ELLab

SUNG KYUN KWAN UNIVERSITY

BYOL & SimSiam

2026.02.06

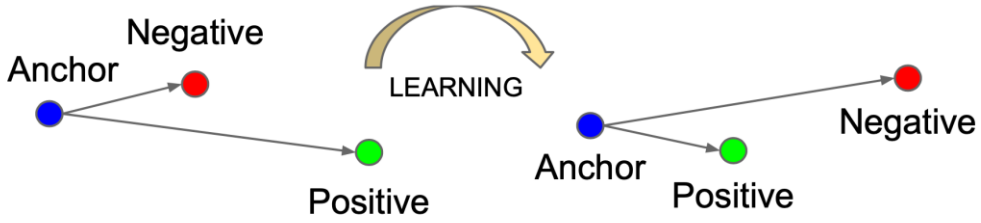
Introduction

Contrastive learning

한 배치에서 anchor x에 대해

- positive: 같은 이미지의 다른 view x^+
- negative: 다른 이미지들의 view들 $x_1^-, x_2^-, \ldots$

목표는 $f(x)$와 $f(x^+)$는 가깝게, $f(x)$와 $f(x^-)$는 멀게함



Negative가 없다면

목표는 "같은 이미지의 두 view는 같게 만들어라"

Collapse 발생→loss 최소

- 입력 이미지가 무엇이든 encoder 출력이 항상 같은 상수 벡터가 되는 현상
- Loss는 좋은데 representation에 정보가 없음, feature가 무의미해짐

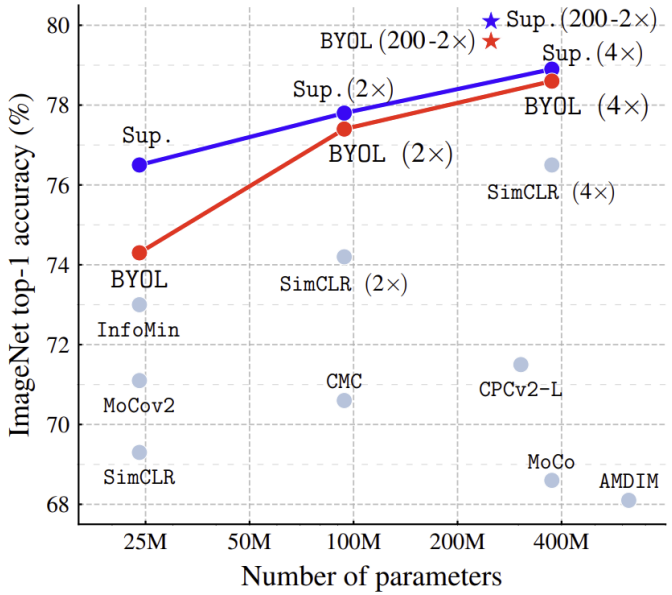
Introduction

Idea

- SimCLR, MoCo 등 기존 Contrastive learning은 많은 negative가 필요함
- 따라서 batch size를 늘리거나 queue를 사용하는 등 부가적인 방법이 필요했음
- Negative 없이도 Collapse 피할 수 있다면?
  - Online/target 비대칭
  - Use of a slow-moving average

"Negative pairs 없이도 self-supervised representation learning을 가능하게 함"

- 성능 동등하거나 우세함
- Batch size와 augmentation에 더 robust 함



Methodology

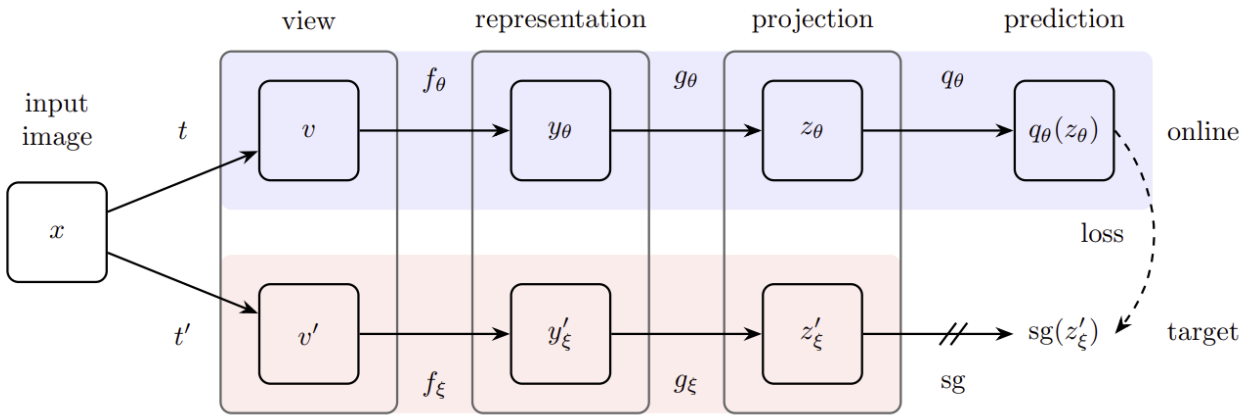
Architecture

Online Network(학습 대상)

- Encoder f_θ
- Projector g_θ
- Predictor q_θ

Target Network

- Encoder f_ξ
- Projector g_ξ
- Stop gradient



1. Target representation을 정의
2. Online network가 이를 예측하도록 학습
3. Online network를 새로운 target으로 사용
4. 이 과정을 반복하면 representation이 점진적으로 개선됨

Methodology

Loss function

- Online prediction과 target projection을 맞추는 regression loss
- L2-normalized MSE (cosine similarity 기반)

$$\mathcal{L}_{\theta,\xi} \triangleq \left\| \overline{q_\theta}(z_\theta) - \overline{z}'_\xi \right\|_2^2 = 2 - 2 \cdot \frac{\langle q_\theta(z_\theta), z'_\xi \rangle}{\left\| q_\theta(z_\theta) \right\|_2 \cdot \left\| z'_\xi \right\|_2}.$$

- Symmetric loss: view $v \leftrightarrow v'$를 바꿔 한 번 더 계산
- 최종 loss $\mathcal{L}_{\theta,\xi}^{\texttt{BYOL}} = \mathcal{L}_{\theta,\xi} + \widetilde{\mathcal{L}}_{\theta,\xi}$

Optimization Dynamics

- Online parameters θ: gradient descent로 업데이트
- Target parameters ξ: EMA 업데이트
- Online은 target을 따라가며 학습하며 Target은 online의 느린 평균임

$$\theta \leftarrow \text{optimizer}\left(\theta, \nabla_\theta \mathcal{L}_{\theta,\xi}^{\texttt{BYOL}}, \eta\right),$$
$$\xi \leftarrow \tau\xi + (1-\tau)\theta,$$

Methodology

Intuitions

Jointly minimize gradient descent

- 두 파라미터: $\theta \leftarrow \theta - \eta \nabla_\theta L(\theta, \xi)$
  $\xi \leftarrow \xi - \eta \nabla_\xi L(\theta, \xi)$
- Loss 함수: $L(\theta, \xi) = \mathbb{E}\left[\|f_\theta(x) - g_\xi(x)\|^2\right]$

- Collapsed Solution: $f_\theta(x) = c, \quad g_\xi(x) = c \quad \forall x$
- Loss 관점에서 완벽한 해

BYOL

$$\theta \leftarrow \text{optimizer}\left(\theta, \nabla_\theta \mathcal{L}^{\text{BYOL}}_{\theta,\xi}, \eta\right),$$
$$\xi \leftarrow \tau\xi + (1-\tau)\theta,$$

- Joint gd 아님
- 따라서 collapse가 loss의 critical point라고 해도 BYOL의 업데이트 규칙이 그 방향을 향한다는 보장 없음

Methodology

Intuitions

$$q^\star \triangleq \arg\min_q \mathbb{E}\left[\left\|q(z_\theta) - z'_\xi\right\|_2^2\right], \quad \text{where} \quad q^\star(z_\theta) = \mathbb{E}\left[z'_\xi | z_\theta\right],$$

$$\nabla_\theta \mathbb{E}\left[\left\|q^\star(z_\theta) - z'_\xi\right\|_2^2\right] = \nabla_\theta \mathbb{E}\left[\left\|\mathbb{E}\left[z'_\xi | z_\theta\right] - z'_\xi\right\|_2^2\right] = \nabla_\theta \mathbb{E}\left[\sum_i \mathrm{Var}(z'_{\xi,i} | z_\theta)\right],$$

Variance aspect

- BYOL loss는 MSE regression임
- Optimal predictor 가정 ⇒ mse 최소화하는 최적 함수는 조건부 평균
- 즉, Loss는 조건부 분산을 최소화하는 것과 같음
  - online representation을 알고 있을 때 target representation의 불확실성을 줄이는 방향으로 학습함
- Collapse 즉, z_θ가 constant이면 분산 최대

  $\mathrm{Var}(z'_\xi \mid z_\theta) = \mathrm{Var}(z'_\xi \mid c) = \mathrm{Var}(z'_\xi)$
- 반대로 z_θ가 이미지 의미를 잘 담고 있으면

  $\mathrm{Var}(z'_\xi \mid z_\theta) \ll \mathrm{Var}(z'_\xi)$

"BYOL의 θ업데이트는 정보를 버리는 방향으로 움직일 수 없으므로 collapse 되지 않음"

Experiment

Linear evaluation on ImageNet

- ImageNet self-supervised pretraining
- Encoder freeze + linear classifier 학습
- 기존 contrastive SOTA(SimCLR, MoCo 등)보다 높거나 동등
- Negative pairs 없이 SOTA 달성

→ BYOL은 contrastive learning 없이도 매우 강한 representation을 학습함

Method	Top-1	Top-5
Local Agg.	60.2	-
PIRL [35]	63.6	-
CPC v2 [32]	63.8	85.3
CMC [11]	66.2	87.0
SimCLR [8]	69.3	89.0
MoCo v2 [37]	71.1	-
InfoMin Aug. [12]	73.0	91.1
BYOL (ours)	**74.3**	**91.6**

(a) ResNet-50 encoder.

Method	Architecture	Param.	Top-1	Top-5
SimCLR [8]	ResNet-50 (2×)	94M	74.2	92.0
CMC [11]	ResNet-50 (2×)	94M	70.6	89.7
BYOL (ours)	ResNet-50 (2×)	94M	77.4	93.6
CPC v2 [32]	ResNet-161	305M	71.5	90.1
MoCo [9]	ResNet-50 (4×)	375M	68.6	-
SimCLR [8]	ResNet-50 (4×)	375M	76.5	93.2
BYOL (ours)	ResNet-50 (4×)	375M	78.6	94.2
BYOL (ours)	ResNet-200 (2×)	250M	**79.6**	**94.8**

(b) Other ResNet encoder architectures.

Semi-Supervised Learning

- ImageNet 라벨의 일부만 사용 (1%, 10%)
- Pretrained encoder fine-tuning
- 모든 label fraction에서 BYOL이 기존 방법들보다 우수

→ BYOL representation은 소량의 라벨만으로도 잘 일반화됨

Method	Top-1		Top-5	
	1%	10%	1%	10%
Supervised [77]	25.4	56.4	48.4	80.4
InstDisc	-	-	39.2	77.4
PIRL [35]	-	-	57.2	83.8
SimCLR [8]	48.3	65.6	75.5	87.8
BYOL (ours)	**53.2**	**68.8**	**78.4**	**89.0**

Method	Architecture	Param.	Top-1		Top-5	
			1%	10%	1%	10%
CPC v2 [32]	ResNet-161	305M	-	-	77.9	91.2
SimCLR [8]	ResNet-50 (2×)	94M	58.5	71.7	83.0	91.2
BYOL (ours)	ResNet-50 (2×)	94M	62.2	73.5	84.1	91.7
SimCLR [8]	ResNet-50 (4×)	375M	63.0	74.4	85.8	92.6
BYOL (ours)	ResNet-50 (4×)	375M	69.1	75.7	87.9	92.5
BYOL (ours)	ResNet-200 (2×)	250M	**71.2**	**77.7**	**89.5**	**93.7**

Experiment

Transfer learning

Method	Food101	CIFAR10	CIFAR100	Birdsnap	SUN397	Cars	Aircraft	VOC2007	DTD	Pets	Caltech-101	Flowers
Linear evaluation:												
BYOL (ours)	**75.3**	91.3	**78.4**	**57.2**	**62.2**	**67.8**	60.6	82.5	75.5	90.4	94.2	**96.1**
SimCLR (repro)	72.8	90.5	74.4	42.4	60.6	49.3	49.8	81.4	**75.7**	84.6	89.3	92.6
SimCLR [8]	68.4	90.6	71.6	37.4	58.8	50.3	50.3	80.5	74.5	83.6	90.3	91.2
Supervised-IN [8]	72.3	**93.6**	78.3	53.7	61.9	66.7	**61.0**	**82.8**	74.9	**91.5**	**94.5**	94.7
Fine-tuned:												
BYOL (ours)	**88.5**	**97.8**	86.1	**76.3**	63.7	91.6	**88.1**	**85.4**	**76.2**	91.7	**93.8**	97.0
SimCLR (repro)	87.5	97.4	85.3	75.0	63.9	91.4	87.6	84.5	75.4	89.4	91.7	96.6
SimCLR [8]	88.2	97.7	85.9	75.9	63.5	91.3	88.1	84.1	73.2	89.2	92.1	97.0
Supervised-IN [8]	88.3	97.5	**86.4**	75.8	**64.3**	**92.1**	86.0	85.0	74.6	**92.1**	93.3	**97.6**
Random init [8]	86.9	95.9	80.2	76.1	53.6	91.4	85.9	67.3	64.8	81.5	72.6	92.0

Classification

- Linear evaluation & fine-tuning 모두 우수
- 대부분의 benchmark에서 SimCLR 및 supervised baseline 능가

Other vision task

- 모든 task에서 supervised / SimCLR 대비 동등 이상 성능

→ BYOL은 특정 task에 특화되지 않은 general-purpose representation을 학습함

Method	AP_{50}	mIoU
Supervised-IN [9]	74.4	74.4
MoCo [9]	74.9	72.5
SimCLR (repro)	75.2	75.2
BYOL (ours)	**77.5**	**76.3**

(a) Transfer results in semantic segmentation and object detection.

Method	Higher better			Lower better	
	pct.<1.25	pct.$<1.25^2$	pct.$<1.25^3$	rms	rel
Supervised-IN [83]	81.1	95.3	98.8	0.573	**0.127**
SimCLR (repro)	83.3	96.5	99.1	0.557	0.134
BYOL (ours)	**84.6**	**96.7**	**99.1**	**0.541**	0.129

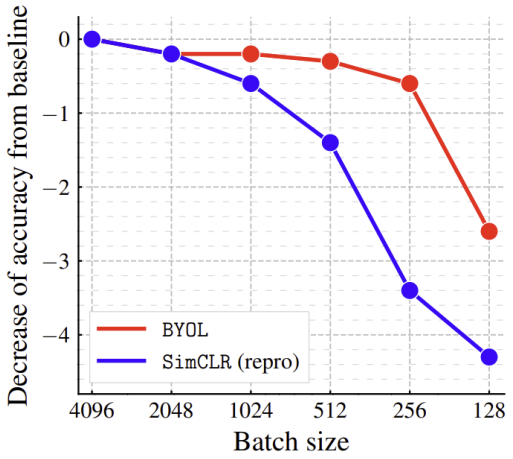
(b) Transfer results on NYU v2 depth estimation.

Ablation

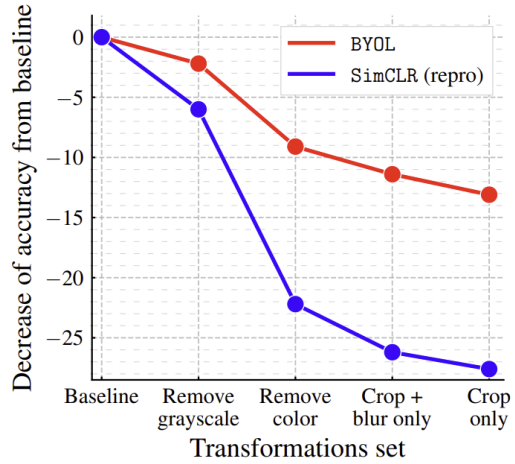
Batch Size

- Contrastive learning은 batch 내 negative samples에 의존
- Batch size ↓ → negative 수 ↓ → 성능 급락
- BYOL은 Batch size 256 ~ 4096 구간에서 성능 거의 유지

→ BYOL은 negative examples에 의존하지 않기 때문에 batch size 변화에 훨씬 더 robust하다.



(a) Impact of batch size



(b) Impact of progressively removing transformations

Image Augmentation

- Augmentation을 단계적으로 제거함
- SimCLR: Color distortion 제거 → 성능 대폭 하락
  - color 정보만으로 contrastive task 해결 가능 (shortcut)
- BYOL: SimCLR과 달리 Crop only에서도 상대적으로 높은 성능 유지함

→ BYOL은 augmentation에 덜 민감함

Ablation

Bootstrapping

$$\xi \leftarrow \tau\xi + (1-\tau)\theta,$$

τ = 1 → target 고정

- target은 초기값 그대로
- iterative improvement 불가능

→ 학습은 안정적이지만 representation이 개선되지 않음

τ = 0 → target = online (즉시 copy)

- 예측 대상이 자기 자신
- Optimization이 불안정함

→ 훈련 붕괴 + 매우 낮은 성능

Target	τ_{base}	Top-1
Constant random network	1	$18.8_{\pm 0.7}$
Moving average of online	0.999	69.8
Moving average of online	0.99	**72.5**
Moving average of online	0.9	68.4
Stop gradient of online†	0	0.3

(a) Results for different target modes. †In the *stop gradient of online*, $\tau = \tau_{\text{base}} = 0$ is kept constant throughout training.

τ = 0.9 ~ 0.999

- Target은 online의 과거 평균이 되고 online은 조금 더 나아진 표현을 만듦

→ 이 과정을 반복하면서 점진적으로 자기 개선 루프가 성립됨

Ablation

contrastive methods

$$\text{InfoNCE}_\theta^{\alpha,\beta} \triangleq \frac{2}{B}\sum_{i=1}^{B} S_\theta(v_i, v_i') - \beta \cdot \frac{2\alpha}{B}\sum_{i=1}^{B} \ln\left(\sum_{j\neq i} \exp\frac{S_\theta(v_i, v_j)}{\alpha} + \sum_{j} \exp\frac{S_\theta(v_i, v_j')}{\alpha}\right),$$

Positive pair

Negative pairs

SimCLR

- (no predictor)
- $\psi(u) = z\theta(u)$ (no target network)

BYOL

- $\beta = 0$

Method	Predictor	Target network	β	Top-1
BYOL	✓	✓	0	**72.5**
–	✓	✓	1	70.9
–		✓	1	70.7
SimCLR			1	69.4
–	✓		1	69.1
–	✓		0	0.3
–		✓	0	0.2
–			0	0.1

(b) Intermediate variants between BYOL and SimCLR.

negative examples 없이 (β = 0) 잘 되는 유일한 방법은"target network + predictor"를 동시에 쓰는 BYOL뿐임

SimCLR에 predictor 연결했을때 성능 영향 미미하지만 target network 붙였을때 성능 향상 있었음(+1.6%)

"negative pair 없을때 target network & predictor 둘 다 필요함 "

Conclusion

BYOL

- BYOL은 negative pairs 없이 representation을 예측하는 bootstrap 기반 self-supervised 방법
- ImageNet linear evaluation에서 SOTA 성능 달성
- Contrastive methods 대비 batch size, image augmentation 선택에 더 강건

- 한계점
  - 여전히 도메인별 augmentation 설계에 의존
  - audio / video / text 등 다른 modality로 확장하기 위해 augmentation 자동 탐색이 중요한 연구 과제

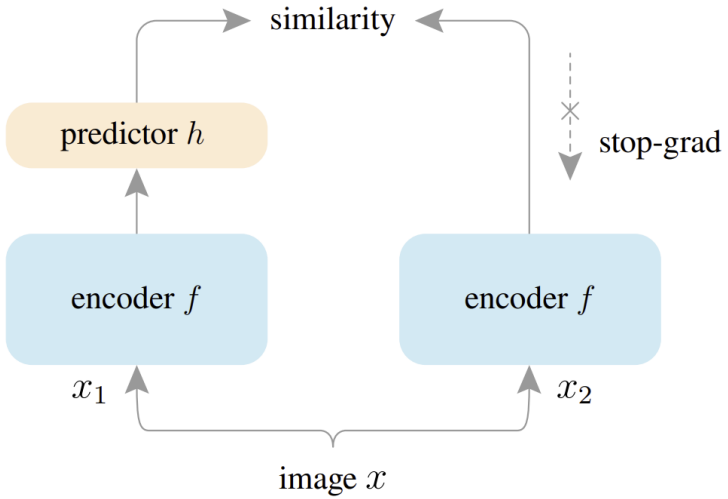
SimSiam

ELLab

Introduction

Siamese network

- 한 이미지 → 서로 다른 두 개의 augmentation
- 두 뷰의 representation을 비슷하게 만들도록 학습
- 문제: 아무 제약 없으면 collapse (모두 같은 벡터) 발생



- SimCLR: negative pairs로 서로 다른 이미지 밀어냄
- MoCo: negative queue + momentum encoder
- BYOL: negative 없이 가능하지만 momentum encoder 필수라고 알려짐

“Siamese 구조 + stop-gradient” 만으로도 의미 있는 표현 학습이 가능함

모멘텀 인코더, negative, 클러스터링이 필수는 아님

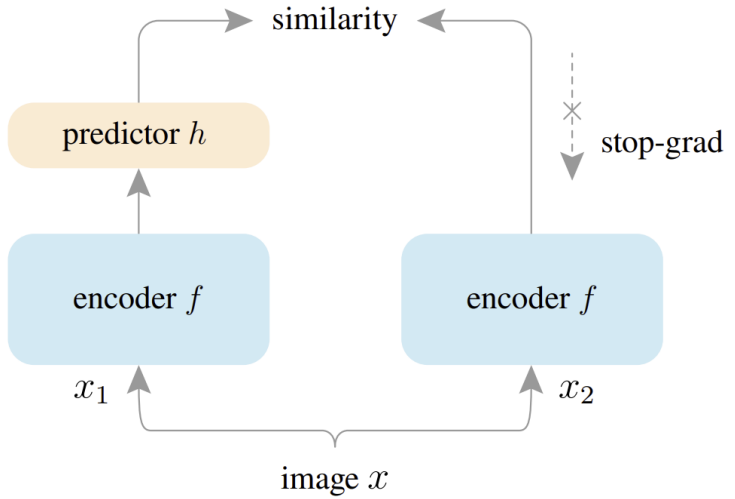
Methodology

Siamese network

Algorithm 1 SimSiam Pseudocode, PyTorch-like

```
# f: backbone + projection mlp
# h: prediction mlp

for x in loader: # load a minibatch x with n samples
   x1, x2 = aug(x), aug(x) # random augmentation
   z1, z2 = f(x1), f(x2) # projections, n-by-d
   p1, p2 = h(z1), h(z2) # predictions, n-by-d

   L = D(p1, z2)/2 + D(p2, z1)/2 # loss

   L.backward() # back-propagate
   update(f, h) # SGD update

def D(p, z): # negative cosine similarity
   z = z.detach() # stop gradient

   p = normalize(p, dim=1) # l2-normalize
   z = normalize(z, dim=1) # l2-normalize
   return -(p*z).sum(dim=1).mean()
```

- Encoder f
  - ResNet & MLP
- Prediction MLP h

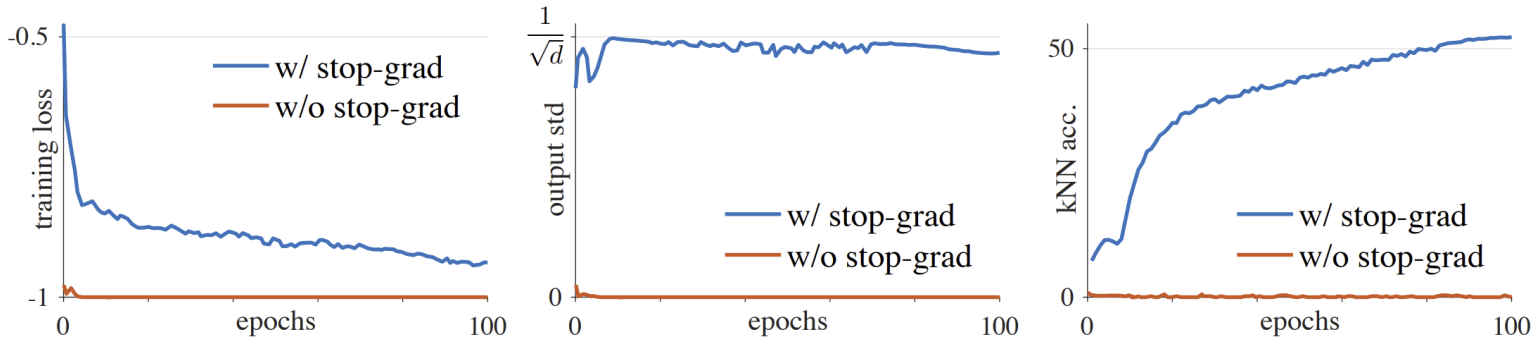


- Stop-gradient
  - p -> h → f 쪽만 업데이트, z쪽 encoder로는 흐르지 않음
- Symmetrized loss
  - 두 view가 번갈아 가면서 online/target 역할 맡음

$$\mathcal{L}=\frac{1}{2}\mathcal{D}(p_1,\texttt{stopgrad}(z_2))+\frac{1}{2}\mathcal{D}(p_2,\texttt{stopgrad}(z_1)).$$

Methodology

Stop-gradient



with stop gradient

- Loss 점차 작아짐 (left)
- Std of $z/\| z \|_2 \approx 1/\sqrt{d}$ → 출력이 고르게 분포함 (middle)
- kNN 성능 accuracy가 꾸준히 상승함 (right)

Without stop gradient

- loss가 빠르게 -1로 수렴함 (left)
- Std of $z/\| z \|_2 = 0$ → 출력이 상수 벡터로 collapse (middle)
- 성능 붕괴 (right)

"SimSiam에서 collapsing solution이 실제로 존재하고 stop gradient가 이것을 막는데 필수적임 "

Methodology

Stop-gradient

- Predictor, Batch Size, Batch Normalization, Similarity function, Symmetrization에 대한 다양한 실험 진행
- 몇가지 요소를 빼거나 바꿨을 때 학습에 실패하거나 성능이 안 좋아지는 결과가 나왔지만
- 이는 collapse 때문이 아닌 것으로 밝혀짐

“The optimizer (batch size), batch normalization, similarity function, and symmetrization may affect accuracy, but we have seen no evidence that they are related to collapse prevention. It is mainly the stop-gradient operation that plays an essential role.”

Experiment

ImageNet Linear Evaluation

method	batch size	negative pairs	momentum encoder	100 ep	200 ep	400 ep	800 ep
SimCLR (repro.+)	4096	✓		66.5	68.3	69.8	70.4
MoCo v2 (repro.+)	**256**	✓	✓	67.4	69.9	71.0	72.2
BYOL (repro.)	4096		✓	66.5	**70.6**	**73.2**	**74.3**
SwAV (repro.+)	4096			66.5	69.1	70.7	71.8
SimSiam	**256**			**68.1**	70.0	70.8	71.3

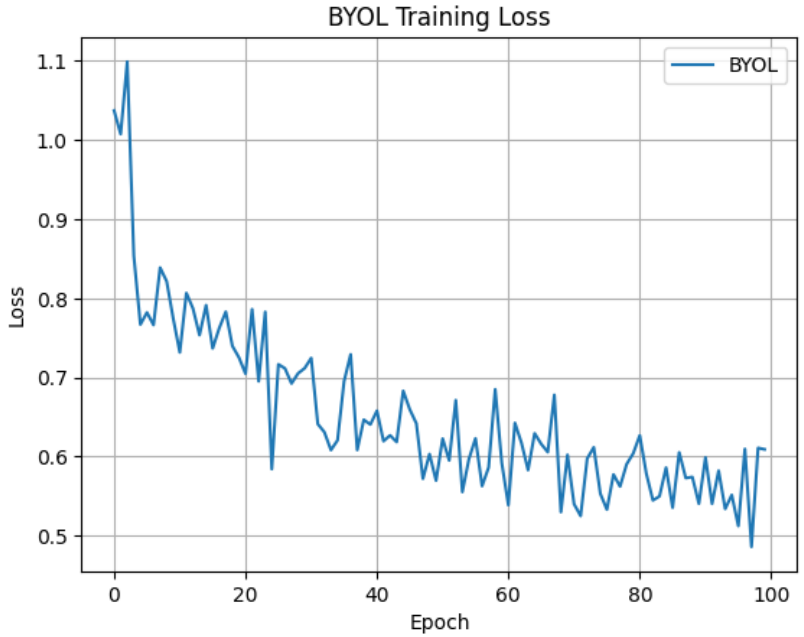
- Negative & momentum 사용안함 / Batch size 256
- 그럼에도 100 epoch에서는 전체 최고 성능
- 하지만 epoch을 길게 늘릴수록 성능 증가 폭은 작고 장기 학습에서는 BYOL이 더 강함

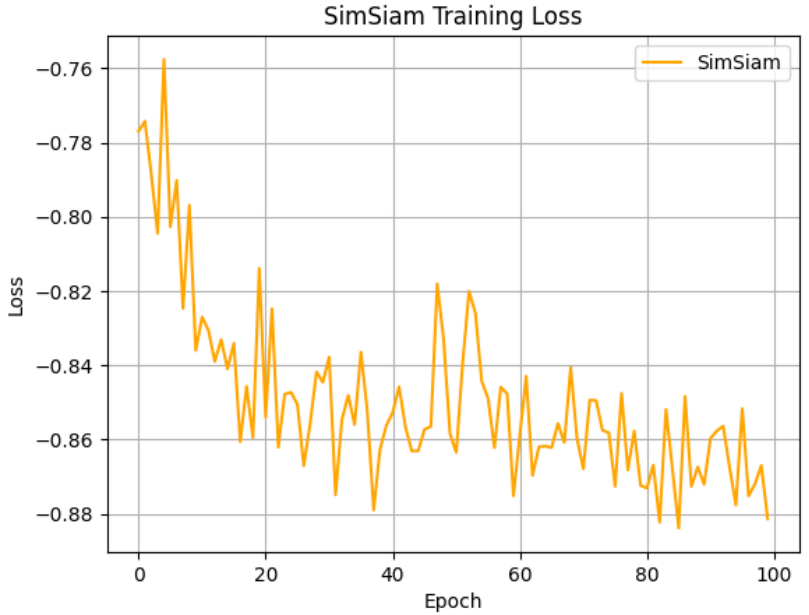
	VOC 07 detection			VOC 07+12 detection			COCO detection			COCO instance seg.		
pre-train	AP_{50}	AP	AP_{75}	AP_{50}	AP	AP_{75}	AP_{50}	AP	AP_{75}	AP^{mask}_{50}	AP^{mask}	AP^{mask}_{75}
scratch	35.9	16.8	13.0	60.2	33.8	33.1	44.0	26.4	27.8	46.9	29.3	30.8
ImageNet supervised	74.4	42.4	42.7	81.3	53.5	58.8	58.2	38.2	41.2	54.7	33.3	35.2
SimCLR (repro.+)	75.9	46.8	50.1	81.8	55.5	61.4	57.7	37.9	40.9	54.6	33.3	35.3
MoCo v2 (repro.+)	**77.1**	**48.5**	**52.5**	**82.3**	**57.0**	**63.3**	**58.8**	**39.2**	**42.5**	**55.5**	**34.3**	**36.6**
BYOL (repro.)	**77.1**	47.0	49.9	81.4	55.3	61.1	57.8	37.9	40.9	54.3	33.2	35.0
SwAV (repro.+)	75.5	46.5	49.6	81.5	55.4	61.4	57.6	37.6	40.3	54.2	33.1	35.1
SimSiam, base	75.5	47.0	50.2	**82.0**	56.4	62.8	57.5	37.9	40.9	54.2	33.2	35.2
SimSiam, optimal	**77.3**	**48.5**	**52.5**	**82.4**	**57.0**	**63.7**	**59.3**	**39.2**	**42.1**	**56.0**	**34.4**	**36.7**

- SimSiam representations는 ImageNet을 넘어 transfer 가능
- SimCLR / MoCo / BYOL / SwAV와 비슷하거나 경쟁적 성능

My Code

Experiment





구분	Accuracy
MoCo v2	67.55%
SimCLR	76.51%
BYOL	76.31%
SimSiam	64.11%

https://colab.research.google.com/drive/17TaY_y8PGE9-mRDuvzcNvbHf8Kxh_r-B?usp=sharing

https://colab.research.google.com/drive/14iH4R8WSPGNvtsOsM7PKn6XEXRDvayDk?usp=sharing

ELLab