Prototypical Contrastive Learning of Unsupervised Representations

School of Industrial and Management Engineering, Korea University

Jungin Kim

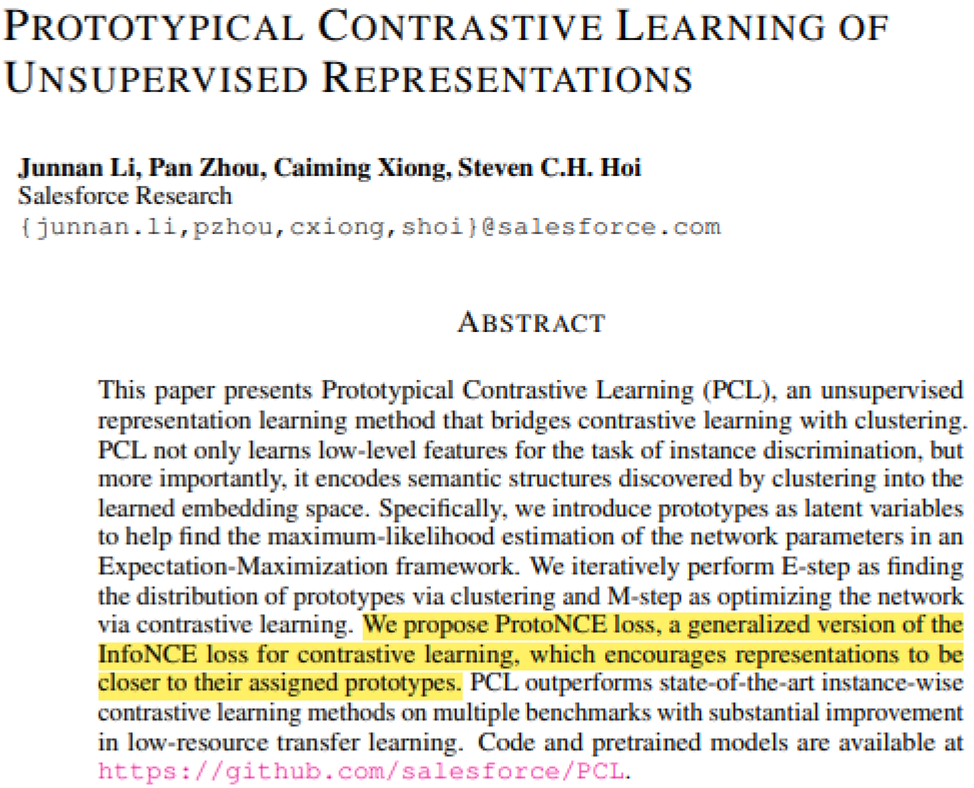
Contents

- Research Purpose
- Proposed Method
- Experiment
- Conclusion

Research Purpose

❖ **Prototypical Contrastive Learning of Unsupervised Representation (ICLR 2021)**

- ✓ Salesforce Research에서 연구하였고, 2022년 6월 13일 기준으로 263회 인용됨
- ✓ **Prototype contrastive loss**를 사용한 새로운 **self supervised learning** 방법론 제안



PROTOTYPICAL CONTRASTIVE LEARNING OF UNSUPERVISED REPRESENTATIONS

Junnan Li, Pan Zhou, Caiming Xiong, Steven C.H. Hoi
Salesforce Research
{junnan.li,pzhou,cxiong,shoi}@salesforce.com

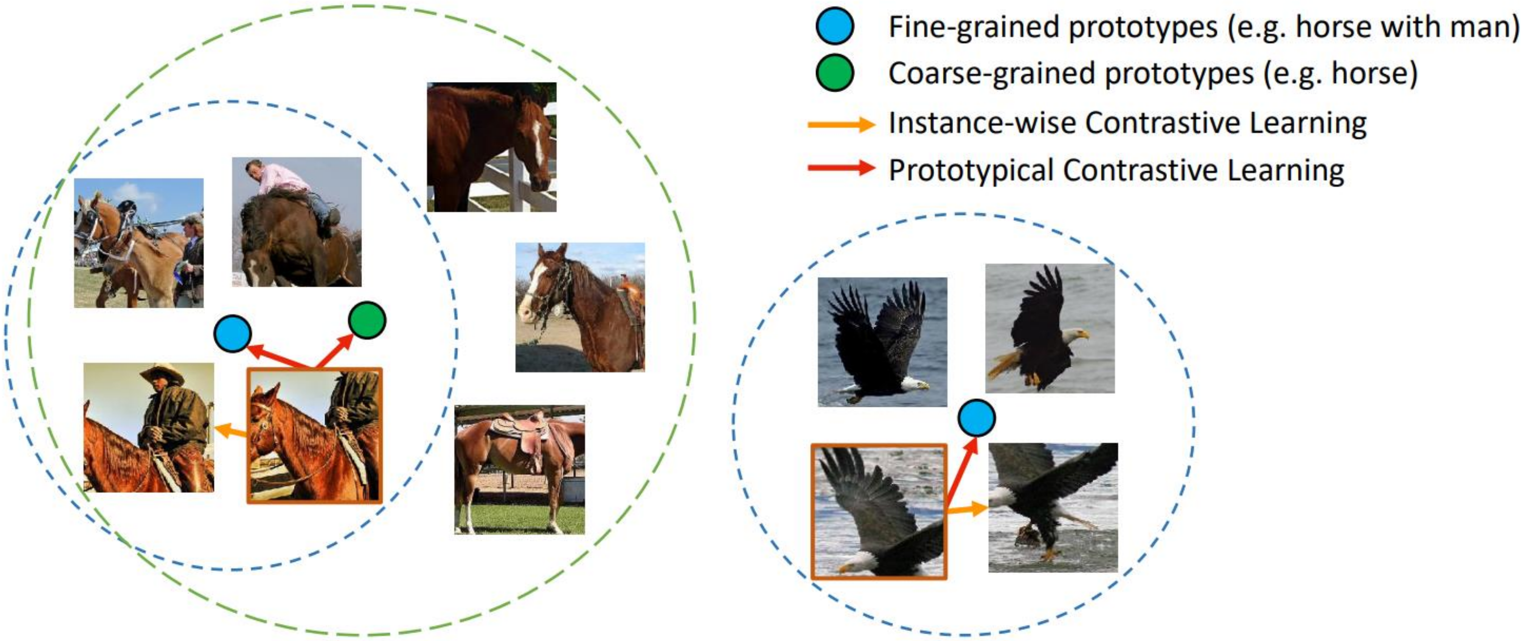
ABSTRACT

This paper presents Prototypical Contrastive Learning (PCL), an unsupervised representation learning method that bridges contrastive learning with clustering. PCL not only learns low-level features for the task of instance discrimination, but more importantly, it encodes semantic structures discovered by clustering into the learned embedding space. Specifically, we introduce prototypes as latent variables to help find the maximum-likelihood estimation of the network parameters in an Expectation-Maximization framework. We iteratively perform E-step as finding the distribution of prototypes via clustering and M-step as optimizing the network via contrastive learning. We propose ProtoNCE loss, a generalized version of the InfoNCE loss for contrastive learning, which encourages representations to be closer to their assigned prototypes. PCL outperforms state-of-the-art instance-wise contrastive learning methods on multiple benchmarks with substantial improvement in low-resource transfer learning. Code and pretrained models are available at https://github.com/salesforce/PCL.

Research Purpose

❖ Main Idea

- 기존 contrastive learning의 문제(비슷한 특징을 지닌 instances를 negative로 정의) 개선
- 비슷한 특징의 instance를 prototype(=clustering) 단위로 묶어 contrastive learning 수행



● Fine-grained prototypes (e.g. horse with man)

● Coarse-grained prototypes (e.g. horse)

→ Instance-wise Contrastive Learning

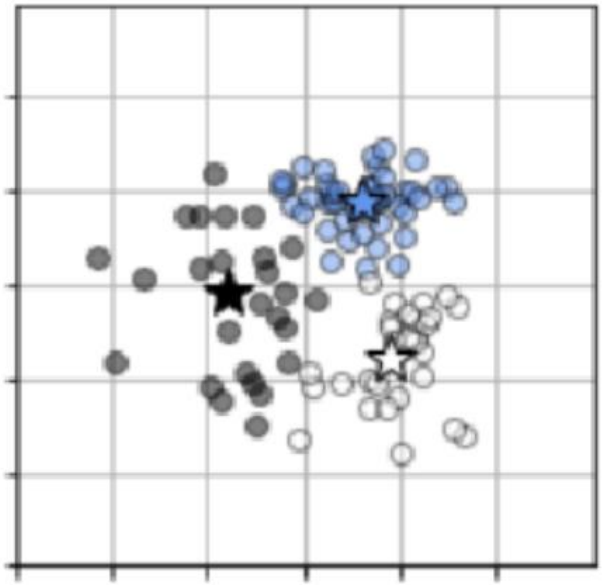
→ Prototypical Contrastive Learning

Proposed Method

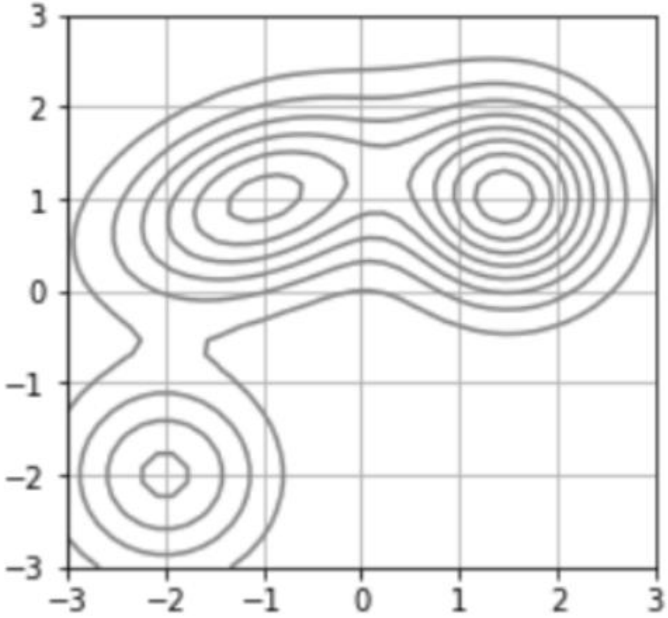
K-means

❖ Preliminary – EM 알고리즘

- 가우시안 혼합 모델을 사용하는 알고리즘으로 확률적 클러스터링에 사용
- K-means와 가우시안 혼합 모델을 이용한 데이터 분포가 서로 비슷함
- 데이터 분포를 가우시안 혼합 모델의 입체적인 분포를 통해 확률로 표현 가능
- 구한 확률을 통해 해당 데이터가 어느 클러스터에 속하는지 확인 가능



K-means

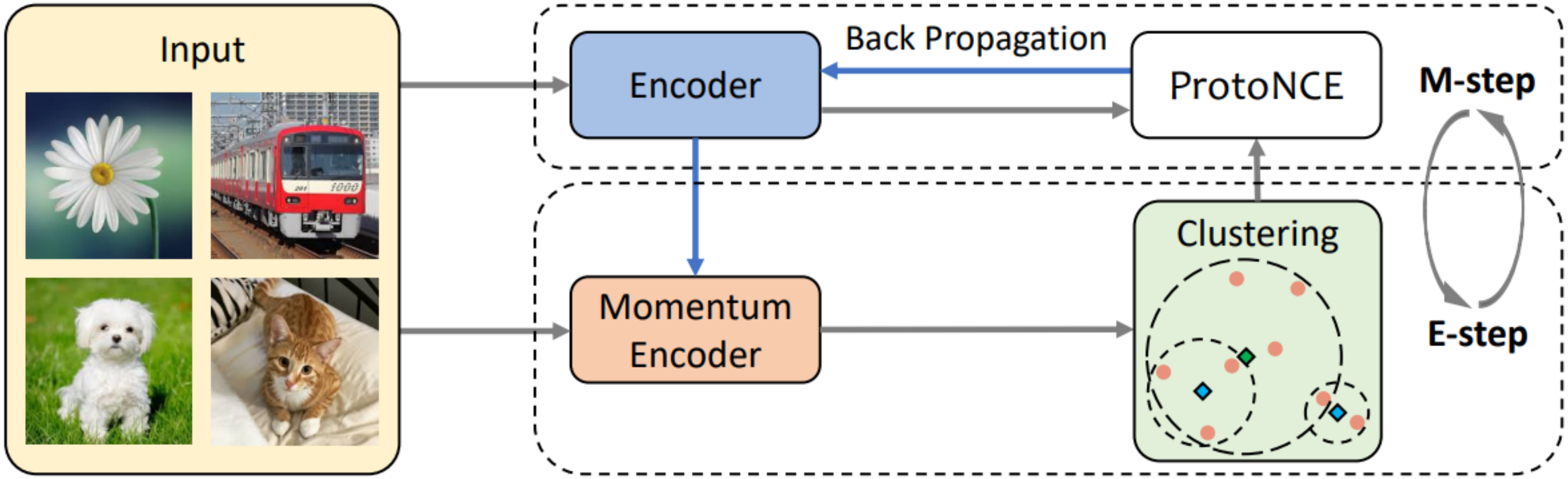


가우시안 혼합 모델

Proposed Method

❖ Framework of PCL

- EM(Expectation Maximization) 기법을 통해 학습 방법 제안
- E-step : 전체 train feature에 대해서 clustering을 수행하여 k개의 cluster로 구분
- M-step : ProtoNCE loss 학습
- Positive Prototype은 가까워지도록, Negative Prototype은 멀어지도록 하는 방식

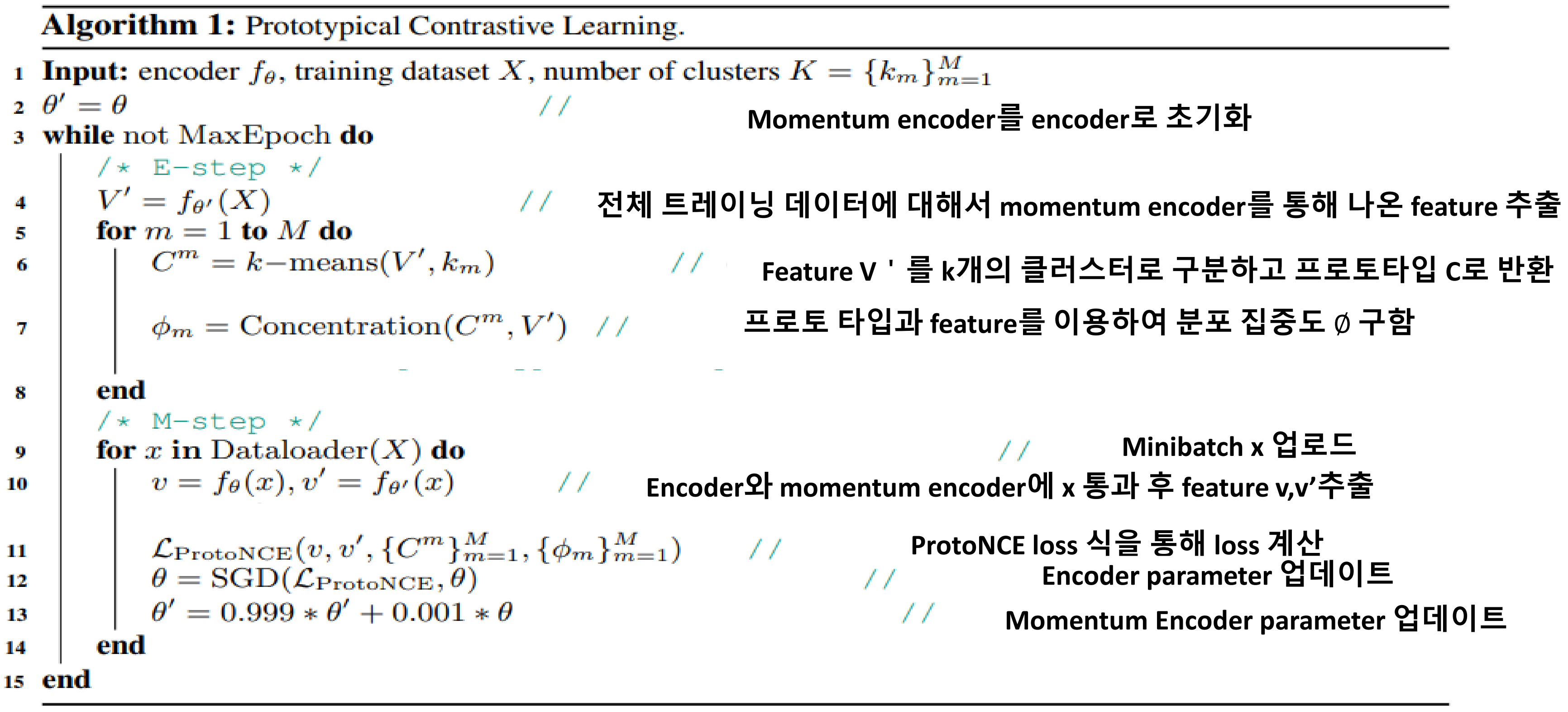


Proposed Method

$$\mathcal{L}_{\text{ProtoNCE}} = \sum_{i=1}^{n} -\left(\log \frac{\exp(v_i \cdot v_i'/\tau)}{\sum_{j=0}^{r} \exp(v_i \cdot v_j'/\tau)} + \frac{1}{M} \sum_{m=1}^{M} \log \frac{\exp(v_i \cdot c_s^m/\phi_s^m)}{\sum_{j=0}^{r} \exp(v_i \cdot c_j^m/\phi_j^m)} \right)$$

❖ Framework of PCL

- Momentum Encoder의 출력 값으로 E-step clustering 수행
- ProtoNCE를 최적화하여 신경망을 M – step 업데이트 진행

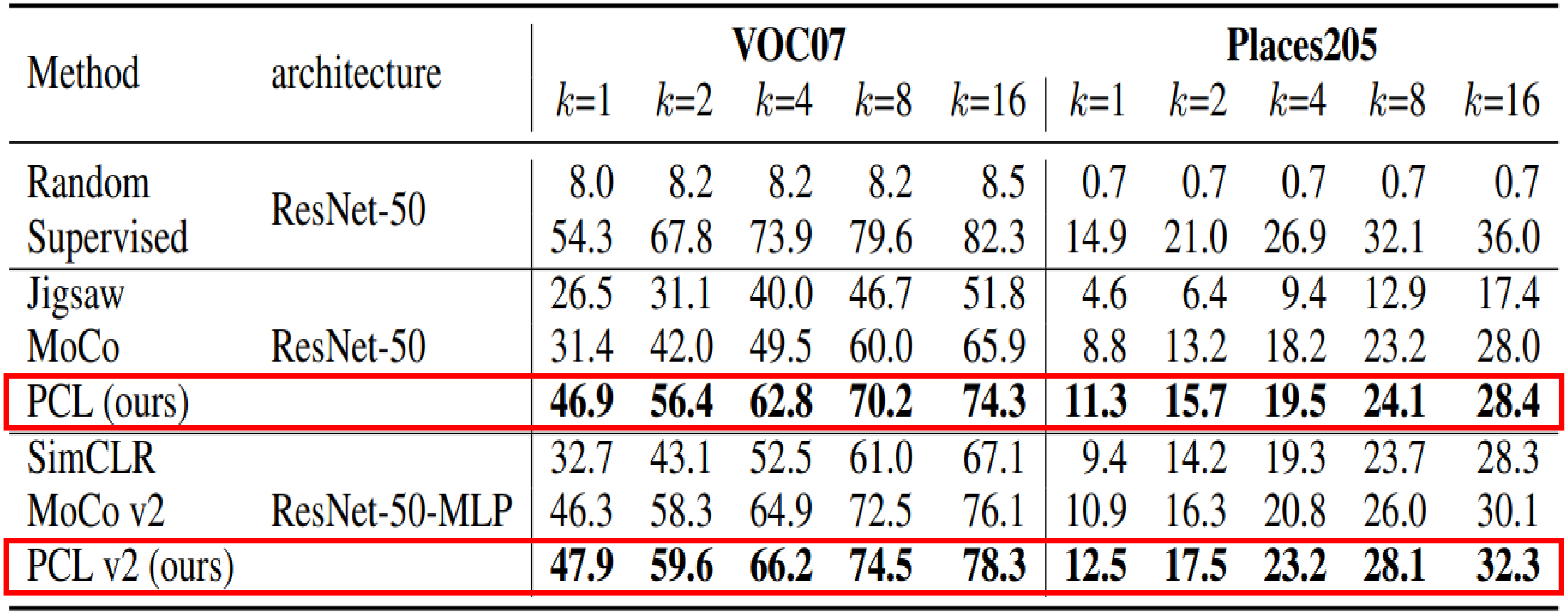


```
Algorithm 1: Prototypical Contrastive Learning.
Input: encoder $f_\theta$, training dataset $X$, number of clusters $K = \{k_m\}_{m=1}^{M}$
$\theta' = \theta$    //    Momentum encoder를 encoder로 초기화
while not MaxEpoch do
    /* E-step */
    $V' = f_{\theta'}(X)$    //    전체 트레이닝 데이터에 대해서 momentum encoder를 통해 나온 feature 추출
    for $m = 1$ to $M$ do
        $C^m = k\text{-means}(V', k_m)$    //    Feature V ' 를 k개의 클러스터로 구분하고 프로토타입 c로 반환
        $\phi_m = \text{Concentration}(C^m, V')$    //    프로토 타입과 feature를 이용하여 분포 집중도 ∅ 구함
    end
    /* M-step */
    for $x$ in Dataloader($X$) do    //    Minibatch x 업로드
        $v = f_\theta(x), v' = f_{\theta'}(x)$    //    Encoder와 momentum encoder에 x 통과 후 feature v,v'추출
        $\mathcal{L}_{\text{ProtoNCE}}(v, v', \{C^m\}_{m=1}^{M}, \{\phi_m\}_{m=1}^{M})$    //    ProtoNCE loss 식을 통해 loss 계산
        $\theta = \text{SGD}(\mathcal{L}_{\text{ProtoNCE}}, \theta)$    //    Encoder parameter 업데이트
        $\theta' = 0.999 * \theta' + 0.001 * \theta$    //    Momentum Encoder parameter 업데이트
    end
end
```

Experiment

❖ Low shot classification

- 카테고리당 훈련 샘플이 거의 없는 이미지 분류 작업에서 성능 평가
- 두 가지 데이터 셋 사용 : Places205 & PASCAL VOC2007
- SimCLR, MOCO보다 좋은 성능 확인

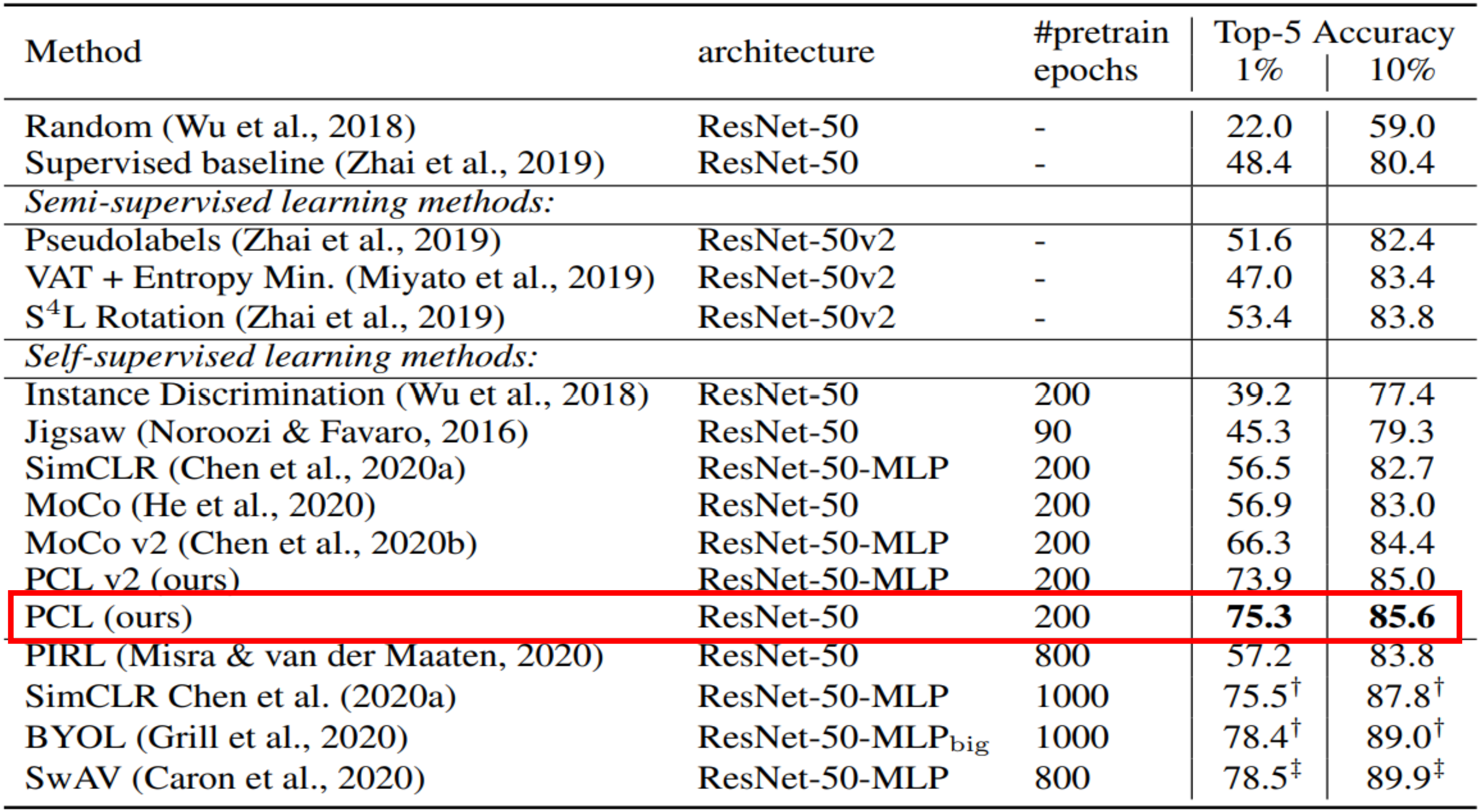


Method	architecture	VOC07					Places205				
		k=1	*k*=2	*k*=4	*k*=8	*k*=16	*k*=1	*k*=2	*k*=4	*k*=8	*k*=16
Random	ResNet-50	8.0	8.2	8.2	8.2	8.5	0.7	0.7	0.7	0.7	0.7
Supervised		54.3	67.8	73.9	79.6	82.3	14.9	21.0	26.9	32.1	36.0
Jigsaw	ResNet-50	26.5	31.1	40.0	46.7	51.8	4.6	6.4	9.4	12.9	17.4
MoCo		31.4	42.0	49.5	60.0	65.9	8.8	13.2	18.2	23.2	28.0
PCL (ours)		**46.9**	**56.4**	**62.8**	**70.2**	**74.3**	**11.3**	**15.7**	**19.5**	**24.1**	**28.4**
SimCLR	ResNet-50-MLP	32.7	43.1	52.5	61.0	67.1	9.4	14.2	19.3	23.7	28.3
MoCo v2		46.3	58.3	64.9	72.5	76.1	10.9	16.3	20.8	26.0	30.1
PCL v2 (ours)		**47.9**	**59.6**	**66.2**	**74.5**	**78.3**	**12.5**	**17.5**	**23.2**	**28.1**	**32.3**

Experiment

❖ Semi-supervised image classification

- Semi-supervised learning과 self-supervised learning 방법론 성능 비교
- 200 epoch 미만에서 SOTA 기록

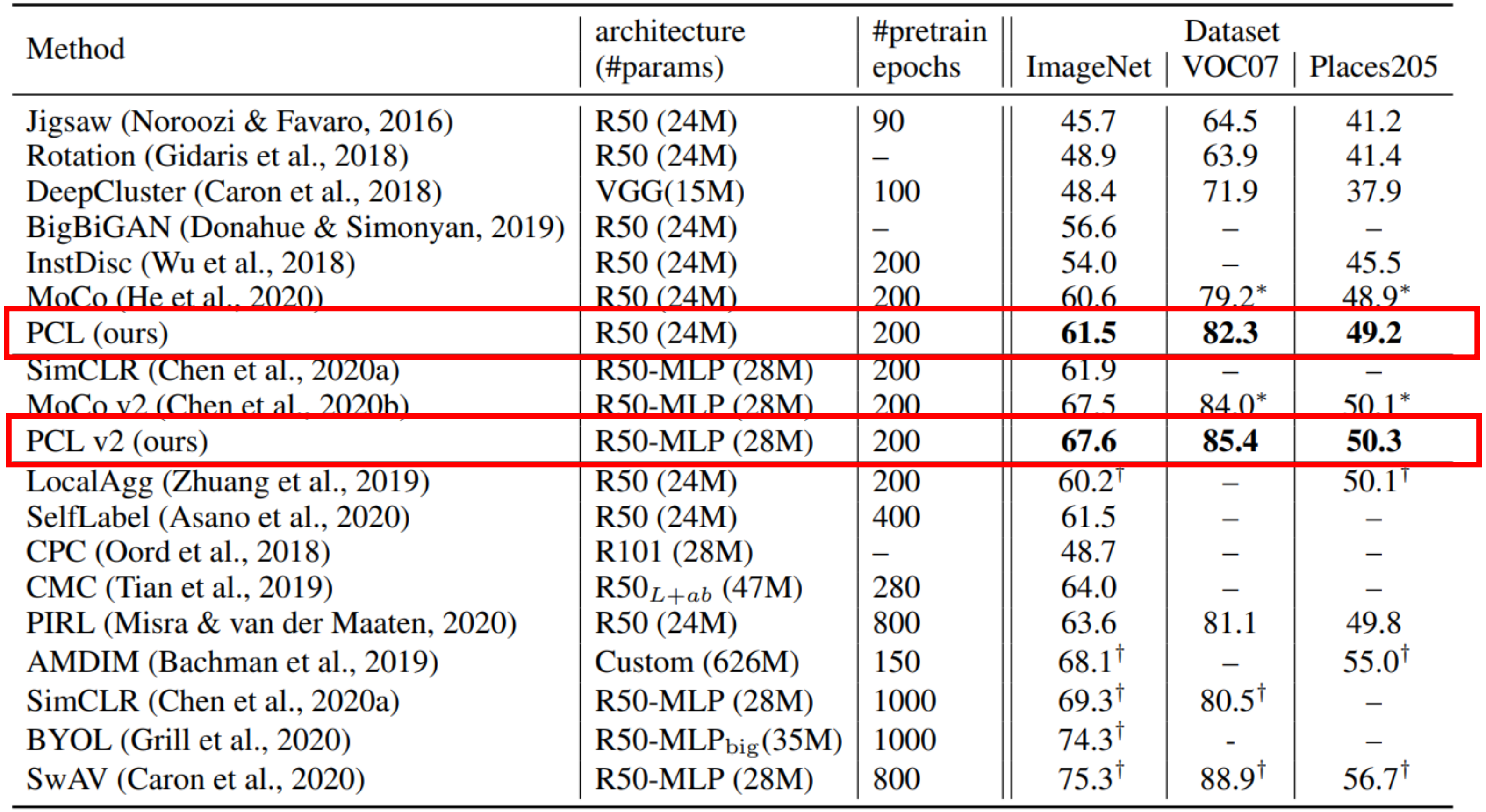


Method	architecture	#pretrain epochs	Top-5 Accuracy 1%	Top-5 Accuracy 10%
Random (Wu et al., 2018)	ResNet-50	-	22.0	59.0
Supervised baseline (Zhai et al., 2019)	ResNet-50	-	48.4	80.4
Semi-supervised learning methods:				
Pseudolabels (Zhai et al., 2019)	ResNet-50v2	-	51.6	82.4
VAT + Entropy Min. (Miyato et al., 2019)	ResNet-50v2	-	47.0	83.4
S^4L Rotation (Zhai et al., 2019)	ResNet-50v2	-	53.4	83.8
Self-supervised learning methods:				
Instance Discrimination (Wu et al., 2018)	ResNet-50	200	39.2	77.4
Jigsaw (Noroozi & Favaro, 2016)	ResNet-50	90	45.3	79.3
SimCLR (Chen et al., 2020a)	ResNet-50-MLP	200	56.5	82.7
MoCo (He et al., 2020)	ResNet-50	200	56.9	83.0
MoCo v2 (Chen et al., 2020b)	ResNet-50-MLP	200	66.3	84.4
PCL v2 (ours)	ResNet-50-MLP	200	73.9	85.0
PCL (ours)	ResNet-50	200	**75.3**	**85.6**
PIRL (Misra & van der Maaten, 2020)	ResNet-50	800	57.2	83.8
SimCLR Chen et al. (2020a)	ResNet-50-MLP	1000	$75.5^{\dagger}$	$87.8^{\dagger}$
BYOL (Grill et al., 2020)	ResNet-50-MLP$_{\text{big}}$	1000	$78.4^{\dagger}$	$89.0^{\dagger}$
SwAV (Caron et al., 2020)	ResNet-50-MLP	800	$78.5^{\ddagger}$	$89.9^{\ddagger}$

Experiment

❖ Image classification with linear classifiers

- 사전 학습된 모델에 parameter를 freeze하고 linear classifier를 학습했을 때 성능 비교
- 200 epoch 이하에서 SOTA를 기록

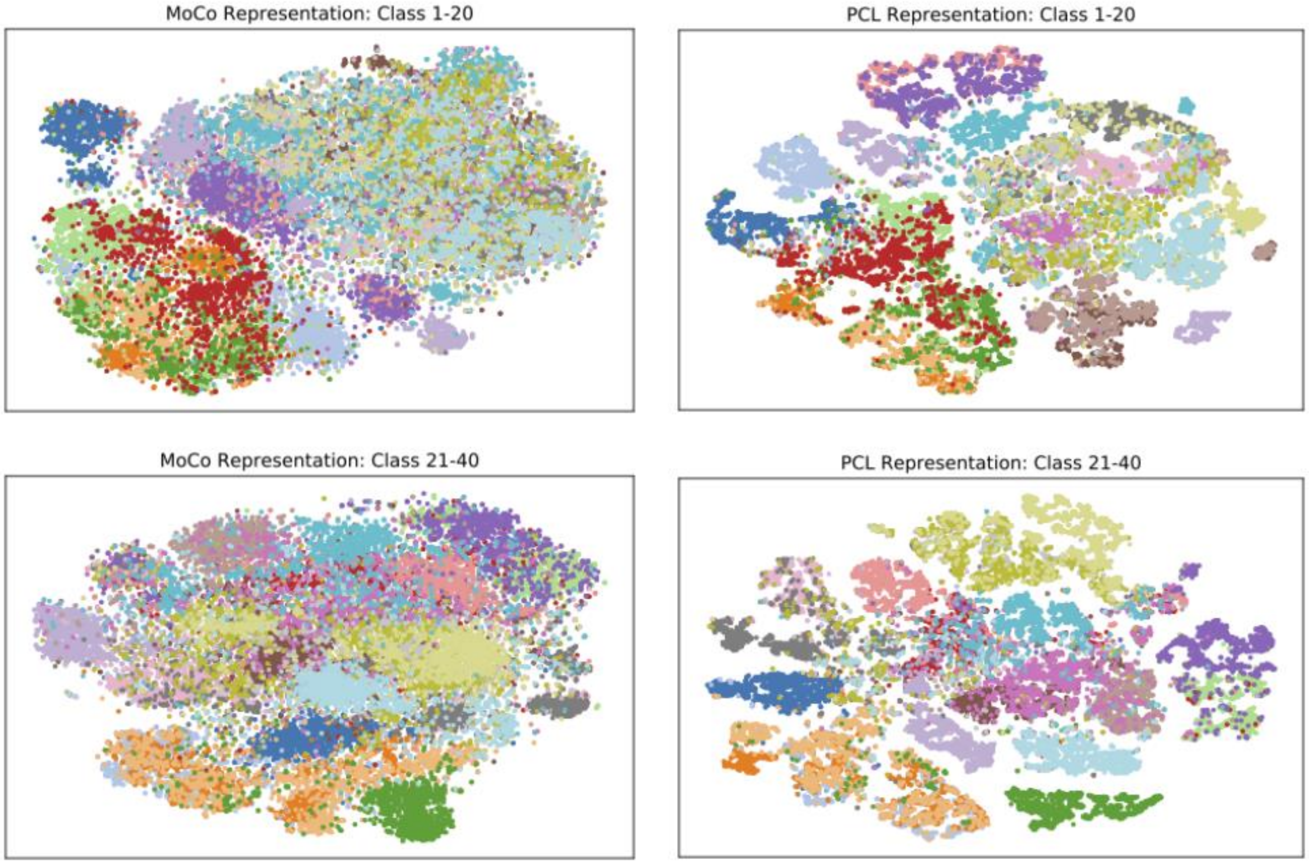


Method	architecture (#params)	#pretrain epochs	Dataset		
			ImageNet	VOC07	Places205
Jigsaw (Noroozi & Favaro, 2016)	R50 (24M)	90	45.7	64.5	41.2
Rotation (Gidaris et al., 2018)	R50 (24M)	–	48.9	63.9	41.4
DeepCluster (Caron et al., 2018)	VGG(15M)	100	48.4	71.9	37.9
BigBiGAN (Donahue & Simonyan, 2019)	R50 (24M)	–	56.6	–	–
InstDisc (Wu et al., 2018)	R50 (24M)	200	54.0	–	45.5
MoCo (He et al., 2020)	R50 (24M)	200	60.6	79.2*	48.9*
PCL (ours)	R50 (24M)	200	**61.5**	**82.3**	**49.2**
SimCLR (Chen et al., 2020a)	R50-MLP (28M)	200	61.9	–	–
MoCo v2 (Chen et al., 2020b)	R50-MLP (28M)	200	67.5	84.0*	50.1*
PCL v2 (ours)	R50-MLP (28M)	200	**67.6**	**85.4**	**50.3**
LocalAgg (Zhuang et al., 2019)	R50 (24M)	200	60.2†	–	50.1†
SelfLabel (Asano et al., 2020)	R50 (24M)	400	61.5	–	–
CPC (Oord et al., 2018)	R101 (28M)	–	48.7	–	–
CMC (Tian et al., 2019)	R50$_{L+ab}$ (47M)	280	64.0	–	–
PIRL (Misra & van der Maaten, 2020)	R50 (24M)	800	63.6	81.1	49.8
AMDIM (Bachman et al., 2019)	Custom (626M)	150	68.1†	–	55.0†
SimCLR (Chen et al., 2020a)	R50-MLP (28M)	1000	69.3†	80.5†	–
BYOL (Grill et al., 2020)	R50-MLP$_{big}$(35M)	1000	74.3†	-	–
SwAV (Caron et al., 2020)	R50-MLP (28M)	800	75.3†	88.9†	56.7†

Experiment

❖ Visualization of learned representation

- T-SNE를 사용한 learned representation 비교
- MOCO와 비교했을 때 PCL이 확실히 feature별로 구분이 잘 되어 있음



MoCo Representation: Class 1-20

PCL Representation: Class 1-20

MoCo Representation: Class 21-40

PCL Representation: Class 21-40

Conclusion

❖ Conclusion

- Clustering을 활용한 Contrastive learning 방법으로 **ProtoNCE loss** 제안
- 기존 **Contrastive learning 방법의 문제점**을 개선한 방법론
- 새로운 ProtoNCE loss function을 제안한 점에서 기여점이 있음
- 카테고리별로 샘플이 별로 없는 데이터 셋에 대해서 분류 성능이 기존의 **Self supervised learning 방법론에 비해 더 좋은 것을 확인**
- 그 뿐만 아니라 **semi-supervised learning의 방법론에 비해 성능**이 더 좋은 것도 확인 가능
  - ✓ 200 epoch보다 적을 때에는 SOTA 기록, 하지만 더 많은 epoch 일 때 SimCLR, BYOL, SWAV가 더 좋은 성능을 보여줌

Thank You