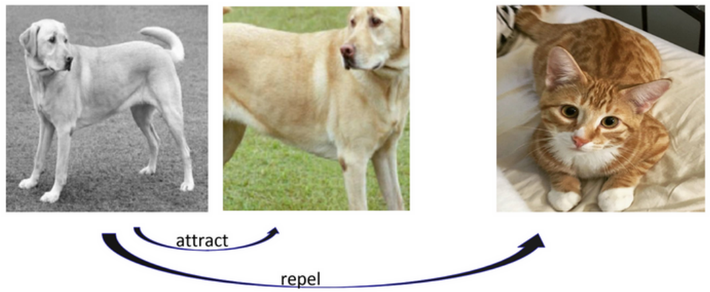
Unsupervised Visual Representation Learning by Online Constrained K-Means

CPVR 2022

발표자 임지우

Instance Discrimination



instance discrimination의 단점

- 많은 양의 데이터를 비교하기는 어렵다
- low-level 신호만을 사용해 충분히 해결이 가능하기 때문에 high-level 신호(특징)을 효과적으로 추출하도록 학습이 안될 수 있다.
- 같은 의미를 가지는 데이터(고양이1, 고양이2)라도 서로는 negative pair로 분류되기 때문에 false negative 문제가 필연적으로 발생한다.
- 큰 규모의 배치 사이즈를 요구한다

Clustering Discrimination

Red, Blue and Yellow data points before Clustering

Red, Blue and Yellow data points after Clusteing

- clustering + discrimination, Unsupervised Learning
- clustering 후 classification은 large scale data에도 학습이 가능하다.
- 비슷한 instance들끼리 끌어당기기 때문에 semantic structure를 학습

매번 전체 데이터셋을 한 batch mode에서 clustering을 하기 때문에 계산량이 많아진다는 문제

online clustering

- 일반적인 clustering 기반 방법들은 cluster 할당과 training step을 번갈아 가면서 진행하는 offline 방식
- 이러한 방법은 모든 데이터에 대해 feature를 매번 새로 추출해야 하기 때문에 target이 계속 변하는 online 학습에는 실용적이지 않음.
  -> online clustering
- mini-batch로 이전 batch에 계산한 instance를 가져와 한 batch mode에서 representation을 계산한다.

1. small subset만으로 pseudo-label(=clustering representation)을 만드는 것이 전체 distribution을 representation했다고 보기가 어렵다.

2. balancing clustering은 각 cluster마다 동일한 수의 instance가 들어가도록 배치하는 것인데 이 방법은 정확히 데이터를 clustering했다고 하기 어렵다.

SwAV VS ODC

ODC

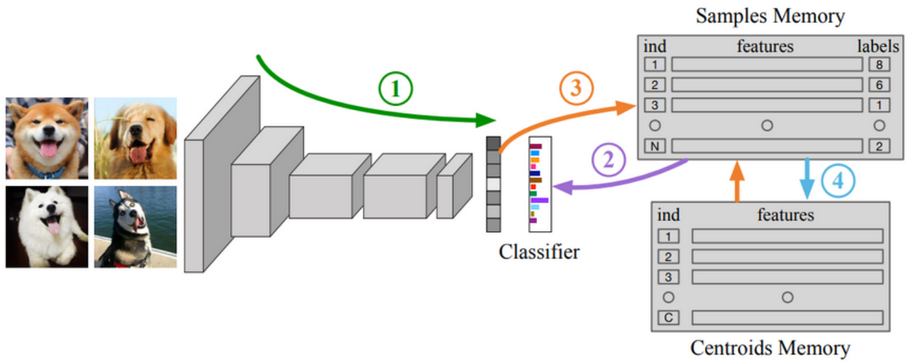


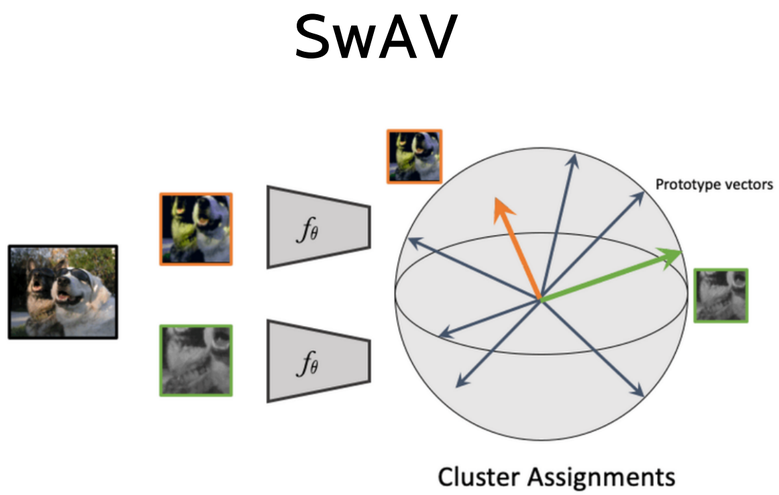
Figure 2. Each ODC iteration mainly contains four steps: 1. forward to obtain a compact feature vector; 2. read labels from the samples memory and perform back-propagation to update the CNN; 3. update samples memory by updating features and assigning new labels; 4. update centroids memory by recomputing the involved centroids.

- DeepClustering의 large batch size문제를 해결한 모델
- Samples Memory: features와 pseudo-labels를 저장
- Centroids Memory: features of class centroids 저장(= mean feature of all samples in data)
- "class": represent temporary cluster

한 batch에서 일어나는 일,
1) input image {x}가 있을 때, 신경망은 이미지를 feature vector로 나타낸다.
2) samples memory로부터 pseudo-labels를 읽어와 back-propagation을 진행한다.
4) L2 normalization을 한 f(x)를 samples memory에 features update를 진행한다. 동시에, each involved sample은 중심점을 찾아 new label이 할당된다.
5) 마지막으로, new members join하는 involved centroids와 old members가 기록된다. 그들의 centroids에 속하는 모든 sample들의 feature들을 평균해서 k번째 iteration에서 centroids momory를 update한다.

ODC의 문제점은 모든 data를 기록해놔야 한다는 것이다.

SwAV VS ODC



- 개별 이미지의 positive/negative 쌍을 직접적으로 비교하는 것이 아닌, 비슷한 feature를 가진 이미지를 군집화시켜 군집끼리 비교한다.
- 같은 이미지로부터 파생된 feature zt, zs와 k 프로토타입 c1, c2,...ck를 통해 할당한 code qt, qs가 있다.

small subset으로 계산한 pseudo-label(=clustering representation)은 전체적인 데이터를 나타낸다고 보기가 어렵다.

CoKe: online Constrained K-means

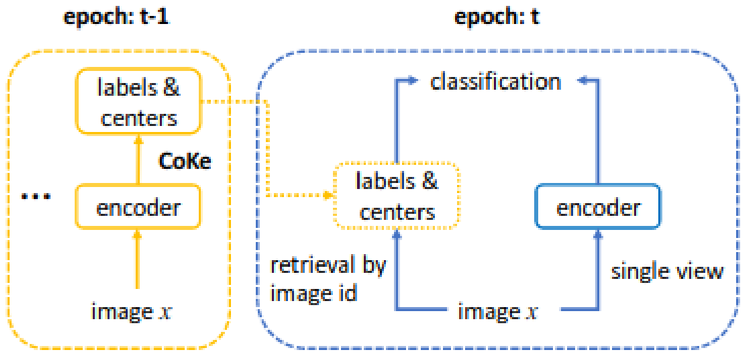
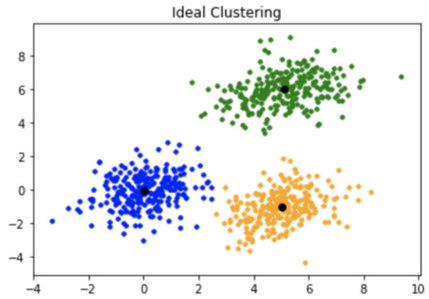


Figure 1. Illustration of CoKe. When a mini-batch arrives, each instance will be assigned to a cluster with our online assignment method. Then, in epoch t, representations from the encoder network are optimized by discrimination using pseudo labels and cluster centers obtained from epoch $t-1$. The pseudo labels from epoch $t-1$ were stored to be retrieved in epoch t using the unique id for each image.



Constrained K-Means

- cluster의 사이즈를 낮게 제한해서 사용하는 알고리즘
- balanced clustering과 달리, 내재된 data structure를 유연하게 학습이 가능하다.

mini-batch 별로 epoch t는 epoch t-1에서 얻은 pseudo labels과 cluster centers를 사용해 softmax loss계산을 통해 discrimination 과정에서 최적화한다.

- clustering에서 우세한 cluster로 많이 몰리게 되는 문제를 해결하고자 data structure를 유연하게 capture할 수 있도록 각 cluster의 사이즈를 최소한으로 제약하는 것이다.

CoKe는 each instance로부터 a single view로 representation을 학습하므로 작은 batch size에서 최적화가 가능하다.

CoKe는 각 instance로부터 a single view로 최적화할 때 좋은 성능을 보였다.

- momentum encoder, batch mode solver같은 추가적인 요소가 없는 simple framework
- moco-v2보다 성능이 좋음.

CoKe: online Constrained K-means

Two Views

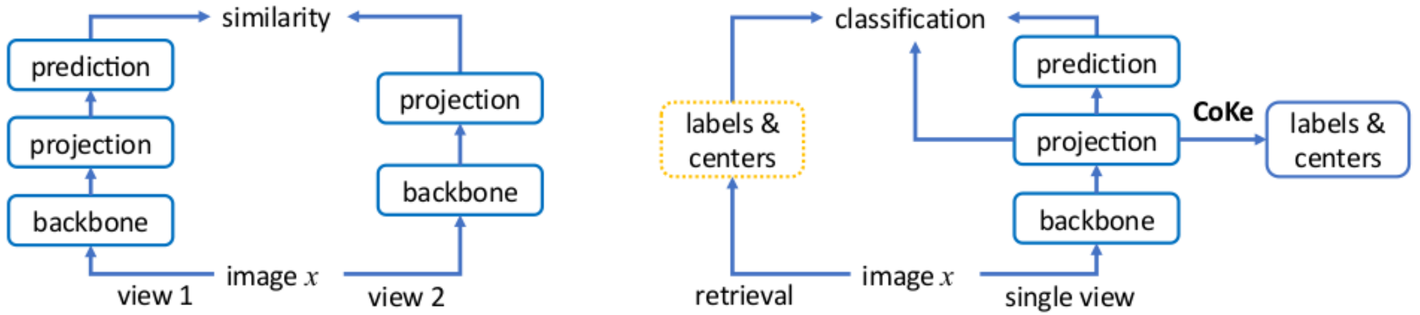
CoKe는 각 iteration에서 two views로 평가할 때, 향상된다.

Compared with the single view, multiple views can reduce the variance from different augmentations and make the assignment more stable.

Algorithm 2 Pseudo-code of CoKe with Two Views.

```
# f: encoder network for input images
# u: pseudo one-hot labels (Nx1)
# C: cluster centers
# rho: dual variable for constraints (Kx1)
# gamma: lower-bound of cluster size
# lambda: temperature
# alpha: ratio between labels

for z in loader: # load a minibatch with b samples
   z_1, z_2 = aug(z), aug(z) # two random views from z
   x_1, x_2 = f(z_1), f(z_2) # encoder representations
   s_1, s_2 = x_1C, x_2C # logits over centers
   y = u(z_id) # retrieve label from last epoch
   # compute reference distribution for each view
   p_1 = softmax(s_1/lambda)
   p_2 = softmax(s_2/lambda)
   # obtain soft label for discrimination
   y_1 = alpha*y + (1-alpha)*p_2
   y_2 = alpha*y + (1-alpha)*p_1
   # loss over two views
   loss = 0.5*(-y_1*log(p_1) -y_2*log(p_2))
   loss.backward() # update encoder
   # update clustering
   x_mean = 0.5*(x_1+x_2) # mean vector of two views
   u(z_id) = update(x_mean, C, rho) # as in Eqn. 13
   C = update(C, x_mean, u(z_id)) # as in Eqn. 11
   rho = update(rho, gamma, u(z_id)) # as in Eqn. 14
```



(a) Existing methods with prediction

(b) CoKe with prediction

Figure 2. Illustration of architecture with the additional prediction head. The yellow bounding box denotes the results from the last epoch.

Ratio: γ'	Acc%	#Cons	#Min	#Max
1	63.1	427	403	445
0.8	63.8	342	338	1,301
0.6	64.3	256	254	1,404
0.4	64.5	171	168	2,371
0	41.3	0	0	$449k$

Table 1. Comparison of different ratios γ' in CoKe. The performance is evaluated by linear classification with learned representations on ImageNet as in MoCo [17].

- r' = 1: balanced clustering 63.1% < r' = 0.4: 정확도는 1.4% 증가

#max: 가장 큰 cluster의 instance개수
#min: 가장 작은 cluster의 instance개수
#Cons: constrained cluster size

- balanced clustering과 비교해본 결과

#max는 5배정도 크고 #min은 약 3배 작아졌다.

K	Acc%	#Cons	#Min	#Max
1,000	63.4	512	512	4,639
3,000	64.5	171	168	2,371
5,000	64.3	102	98	1,982

Table 3. Comparison of number of clusters K in k-means.

k = 3000, Acc 64.5%

$K(\times 1,000)$	3	2+3	3+4	3+3+3	3+4+5
Acc%	64.5	65.0	65.2	65.2	65.3

Table 4. Multi-clustering with different K combinations.

Multi-clustering: k=3000으로 계속 clustering을 할 때보다 k의 개수를 늘려서 진행했을 때 더 정확도가 높아짐

Settings	$\{C^{t-1}, \tilde{y}^{t-1}\}$	$\{C^{t-1}, \tilde{y}^{t}\}$	$\{C^{t}, \tilde{y}^{t-1}\}$	$\{C^{t}, \tilde{y}^{t}\}$
Acc%	64.5	0.4	51.2	0.1

Table 2. Comparison of labels and centers from different epochs.

CoKe는 clustering과 discrimination을 분리한다.
지난 epoch에서 얻은 결과를 clustering해서 다음 epoch에서 data를 구분한다. 위에 보이는 Table 2는 다른 labels과 centers들을 비교한다.

MoCo-v2	SwAV	CoKe	CoKe*
18.3	20.8	11.1	8.4

Methods	#View	#Epoch	#Dim	Acc%
SimCLR	2	1,000	128	69.3
MoCo-v2	2	800	128	71.1
DeepCluster-v2	2	400	128	70.2
SwAV	2	400	128	70.1
CoKe	1	800	128	**71.4**

- ResNet - 50을 backbone으로 사용함.
- 기존 방식들은 augmentation을 진행한 two views로 각각의 iteration에서 representation을 출력했다.
- CoKe는 online optimization에서 a single view를 사용한다.
- MoCo-V2와는 달리 view를 하나만 사용해서 최적화했다는 것에 의의가 있다.

Methods	#V	Bs	#D	ME	MB	Acc%
SimSiam [9]	2	256	2,048	✓		71.3
SwAV [9]	2	4,096	128			71.8
MoCo-v2+ [9]	2	256	128	✓	✓	72.2
Barlow Twins [40]	2	2,048	8,192			73.2
MoCo-v3 [10]	2	4,096	256	✓		73.8
BYOL [16]	2	4,096	256	✓		74.3
NNCLR [13]	2	1,024	256	✓	✓	72.9
NNCLR [13]	2	4,096	256	✓	✓	75.4
DeepCluster-v2 [4]	8	4,096	128			75.2
SwAV [4]	8	4,096	128			75.3
DINO [5]	8	4,096	256	✓		75.3
NNCLR [13]	8	4,096	256	✓	✓	75.6
CoKe	1	1,024	128			72.5
CoKe	2	1,024	128			74.9
CoKe	8	1,024	128			**76.4**

ME: Memory encoder, MB: Memory bank

- NNCLR은 large memory bank를 사용하지만 CoKe는 online방식의 clustering을 사용했는데도 좋은 성능을 보였다는 것에 의미가 있다.
- Multi-Crop을 사용한 결과 CoKe가 가장 좋은 성능을 보임.(#V: GPU 개수)