 对比学习存在 “coarse-grained bias”



  - 现有方法利用saliency detectors / part detectors 去regularize SSL，但是saliency regions 和 discriminative regions 可能不一致

GFB

- 方法


MoCo Contrastive Loss
encoder
g
queue
$\mathcal{L}_{CL}$
encoder
g
GradCAM Fitting Branch
gradcam
$H \times W \times C$
$H \times W \times K$
$H \times W$
cov5 featmap
1 × 1 Conv
K projs
max
attention mask
$\mathcal{L}_{KL}$
Inference
repeat
$C \times 1$
f
Weight Average Pooling

GFB

- 主要实验（linear probing 和 retrieval ）

Table 2. Classification and retrieval of our method evaluated on the CUB-200-2011, Stanford Cars and FGVC Aircraft datasets. "ResNet-50" represents pre-training in the ImageNet dataset [12] in a supervised manner, then freezing the ResNet-50 backbone and only optimizing the supervised linear classifier in the classification task. We report the Top 1 and Top 5 (in %) on the classification task, rank-1, rank-5, and mAP (in %) on the retrieval task. 100, 50, and 20 are the three different label proportions (in %) in the classification task.

Dataset	Method	Classification			Retrieval		
		Top 1/Top 5(100)	Top 1/Top 5(50)	Top 1/Top 5(20)	rank-1	rank-5	mAP
CUB-200-2011	ResNet-50	68.17/90.42	58.99/85.90	46.54/77.09	10.65	29.32	5.09
	MoCo v2	68.30/90.85	60.96/87.00	46.91/76.59	17.07	41.46	8.13
	Ours	**71.31/92.03**	**66.52/90.06**	**55.33/83.52**	**49.69**	**75.23**	**24.01**
Stanford Cars	ResNet-50	57.41/83.55	46.23/74.31	31.19/58.67	4.91	16.98	2.34
	MoCo v2	58.43/84.85	50.17/77.38	35.14/64.10	10.94	29.57	3.12
	Ours	**60.75/86.44**	**53.87/81.72**	**40.88/69.18**	**34.56**	**60.75**	**8.87**
FGVC Aircraft	ResNet-50	47.38/74.73	37.83/67.12	28.20/54.73	5.16	14.22	2.61
	MoCo v2	52.54/80.74	45.52/73.85	35.17/65.08	19.38	39.90	6.30
	Ours	**55.87/84.73**	**48.22/77.14**	**38.55/68.53**	**34.33**	**61.09**	**15.43**

GFB

- 次要实验（不同自SSL frameworks）

Table 3. Compared to other state-of-the-art self-supervised learning frameworks with Top 1 accuracy on the CUB-200-2011, Stanford Cars and FGVC Aircraft datasets; running time and peak memory on the CUB-200-2011 dataset. The training time is measured on 4 Tesla V100 GPUs with 100 epochs, and the peak memory is calculated on a single GPU. Top 1 accuracy (%) is reported on linear classification with the frozen representations of their feature extractor. For fairness, all following models use the ResNet-50 as the network backbone and initialize the ResNet-50 architecture with ImageNet-trained weights. blue=best, green=second best.

Method	Batch Size	Top 1(CUB)	Top 1(Cars)	Top 1(Aircraft)	time(CUB)	GPU Memory(CUB)
Supervised	-	81.34	91.02	87.13	-	-
DINO [6]	32/128	12.37/16.66	9.27/10.51	8.52/12.93	1.5h/1.5h	4.9G/8.4G
SimCLR [7]*	32/128	33.49/38.39	44.31/49.41	40.56/45.22	2.5h/2.5h	7.1G/23.8G
BYOL [16]*	32/128	36.64/39.27	43.66/45.21	34.90/37.62	4.0h/4.0h	7.4G/24.6G
SimSiam [9]	32/128	35.82/39.97	56.87/58.89	41.59/43.06	2.0h/2.0h	4.4G/8.9G
MoCo v2 [8]	32/128	68.03/68.30	52.61/58.43	42.51/52.54	2.0h/2.0h	6.1G/10.3G
BarlowTwins [40]	32/128	28.58/33.45	23.34/31.91	28.35/34.77	1.5h/1.5h	7.0G/8.9G
VICReg [4]	32/128	30.07/37.78	19.29/30.80	29.97/36.00	1.0h/1.0h	7.0G/9.0G
Ours	32/128	70.43 /71.31	56.90/60.75	46.92/55.87	2.0h/2.0h	6.1G/10.3G
BYOL+Ours*	32/128	45.79/51.20	48.53 /50.64	40.08/45.94	4.0h/4.0h	7.4G/24.6G

* Due to computational constraints, we are unable to evaluate on the batch size of 4096 as used in the original paper; we leave this for future work.

- 作者介绍
- 研究动机
- GFB
- SimCore
- 实验效果
- 总结反思

SimCore

Coreset Sampling from Open-Set for Fine-Grained Self-Supervised Learning

Sungnyun Kim* Sangmin Bae* Se-Young Yun
KAIST AI
{ksn4397, bsmn0223, yunseyoung}@kaist.ac.kr

SimCore

- 作者

Hi. I'm **Se-Young Yun**.

I'm an associate professor of Graduate School of AI at KAIST and a member of OSI lab. My research interests lie in mathematical modeling and analysis on networks at large, with a specific focus on clustering and learning problems.
Learn about what I do

Contact me : yunseyoung at gmail or yunseyoung at kaist dot ac dot kr

SimCore

Meta Learning

"BOIL: Towards Representation Change for Few-shot Learning" with Jaehoon Oh, Hyungjun Yoo, and ChangHwan Kim, ICLR2021

Community Detection, Clustering

"Clustering in Block Markov Chains" with Alexandre Proutiere and Jaron Sanders, in Annals of Statistics

"Optimal Sampling and Clustering in the Stochastic Block Model" with Alexandre Proutiere, NeurIPS2019

MCMC, BP, Graphical Model

"Spectral Approximate Inference" with Sejun Park, Eunho Yang, and Jinwoo Shin, ICML 2019

Low-Rank Approximation

"Low Rank Approximation for Streaming and Distributed Data" with Jaeseong Jeong and Alexandre Proutiere (submitted)

"Fast and Memory Optimal Low-Rank Matrix Approximation" with Marc Lelarge and Alexandre Proutiere, NIPS 2015

Ranking

"Convergence Rates of Gradient Descent and MM Algorithms for Bradley-Terry Models" with Milan Vojnovic and Kaifang Zhou, AISTATS2020

"Parameter Estimation for Generalized Thurstone Choice Models" with Milan Vojnovic, ICML2016

Reinforcement Learning/Bandit

"Improved Regret Bounds of Bilinear Bandits using Action Space Dimension Analysis" with Jang Kyoungseok, Kwang-Sung Jun, and Wanmo Kang, ICML2021

"Minimal Regret in Online Rec- ommendation Systems" with Kaito Ariu, Narae Ryu, and Alexandre Proutiere, NeurIPS2020

"Reinforcement with fading memories" with Kuang Xu, in Mathematics of Operations Research. Preliminary version: Proceedings of ACM SIGMETRICS 2018.

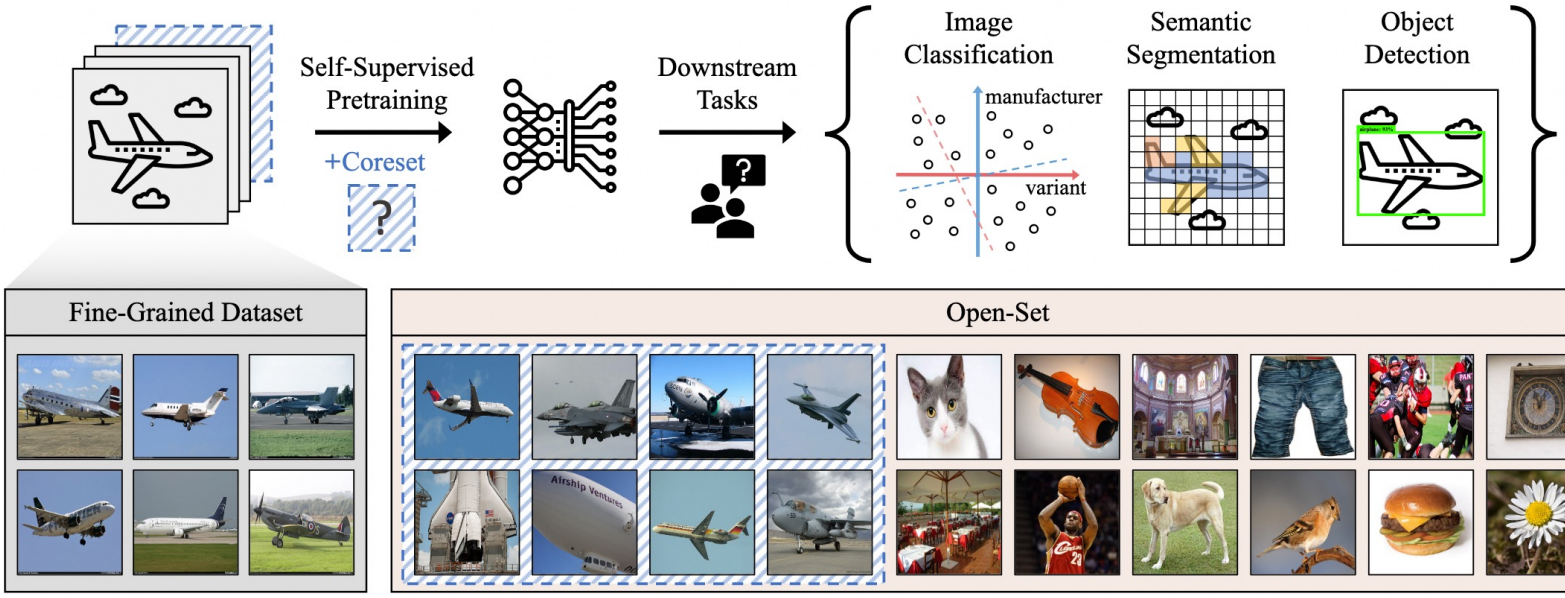
Submodular Optimizatoin

"Test Score Algorithms for Budgeted Stochastic Utility Maximization" preprint

"Sketching with Test Scores and Submodular Maximization" with Shreyas Sekar and Milan Vojnovic, in Management Science arXiv

SimCore

- 研究动机
  - 细粒度任务需要专家标注，需要通用模型(pretrain)
  - 提出任务：Open-SSL，用于大/Web数据(open-set)帮助fine-grained数据集(target-set)做pretrain



SimCore

- 研究动机
  - Target set和open-set存在distribution mismatch.

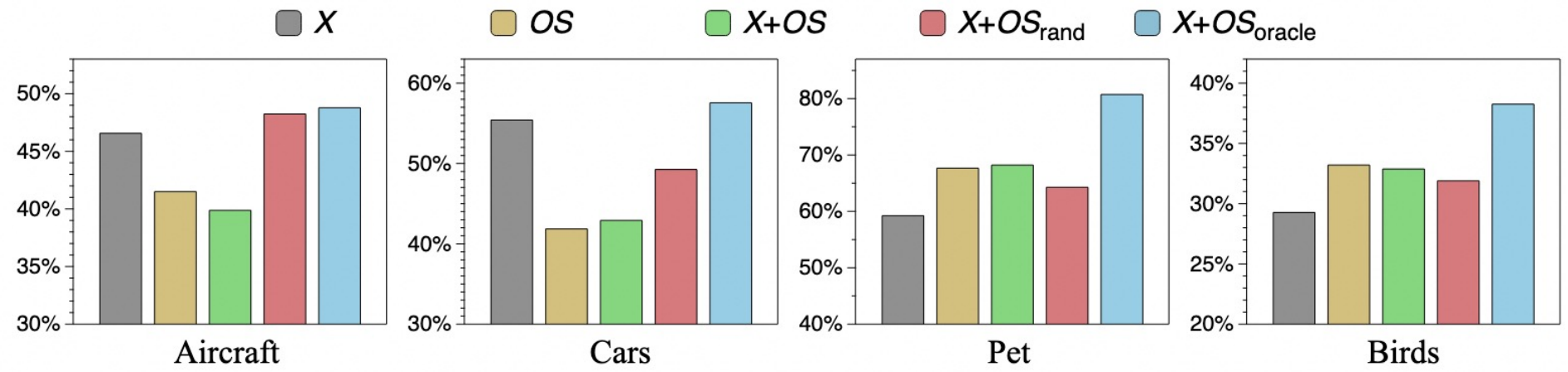


Figure 2. Linear evaluation performance on the fine-grained target dataset. Each color corresponds

SimCore

- 方法
  - 从open-set里采样与target-set相似的数据用于pretrain
    - 符号定义

Open-set $\mathcal{U}$ Target-set X Subset $\mathcal{S}$

- 目标函数

$$f(\mathcal{S}) = \sum_{x \in X} \max_{u \in \mathcal{S}} w(x,u), \quad \text{where } \mathcal{S} \subseteq \mathcal{U},\ \mathcal{U} \cap X = \emptyset \quad (2)$$

$$w(x,u) = z_x^\top z_u$$

SimCore

- 方法
  - 采样算法

Algorithm 1: Simple coreset sampling from open-set

Require: E_θ: encoder pretrained on X;
Require: $\mathcal{U}_0$: initial candidate set (open-set);
Require: $\mathcal{B}$, τ: coreset budget, threshold;
initialize $\mathcal{I} \leftarrow \emptyset$, $t \leftarrow 0$;
replace $\hat{X} \leftarrow$ cluster centroids of X;
calculate $z_x, z_u \leftarrow E_\theta(x), E_\theta(u)$ for $\forall x, u \in \hat{X} \times \mathcal{U}_0$;
while $|\mathcal{I}| < \mathcal{B}$ **do**
    set $\mathcal{S}_t^*$ as the elements in $\mathcal{U}_t$ that are closest to each element in $\hat{X}$ (Eq. 2);
    $\mathcal{I} \leftarrow \mathcal{I} \cup \mathcal{S}_t^*$, $\mathcal{U}_{t+1} \leftarrow \mathcal{U}_t \setminus \mathcal{S}_t^*$
    $t \leftarrow t + 1$
    // stopping criterion
    if $\hat{f}(\mathcal{S}_t^*) < \tau \cdot \hat{f}(\mathcal{S}_1^*)$ **then**
        stop sampling;
    end
end
re-initialize θ and pretrain E_θ with $X \cup \mathcal{I}$;

SimCore


```
def greedy(sim, sampling_nums=0):
    N, K = sim.shape
    queue_list = [deque(torch.argsort(sim[:,k], descending=True).numpy()) for k in range(K)]
    indices, tmp = set(), set()
    # for one iteration, each cluster picks one sample with the maximum utility function (i.e
    FLAG, threshold = 1, 0
    while len(indices) < sampling_nums:
        # for-loop according to the centroids
        func_value = 0
        for q, queue in enumerate(queue_list):
            while True:
                i = queue.popleft()
                if i not in indices: break
            tmp.update({i})
```

```
        indices.update(tmp)
        tmp = set()
```

SimCore

- 实验一（linear probe）

		Target dataset (X) and its number of samples										
pretrain	p	Aircraft 6,667	Cars 8,144	Pet 3,680	Birds 5,990	Dogs 12,000	Flowers 2,040	Action 4,000	Indoor 5,360	Textures 3,760	Faces 4,263	Food 13,296
X	-	46.56	55.42	59.23	29.27	49.88	80.14	43.76	54.10	58.78	56.63	87.99
OS	-	41.50	41.86	67.66	33.21	49.94	85.67	60.65	64.46	67.23	52.84	86.14
X+OS	-	39.88	42.92	68.22	32.88	50.42	85.34	60.61	63.66	67.98	52.76	85.90
X+OS_{rand}	1%	48.24	49.26	64.27	31.90	49.62	83.17	47.25	55.37	61.33	57.37	88.08
X+$OS_{\text{SimCore}\dagger}$	1%	48.06	58.56	74.82	33.37	57.42	82.12	51.37	57.84	61.76	56.95	90.35
X+$\boldsymbol{OS_{\text{SimCore}}}$	1%	**48.45**	<u>59.00</u>	77.13	36.56	59.83	86.70	52.98	59.18	63.40	58.85	89.78
X+OS_{rand}	5%	45.75	46.03	68.38	33.63	50.24	84.52	57.27	60.71	65.80	56.05	87.75
X+$OS_{\text{SimCore}\dagger}$	5%	45.57	50.75	<u>80.20</u>	35.56	64.62	85.11	64.53	68.13	66.22	58.93	89.87
X+$\boldsymbol{OS_{\text{SimCore}}}$	5%	47.14	52.22	**81.75**	**39.21**	**66.82**	**87.28**	<u>66.38</u>	<u>70.96</u>	**68.13**	**59.34**	<u>90.74</u>
Stopping Criterion		1.03%	0.95%	14.4%	13.7%	9.72%	7.96%	15.6%	13.5%	5.89%	0.27%	3.86%
X+$\boldsymbol{OS_{\text{SimCore}}}$	-	<u>48.27</u>	**60.29**	79.66	<u>37.65</u>	<u>66.48</u>	<u>87.04</u>	**67.46**	**71.95**	<u>67.66</u>	<u>59.01</u>	**91.31**

Table 2. Linear evaluation performance on eleven fine-grained datasets. We used ImageNet-1k [20] as an open-set. The ratio p is a fixed budget size to sample from the open-set, either via random sampling (OS_{rand}) or SimCore (OS_{SimCore}). Given that OS has 1.3M samples, $p = 1\%$ corresponds to 13K samples. † denotes SimCore with $k = 1$, a single cluster. **Bold** and <u>underline</u> indicate the best and the second best accuracy for each target dataset, respectively.

SimCore

- 实验二（backbone、SSL methods）

method	architecture	pretrain	Aircraft	Cars	Pet	Birds
SimCLR	EfficientNet	X	25.5	37.0	58.1	27.8
SimCLR	EfficientNet	OS	31.6	29.5	57.8	26.5
SimCLR	EfficientNet	**SimCore**	**41.7**	**52.8**	**69.5**	**29.6**
SimCLR	ResNet18	X	43.4	51.9	58.2	25.9
SimCLR	ResNet18	OS	33.9	33.1	62.5	27.7
SimCLR	ResNet18	**SimCore**	**44.5**	**55.1**	**72.7**	**31.3**
SimCLR	ResNeXt50	X	45.9	56.5	63.4	28.6
SimCLR	ResNeXt50	OS	39.2	39.4	68.2	32.6
SimCLR	ResNeXt50	**SimCore**	**49.5**	**59.5**	**81.0**	**37.4**
SimCLR	ResNet101	X	49.4	54.5	64.0	29.1
SimCLR	ResNet101	OS	40.4	41.9	69.5	34.2
SimCLR	ResNet101	**SimCore**	**50.9**	**58.8**	**83.0**	**39.1**

Table 3. Linear evaluation performance with different architectures. SimCore corresponds to X+OS_{SimCore} with a stopping criterion.

method	architecture	pretrain	Aircraft	Cars	Pet	Birds
BYOL	ResNet50	X	40.6	49.4	56.5	27.6
BYOL	ResNet50	OS	46.1	49.6	78.4	44.7
BYOL	ResNet50	**SimCore**	**46.5**	**50.4**	**85.1**	**47.9**
SwAV	ResNet50	X	34.5	42.4	49.4	21.6
SwAV	ResNet50	OS	33.8	30.0	64.2	27.3
SwAV	ResNet50	**SimCore**	**45.0**	**45.1**	**80.2**	**36.6**
DINO	ViT-Ti/16	X	27.3	**48.2**	42.4	28.5
DINO	ViT-Ti/16	OS	42.0	39.1	78.4	61.2
DINO	ViT-Ti/16	**SimCore**	**43.2**	47.2	**83.3**	**72.6**
MAE	ViT-B/16	X	**55.9**	44.7	56.3	32.2
MAE	ViT-B/16	OS	39.8	37.3	68.3	31.4
MAE	ViT-B/16	**SimCore**	48.1	**52.4**	**77.8**	**42.1**

Table 4. Linear evaluation performance with different SSL methods SimCore corresponds to X+OS_{SimCore} with a stopping criterion.

SimCore

- 实验三（openset、feature distribution）

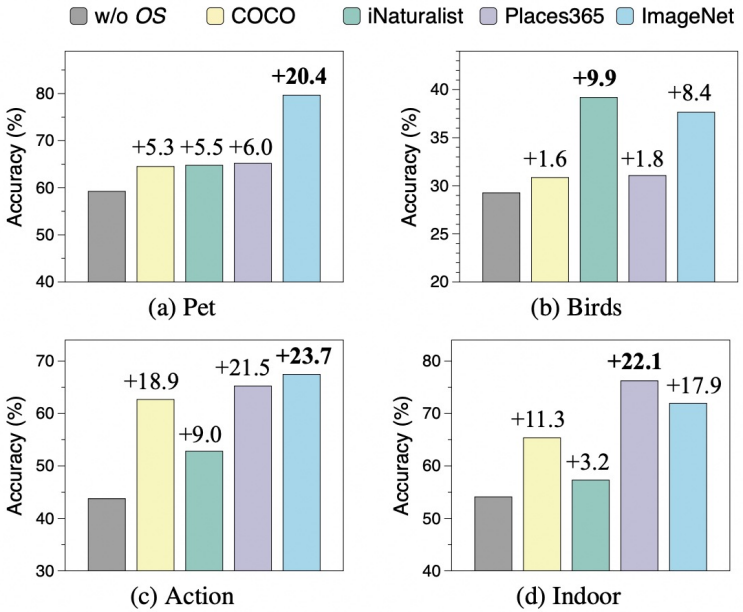


Figure 3. SimCore performances compared to the X pretraining (w/o *OS*). In addition to ImageNet-1k, we included MS COCO, iNaturalist 2021-mini, and Places365 as an open set. For target

OS	Aircraft	Cars	Birds	Pet	Action	Indoor
✗	46.6	55.4	29.3	59.2	43.8	54.1
ImageNet-1k	48.3	60.3	37.7	79.7	**67.5**	72.0
ALL	**58.8**	**64.8**	**38.0**	79.5	67.2	**75.1**
WebVision	48.2	59.1	36.7	**80.3**	67.1	72.6
WebFG-496	55.4	63.1	37.7	-	-	-

Table 5. Linear evaluation of SimCore with uncurated open-sets.

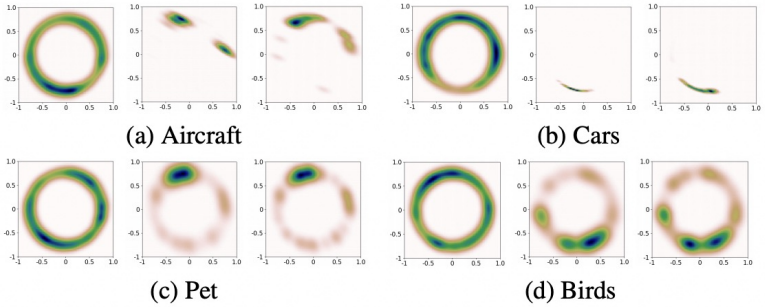
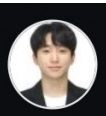


Figure 5. Feature distribution map of *OS* (left), X (middle), and the coreset sampled by SimCore (right).

SimCore

- 实验四（fully-labeled fine-tuning）



raymin0223 commented last month Collaborator ...

Hi and thanks for your good question!

Yes, in **Table 7b**, we summarized **end-to-end supervised fine-tuning** results with SSL pretrained ResNet50 on four fine-grained datasets.
But, we only reported *not* fully-labeled scenarios as in previous SSL works.

Here, we additionally summarize e2e fine-tuning results on fully-labeled (100%) target datasets:

Pretrain	Aircraft	Cars	Pet	Birds
X	79.89	88.63	78.24	67.46
OS	81.28	87.43	85.01	70.12
SimCore	**83.28**	**89.54**	**85.95**	**71.24**

cf) Note that we fine-tuned models for 100 epochs with a momentum SGD optimizer and weight decay of 1e-4.
We searched the optimal learning rate among three logarithmically spaced values from 1e-1 to 1e-2 (*i.e.*, {1e-1, 3e-2, 1e-2}), and they are decayed after 60 and 80 epochs by the ratio of 0.1.

- 作者介绍
- 研究动机
- GFB
- SimCore
- 实验效果
- 总结反思

总结反思

- 一个新任务被认可：在小数据集上做pretrain
- 从模型和数据角度考虑方法设计的motivation
- 方法简单，实验足够充分，写作思路清晰

Thank for your attention !