

SELF SUPERVISED LEARNING

Self-supervised Learning

Unsupervised Pre-train, Supervised Fine-tune.

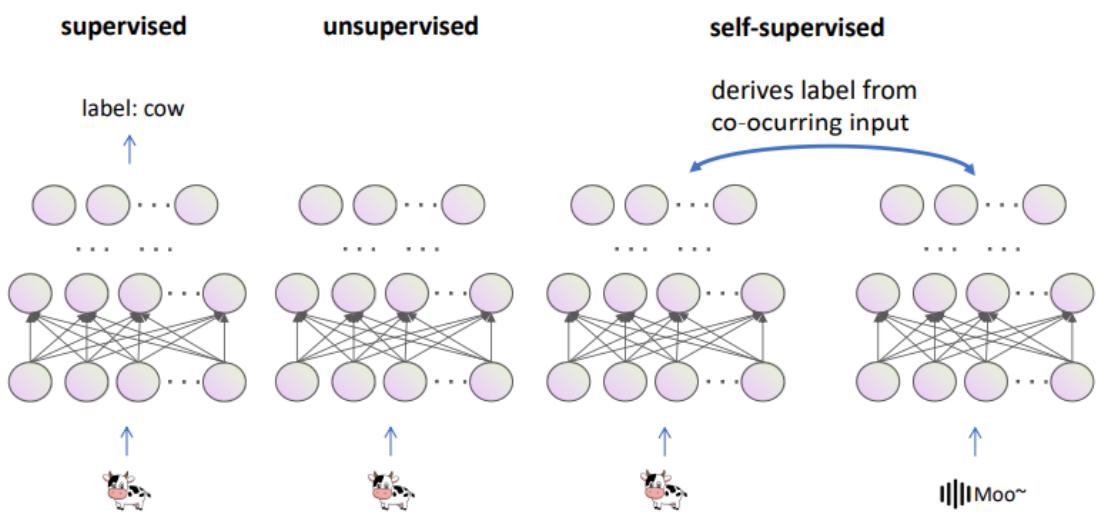


Fig. 3: The differences among supervised learning, unsupervised learning, and SSL. The image is reproduced from [32]. SSL utilizes freely derived labels as supervision instead of manually annotated labels.

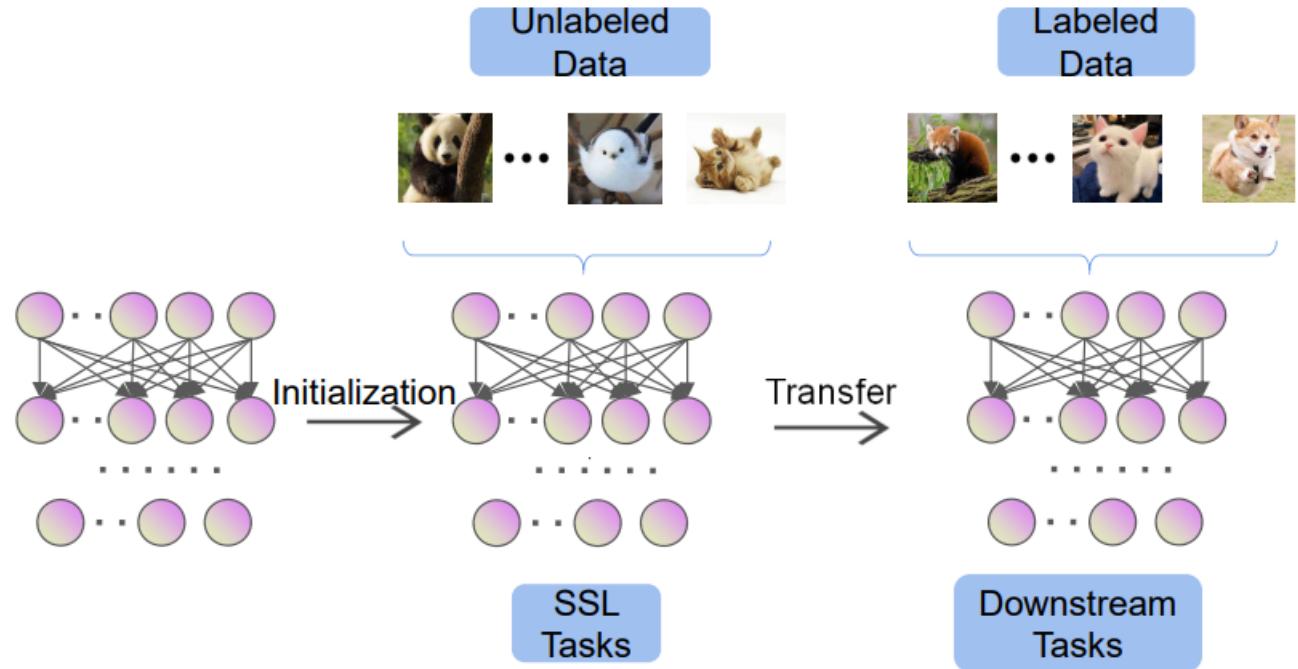


Fig. 1: The general pipeline of applying SSL methods to downstream tasks. The SSL models are first pre-trained on the unlabeled data and then fine-tuned, or directly evaluated, on the labeled data of the downstream tasks.

Self supervised Learning

SSL is a subset of unsupervised learning

Feature	Self-Supervised Learning	Unsupervised Learning
Data Utilization	Generates supervisory signals from unlabeled data	Operates entirely without labels
Learning Objective	Learns representations by solving specific pretext tasks	Discovers patterns and structures in the data
Feedback Mechanism	Involves feedback through pseudo-labels derived from data	Lacks feedback loops; focuses on overall data analysis
Common Techniques	Predicting missing parts, contrastive learning	Clustering, dimensionality reduction

指使用大量未標記的資料，藉由資料本身的特性找出潛在的規則

How Much Information is the Machine Given during Learning?

► “Pure” Reinforcement Learning (**cherry**)

- The machine predicts a scalar reward given once in a while.

► **A few bits for some samples**



► Supervised Learning (**icing**)

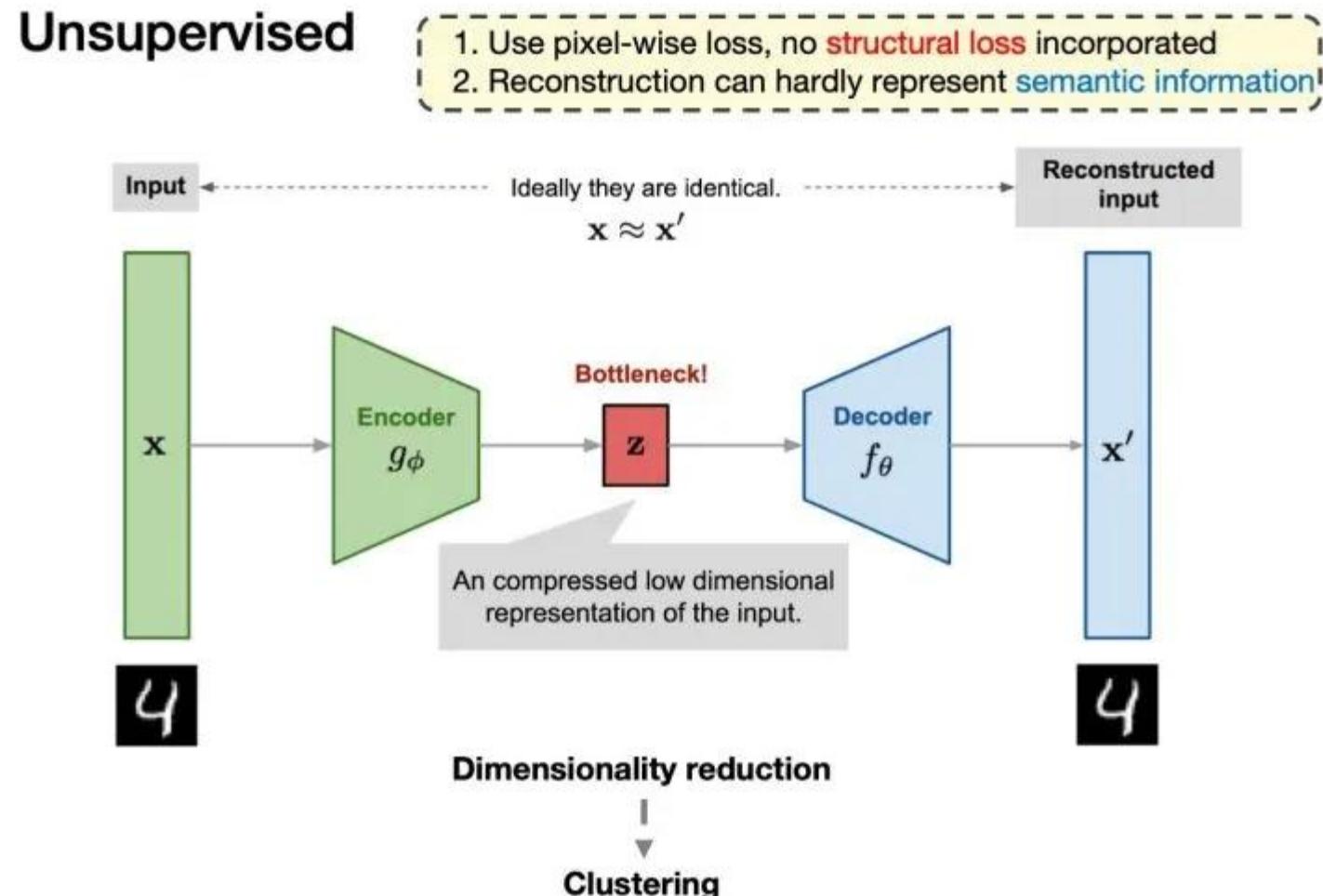
- The machine predicts a category or a few numbers for each input
- Predicting human-supplied data
- **10→10,000 bits per sample**

► Self-Supervised Learning (**cake génoise**)

- The machine predicts any part of its input for any observed part.
- Predicts future frames in videos
- **Millions of bits per sample**

Unsupervised Learning with AutoEncoder

- Map input sample to latent space by encoder
- Reconstruct input by decoder



	Low-Level Targets				High-Level Targets			Self-Distillation		Contrastive / Multi-modal Teacher	
Algorithm	ViT [5]	MAE [70]	SimMIM [101]	Maskfeat [106]	BEiT [99]	CAE [100]	PeCo [107]	data2vec [108]	SdAE [109]	MimCo [110]	BEiT v2 [111]
Target	Raw Pixel			HOG	VQ-VAE		VQ-GAN	self		MoCo v3	CLIP

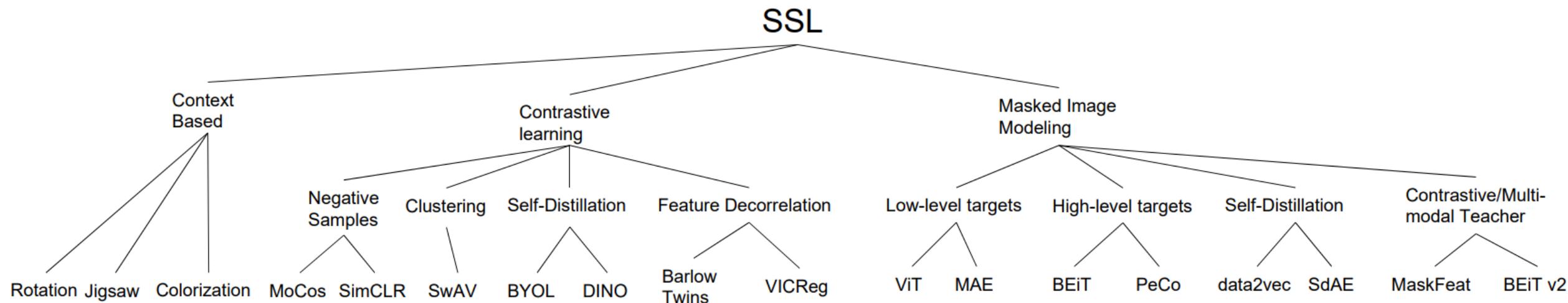
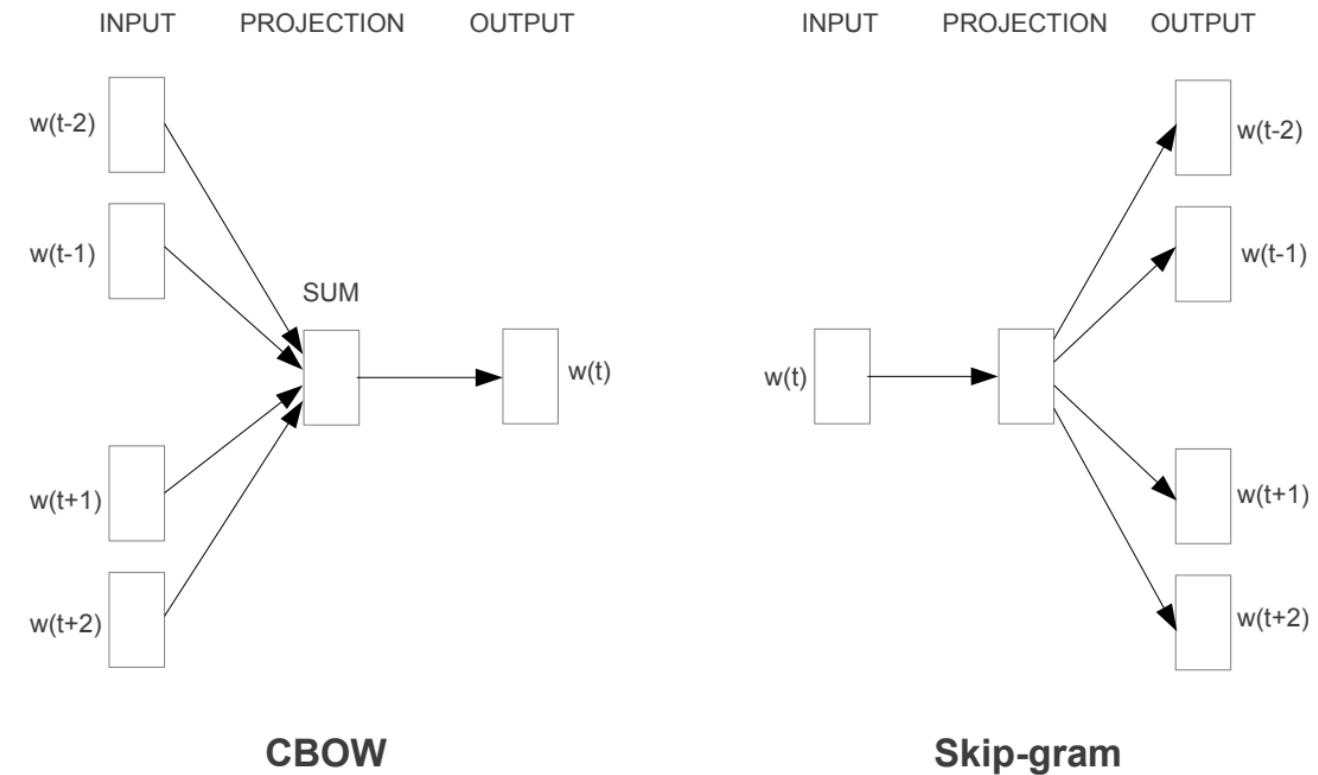
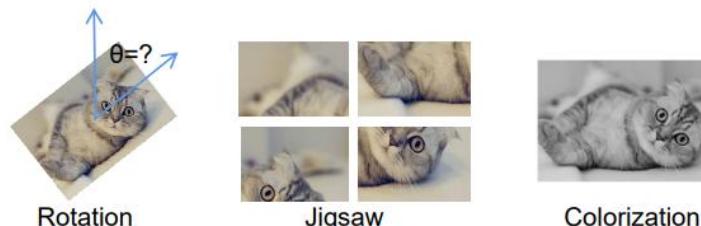


Fig. 8: Several representative pretext tasks of SSL.

Types of Self Supervised Learning

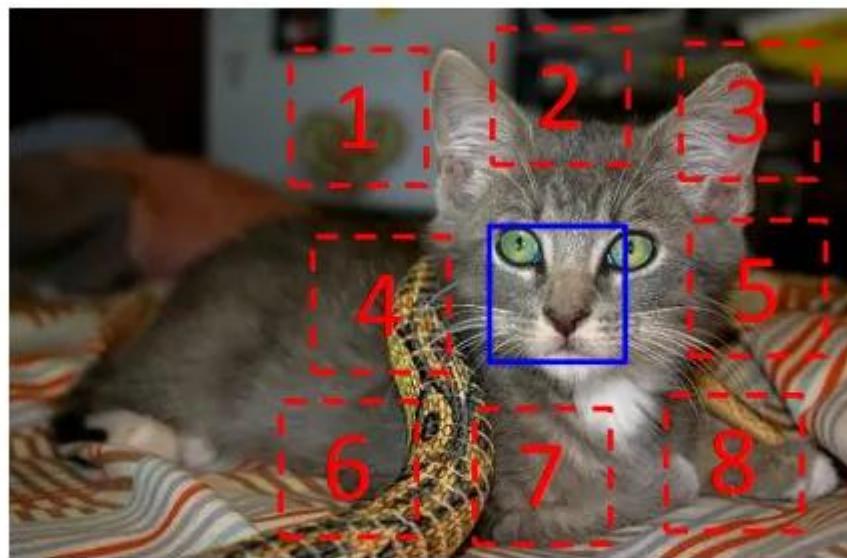
- Context based

- 基於資料本身的上下文
- Word2vec 主要是利用語句的順序，例如 CBOW 通過前後的詞來預測中間的詞，而 Skip-Gram 通過中間的詞來預測前後的詞。

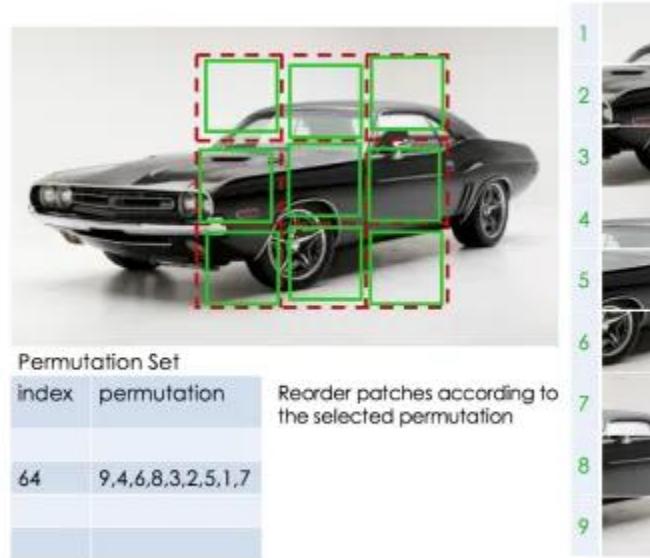


→ 1: New model architectures. The CBOW architecture predicts the current word based on the context, and the Skip-gram predicts surrounding words given the current word.

Fig. 4: Illustration of three common context-based methods: rotation, jigsaw, and colorization.

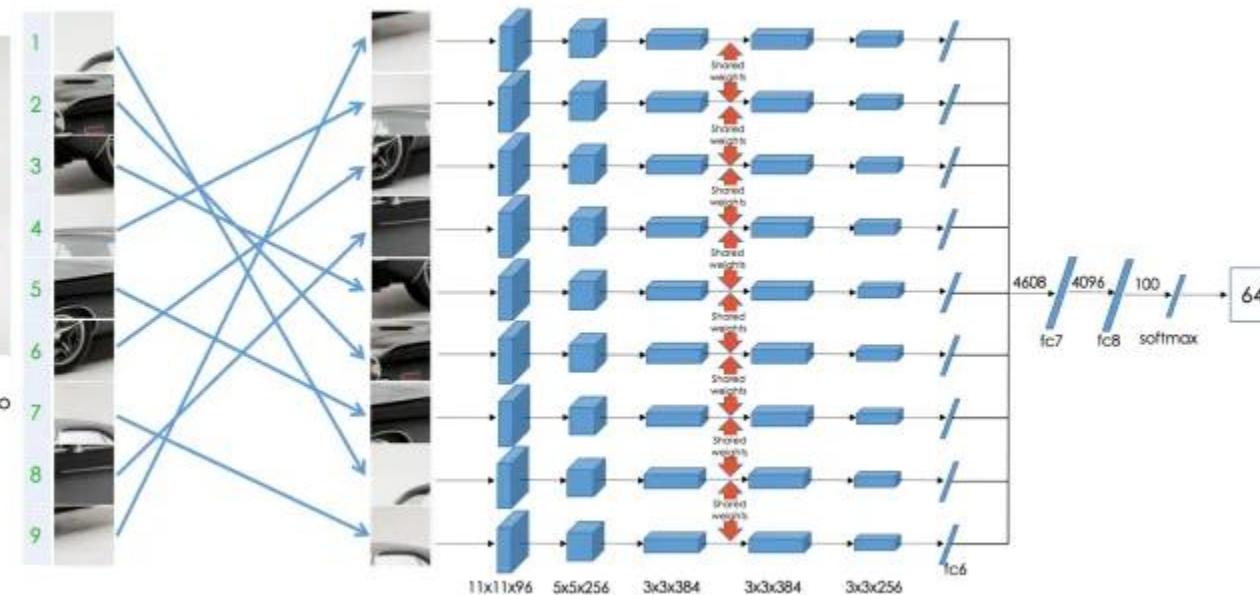


$$X = (\text{[cat eye]}, \text{[cat ear]}); Y = 3$$



透過在圖像中選定兩個 patch，並利用其中一個 patch 來預測另一個 patch 的相對位置。

模型僅基於 pixel 生成出 high-level representation 是比較困難的任務，並且運算開銷也很大，因此希望能採用其他不重建圖像的方式。Predictive methods 不需重建圖像，而是藉由對圖像做各種變換，再讓模型還原回來。

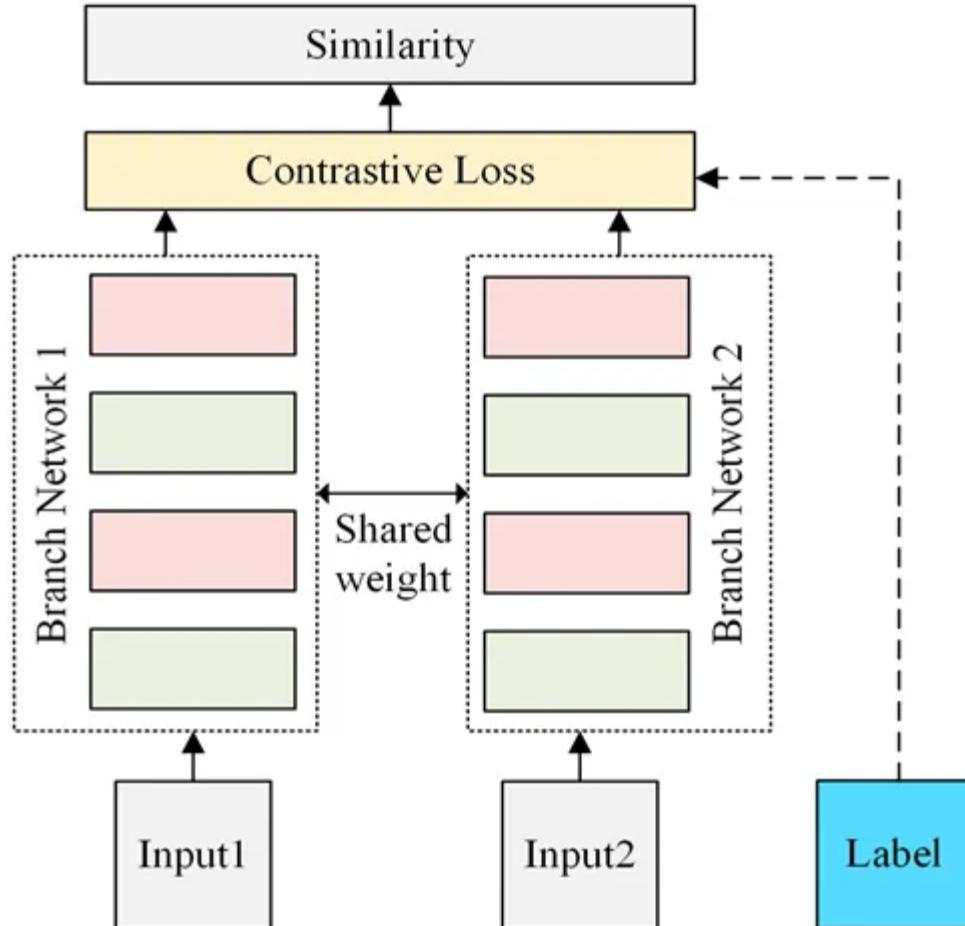


CONTRAST LEARNING

Contrastive Learning

- 概念
 - 相同類別的圖像間的相似度越高越好 (即距離盡可能地近) ,
 - 不同類別的圖像相似度越低越好 (即距離盡可能地遠) ,
 - 模型架構主要是使用 Siamese Network 。
- 代表作有
 - FAIR 提出的 MoCov1、MoCov2、MoCov3，Google Brain 提出的 SimCLRv1、SimCLRv2
- 演進
 - 早期的 CL 方法是基於負例的概念 。
 - 隨著 CL 的發展，一系列無需負例的方法應運而生 。

Contrastive Learning



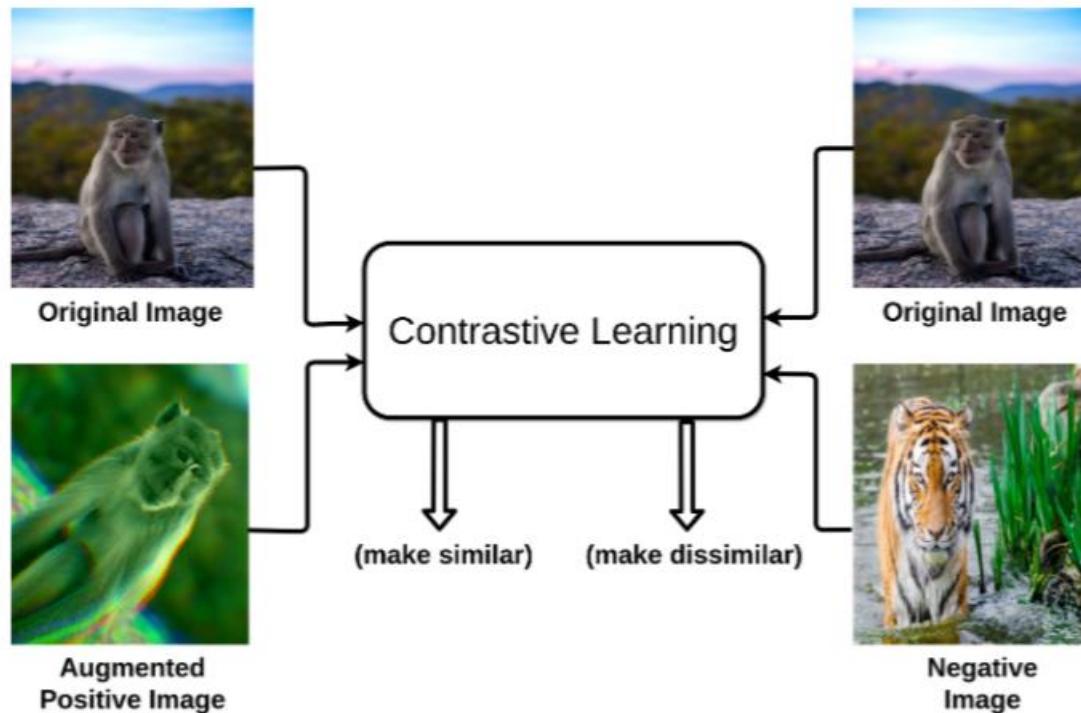
資料並沒有標籤，該怎麼區分哪些圖像是相同類別呢？因此我們先對圖像做各種 **data augmentation** 後，再讓模型去預測經過 **data augmentation** 後的圖像與原圖之間的相似度，該結果要越高越好，同時對其他圖的預測則是越低越好。

對比學習的 5 大要素

- 設計一套有效的對比學習 (Contrastive Learning, CL) 方法，核心在於
 - 如何在「**學習特徵的不變性 (Invariance)**」與「**防止模型坍塌 (Collapse)**」之間取得平衡。
- Data Augmentation:
 - 這是對比學習的基礎成分。提出新的數據擴增方法也很有學術價值，例如 MixUp, CutMix。
- Parallel Augmentation:
 - 指的是如何運用 noise data (擴增數據) 提取特徵。主要有兩種方式：端到端 End-to-End 與 動量編碼器 Momentum Encoder
- Architecture:
 - 指的是如何操作正負樣本來計算損失函數的架構設計。主要可分 4 類：End-to-End, Memory Bank, MoCo, Clustering。
- Loss Function:
 - 損失函數的設計。最經典的可能是 InfoNCE。
- Data-Modal:
 - 不同模態的數據會衍生不同的對比學習方法。

Contrast Learning

- One type of self-supervised learning



Basic intuition behind contrastive learning paradigm: push original and augmented images closer and push original and negative images away

關鍵: 如何定義正樣本和負樣本 (想的到的組合都可以)

History

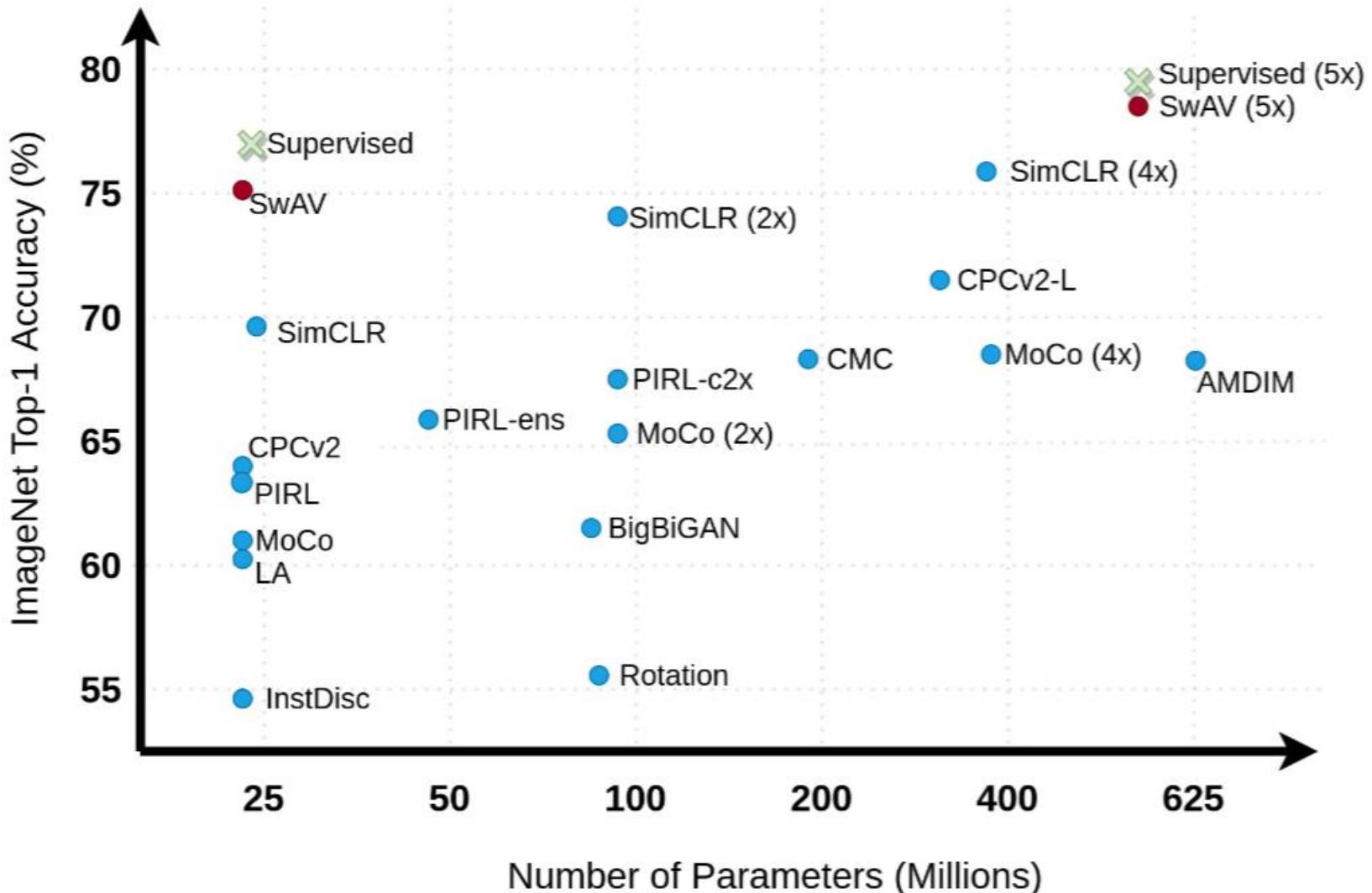
- 第一階段：百花齊放（2018 年至 2019 年中期）
- 此階段方法、模型、目標函數和代理任務尚未統一，呈現多元發展的狀態。
- 代表性工作包括：
 - InstDisc (Instance Discrimination)：
 - 奠定了對比學習的基礎，提出了「個體判別」代理任務，即將每張圖片視為獨立類別。
 - 使用 Memory Bank 儲存大量負樣本特徵，並引入動量更新機制。
 - InvaSpread：
 - 可視為 SimCLR 的前身，採用端到端學習，僅使用單一編碼器，正負樣本來自同一個 minibatch。
 - 由於缺乏 SimCLR 的數據增強、MLP Projector 和大 batch size，效果不如 SimCLR 顯著。
 - CPC (Contrastive Predictive Coding)：使用生成式任務
 - 使用「預測」作為代理任務，利用上下文特徵預測未來時刻的特徵輸出。
 - 可應用於圖像、音頻、文本等多種數據類型。
 - CMC (Contrastive Multiview Coding)：
 - 提出「多視角」概念，將同一物體的不同視角視為正樣本，例如：同一張圖片的 RGB 圖像、深度資訊、表面法線和分割圖像。
 - 證明了對比學習的靈活性，可應用於多視角、多模態學習，為後續 CLIP 等模型的發展奠定了基礎。

- 第二階段：雙雄爭霸（2019 年中期至 2020 年中期）
- 以 MoCo 和 SimCLR 為代表，這兩個模型在 ImageNet 上取得了突破性的成果，推動了對比學習的快速發展。
- MoCo：
 - 將對比學習歸納為字典查詢問題，提出了「queue」和「動量編碼器」來構建更大、更一致的字典。
 - 繼承了 InstDisc 的許多設計，包括使用 Res 50、128 維特徵、InfoNCE loss 和數據增強方式。
 - MoCo v2 融合了 SimCLR 的 MLP Projection head 和更強的數據增強技巧，進一步提升了性能。
- SimCLR：
 - 簡化了對比學習流程，僅使用單一編碼器，正負樣本來自同一個 minibatch。
 - 提出 MLP Projector，將編碼器輸出的特徵進行非線性變換，顯著提升了性能。
 - 採用更強的數據增強、更大的 batch size 和更長的訓練時間，進一步提升了性能。*
 - SimCLR v2 主要關注半監督學習，並通過更大的模型、更深的 MLP Projection head 和動量編碼器提升了自監督性能。
- 其他重要工作：
 - SwAV 結合了聚類和對比學習，並提出了 multi-crop 技巧。
 - CPC v2 和 InfoMin 分別對 CPC 和 CMC 進行了改進。

- 第三階段：擺脫負樣本（2020 年中期至 2021 年初）
 - 以 BYOL 和 SimSiam 為代表，探索了不使用負樣本的對比學習方法。
- BYOL：
 - 提出「自舉你的潛在表示」概念，不使用任何形式的負樣本，僅通過預測任務進行模型訓練。
 - 採用動量編碼器和 MLP Projector，並使用 MSE loss 作為目標函數。
 - 引發了關於 Batch Normalization 在 BYOL 中作用的討論，最終證明 Batch Normalization 主要作用是穩定訓練，而非提供隱式負樣本。
- SimSiam：
 - 進一步簡化了 BYOL，不使用動量編碼器和大 batch size，僅依靠 Stop Gradient 操作和對稱性 loss 避免模型坍塌。
 - 將 SimSiam 解釋為一種 EM 演算法，並提出了 k-means 聚類的解釋。
- Barlow Twins：
 - 提出新的目標函數，通過比較正樣本特徵的關聯矩陣和單位矩陣的相似性來進行模型訓練。

- 第四階段：Transformer 時代（2021 年至今）
 - Vision Transformer 的出現，為對比學習帶來了新的挑戰和機遇，研究重點轉向如何有效地訓練自監督 Vision Transformer 模型。
- MoCo v3：
 - 將 MoCo 框架擴展到 Vision Transformer，發現 ViT 在自監督訓練中存在不穩定性問題。
 - 提出凍結 patch projection layer 的技巧，有效解決了訓練不穩定問題。
- DINO：
 - 將自監督 Vision Transformer 訓練框架解釋為自蒸餾，並提出了 centering 操作來避免模型坍塌。
 - 發現自監督 Vision Transformer 可以準確地捕捉物體輪廓，展現出強大的潛力。

Contrast Learning



Unsupervised Feature Learning via Non-Parametric Instance Discrimination

- InstDisc

- 觀察

- 分類器輸出機率較高的類別通常是與該圖片內容相似的物體。例如，將一張豹子的圖片輸入分類器，分類結果中排名靠前的類別可能是獵豹、雪豹等，而排名靠後的類別則與豹子無關。

- 原因

- 並非僅僅因為相似的圖片擁有相似的語義標籤，
 - 而是因為它們在視覺上非常相似。
 - 因此提出將每個個體 (instance) 都看作是一個獨立的類別，並希望學習一種能夠區分每個個體的特徵表示方法。
 - 每個圖片都被視為一個獨立的類別

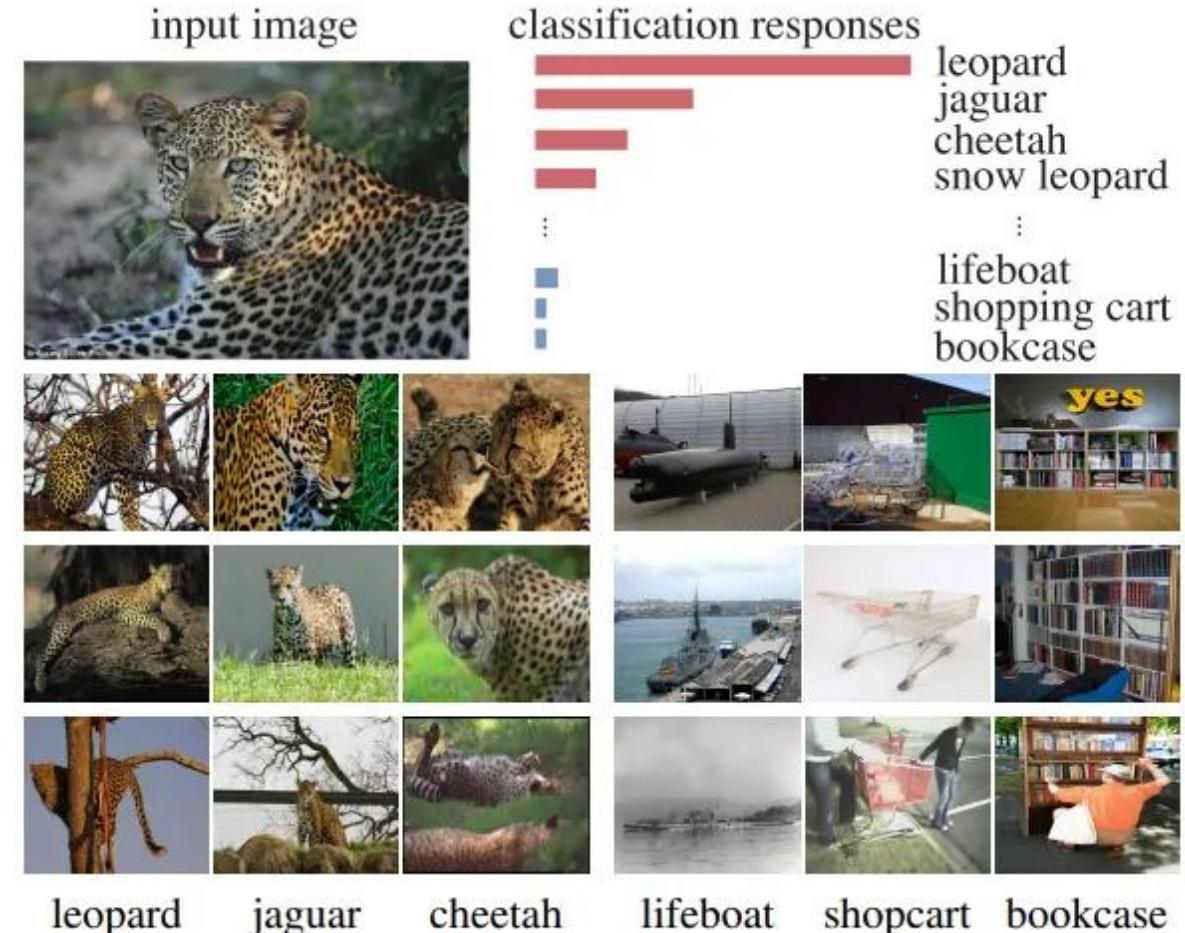
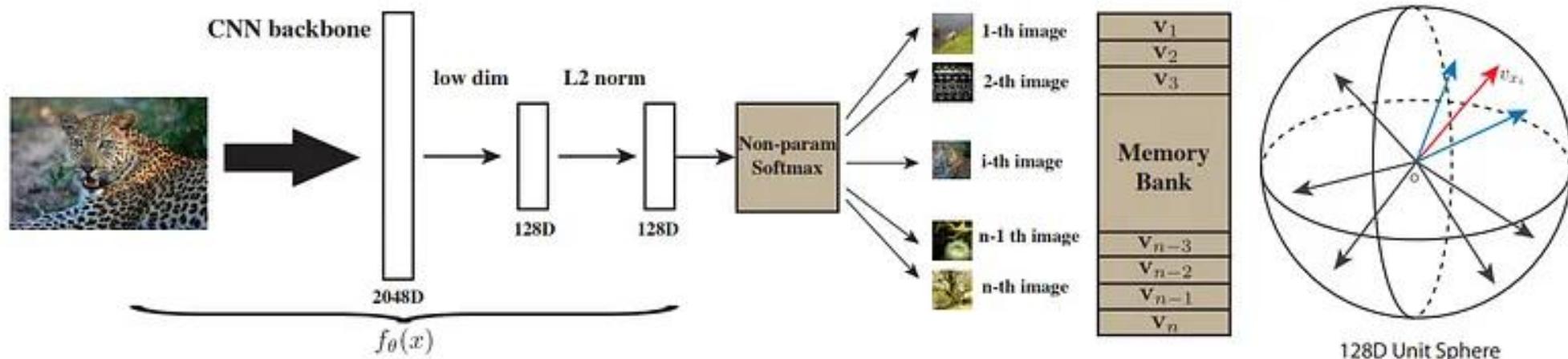


Figure 1: Supervised learning results that motivate our unsupervised approach. For an image from class *leopard*, the classes that get highest responses from a trained neural net classifier are all

- 正樣本
 - 圖片本身和其圖片的數據增強，
- 負樣本
 - 數據集中其餘的圖片
- 使用 memory bank 存儲所有圖片的特徵向量，訓練過程更新
- 使用 proximal regularization 對特徵向量進行動量式更新，類似 MoCo 的動量編碼器
- NCE(Noise Contrastive Estimation) loss



BatchSize為 256，那就會有 256 張正樣本進入 CNN Encoder。而負樣本從 Memory Bank 中隨機採樣出 4096 張(固定

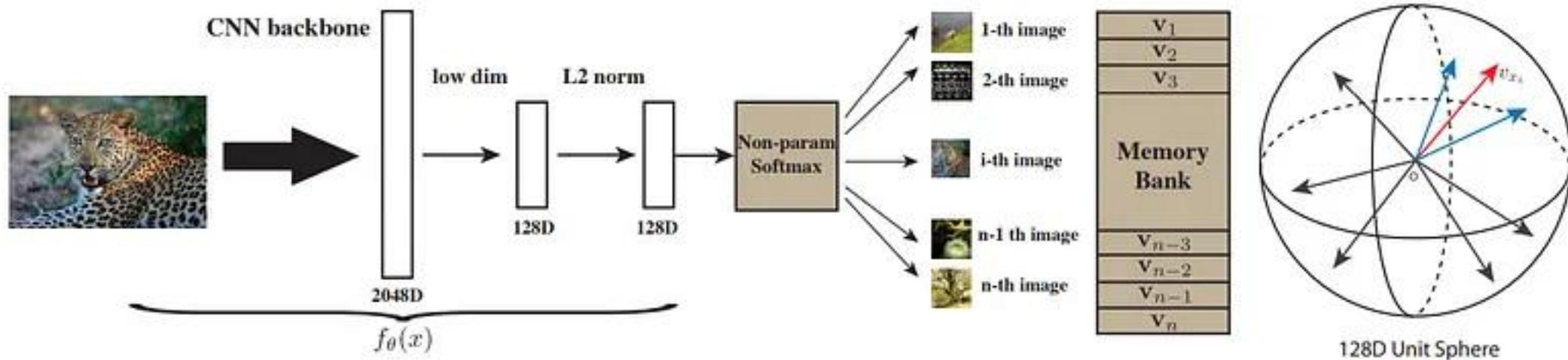


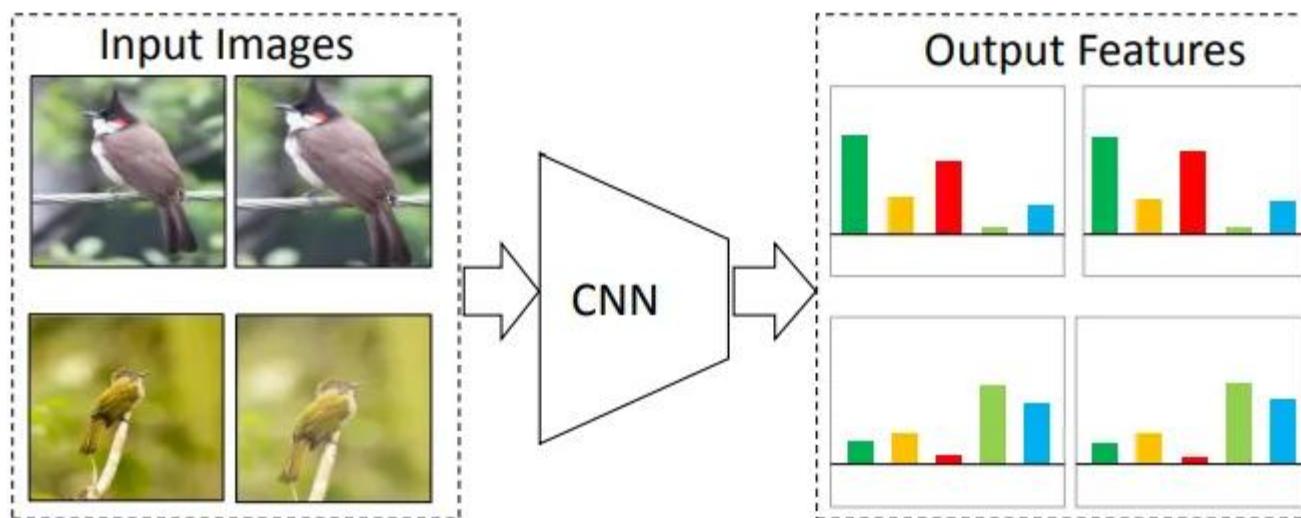
Figure 2: The pipeline of our unsupervised feature learning approach. We use a backbone CNN to encode each image as a feature vector, which is projected to a 128-dimensional space and L2 normalized. The optimal feature embedding is learned via instance-level discrimination, which tries to maximally scatter the features of training samples over the 128-dimensional unit sphere.

- 假設BatchSize為 256，那就會有 256 張正樣本進入 CNN Encoder。而負樣本從 Memory Bank 中隨機採樣出 4096 張，

- 訓練目標
 - 訓練一個網路，對輸入影像編碼，編碼後的特徵，在特徵空間裡能夠盡可能地分開，因為對於個體判別任務來說，每張圖片都是自己的類別
 - 正樣本
 - 圖片本身和其圖片的data augmentation
 - 負樣本
 - 數據集中其餘的圖片
- 特徵向量
 - 128 dimension
 - 使用 memory bank 存儲所有圖片的特徵向量，訓練過程更新特徵向量
 - 使用 proximal regularization 對特徵向量進行動量式更新，類似 MoCo 的動量編碼器
- NCE(Noise Contrastive Estimation) loss

Unsupervised Embedding Learning via Invariant and Spreading Instance Feature

- InvaSpread (正負樣本都來自同一個mini-batch)
 - 數據集中抽取一個 mini batch (e.g. N=256)的圖像，對所有圖片進行數據增強
 - 用mini-batch 為單位去做，可以不用memory bank，用一個編碼器做到end-to-end learning
 - 正樣本為一張圖片及它數據增強的圖片 (pos: 2)
 - 負樣本為剩下的圖片及它們數據增強的圖片。 (neg: 2N-2, (256-1)*2)



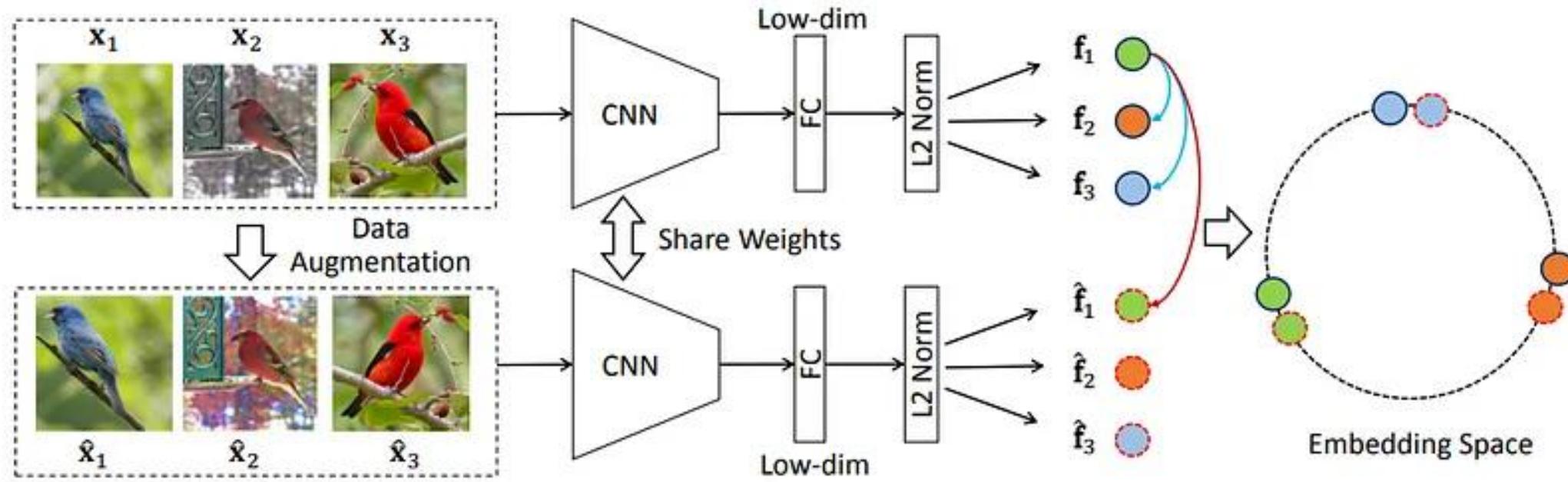
一樣用instance discrimination

對於相似的圖片
相似的物體呢
它的**特徵應該保持不變性**

但是對於不相似的物體
或者完全不沾邊的物體呢
它的**特徵應該盡可能的分散開**

Figure 1: Illustration of our basic idea. The features of the same

- InvaSpread
 - 沒有使用數據結構 Memory Bank 去存儲負樣本，InvaSpread 只用一個 Encoder 進行端到端的學習，可以在同一個 mini batch 中實現正負樣本的對比學習。InvaSpread 可以被理解成是 SimCLR 的前身。
- InvaSpread 跟 SimCLR 這麼像，為什麼沒有取得炸裂的成果呢
 - 是因為做對比學習的時候，**負樣本最好是足夠多**，字典必須足夠大
 - InvaSpread 的作者沒有 GPU，所以 Batchsize 設置為 256，意味著負樣本只有 $(256*2-2) = 510$ 個，負樣本屬實太少囉！



負樣本越多，模型學到的特徵越好

- 直覺比喻：選擇題的難度
 - 想像你要透過考試來學習分辨「哈士奇」。
- 情境 A (負樣本少)：
 - 題目是：「下面哪張是哈士奇？」 選項：(A) 哈士奇 (B) 飛機
 - 結果：你不需要知道哈士奇長什麼樣，你只要知道它「不是飛機」就能答對。模型會變得很懶惰，只學到非常粗糙的特徵（例如：它是活的）。
- 情境 B (負樣本多)：
 - 題目是：「下面哪張是哈士奇？」 選項：(A) 哈士奇 (B) 飛機 (C) 汽車 (D) 貓 (E) 狼 (F) 阿拉斯加雪橇犬 ... (共 1000 個選項)
 - 結果：為了在 1000 個選項中選對，你被迫學習非常詳細的特徵。你不能只學「它是活的」（因為有貓），也不能只學「它像狼」（因為有狼和雪橇犬）。你必須精準掌握哈士奇的毛色、眼睛形狀、耳朵位置。
- 結論：
 - 負樣本越多，任務越難，模型被迫學習到的特徵就越強大且具備鑑別力 (Discriminative)。

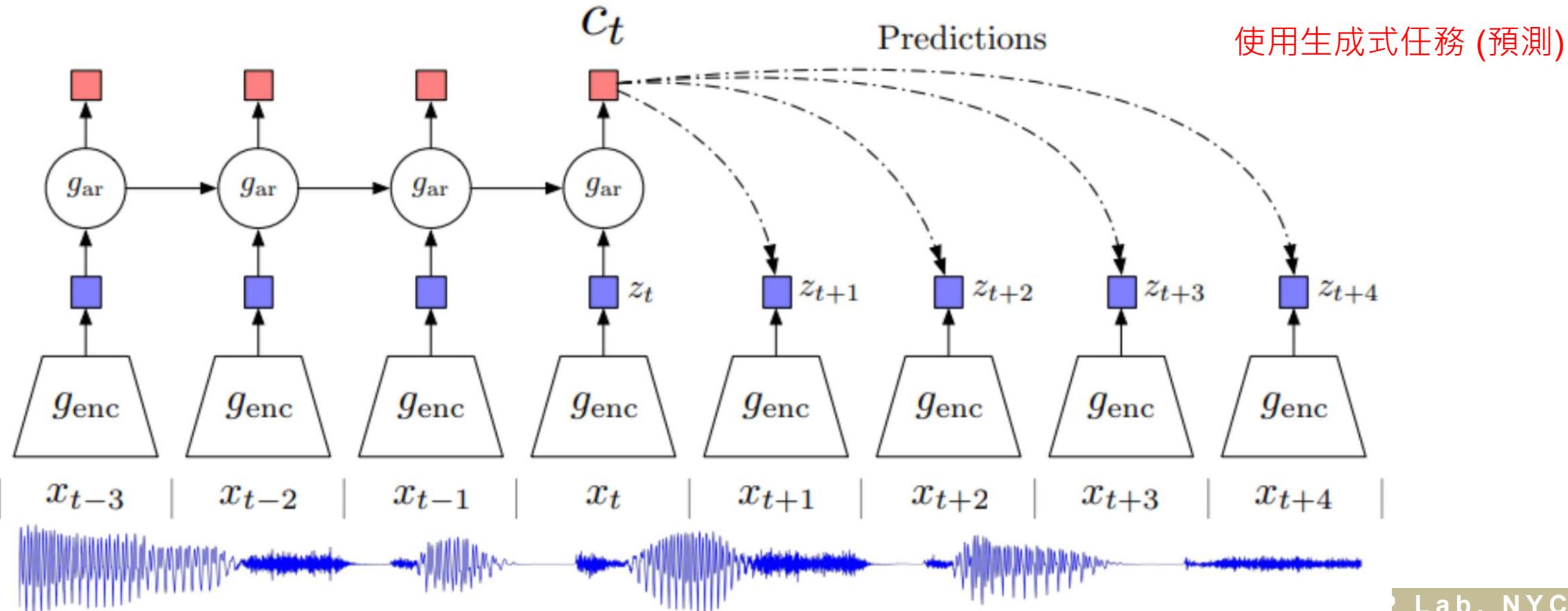
CPC

正樣本：未來時刻的輸入通過編碼器後得到的未來時刻的特徵輸出。

負樣本：可以是任意選取的輸入通過編碼器得到的特徵輸出，它們都應該與你的預測不相似。

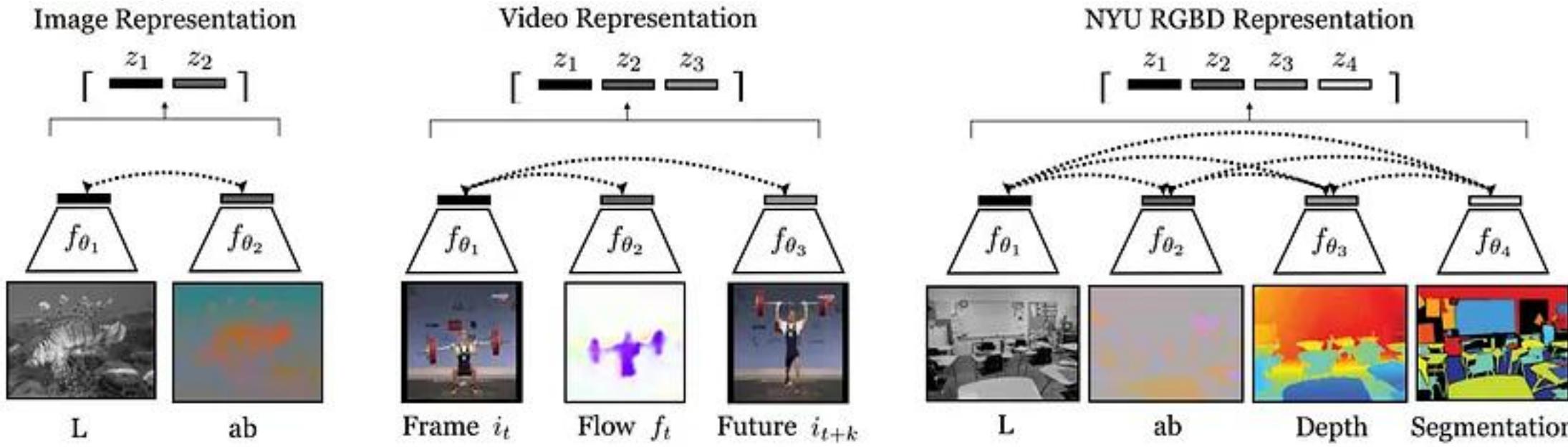
- Contrastive Predictive Coding (CPC)

- Although the figure shows audio as input, similar setup can be used for videos, images, text etc



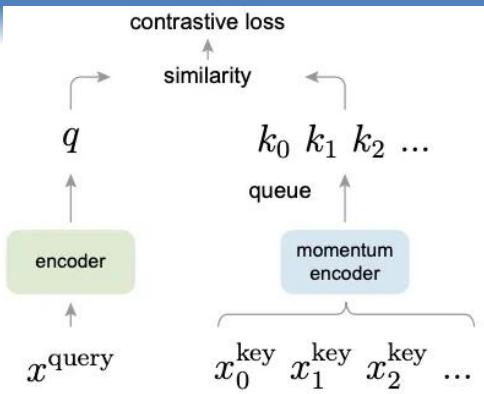
CMC: Contrastive Multiview Coding

- 利用多個視角進行對比學習的方法
 - 其核心概念是，一個物體可以從多個不同的角度或模態進行觀察，而這些不同的視角都包含著關於該物體的重要信息。
 - CMC 方法旨在學習一種能夠捕捉所有視角下關鍵因素的特征表示，從而使其具有視角不變性。
 - 來自同一物體的不同視角被視為正樣本，而來自不同物體的視角被視為負樣本

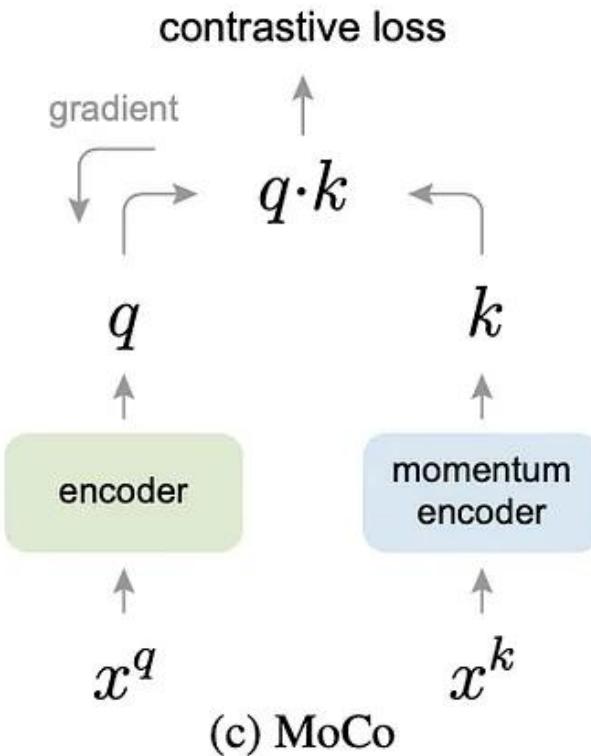
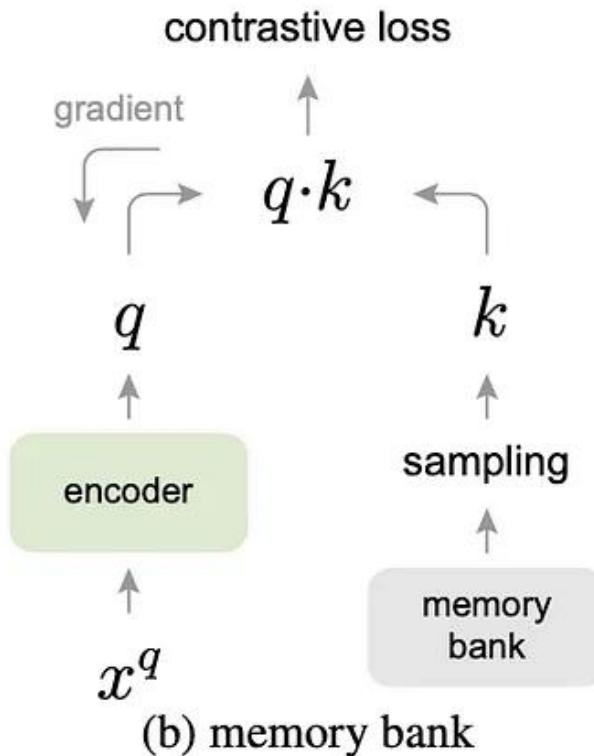
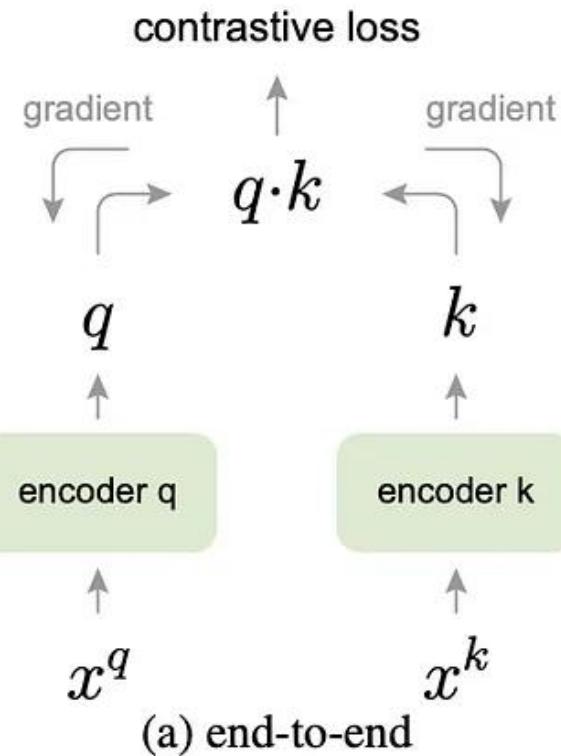


MOCO v1

- 基於 instDisc 的架構進行改進
 - 把 contrast learning 轉化為字典查找
 - 構建了一個帶有 queue 和 moving average Encoder 的 Dynamic Dictionary
區別
 - 字典構建方法：Queue 取代了 InstDisc 中的 Memory Bank
 - Memory Bank 的大小固定，並且更新速度較慢，這限制了對比學習的效能
 - **用 queue**，字典的大小可以動態調整
 - 特徵更新方式：Momentum (moving average) Encoder
 - MoCo 使用動量編碼器來更新特徵，而不是像 InstDisc 那樣直接更新 Memory Bank 中的特徵。這種動量更新的方式可以使字典中的特徵更加一致，有利於對比學習。
 - 使用 queue 可以使字典變大，但它也使得通過反向傳播更新 key-Encoder 變得更難。
 - 直接複製：
 - 從 Queue encoder 複製 weight 到 Key encoder，效果不好，
 - 假設這種失敗是由快速變化的編碼器引起的，這降低了我們希望的 representation 一致性
(字典不一致問題)。提出 Momentum update 方案：
 - 只有參數 θ_q 通過反向傳播更新



$$\theta_k \leftarrow m\theta_k + (1 - m)\theta_q.$$



負樣本的規模就是 batch size，即字典的大小就是 batch size
受限 GPU，很難應用大量的負樣本

採用一個較大的 memory bank 儲存較大的字典，字典可以設定很大，但是一致性卻較差

$$\theta_k \leftarrow m\theta_k + (1 - m)\theta_q.$$

SimCLR v1

- 訓練過程：
 - 從一個 mini-batch 中選取 N 張圖片。
 - 對每張圖片進行兩次不同的數據增強，生成 $2N$ 個增強後的圖片。
 - 將所有 $2N$ 個增強後的圖片輸入編碼器，生成 $2N$ 個特徵向量。
 - 將所有 $2N$ 個特徵向量輸入 **Projection Head**，生成 $2N$ 個低維特徵向量。
 - 計算對比學習的損失函數（例如 InfoNCE loss），使得來自同一張圖片的增強版本的特徵向量彼此靠近，而來自不同圖片的增強版本的特徵向量彼此遠離。
 - 更新encoder和 **Projection Head** 的參數。

正負樣本都來自於同一個 mini-batch
需要使用大的 batch size 才能取得好的效果。

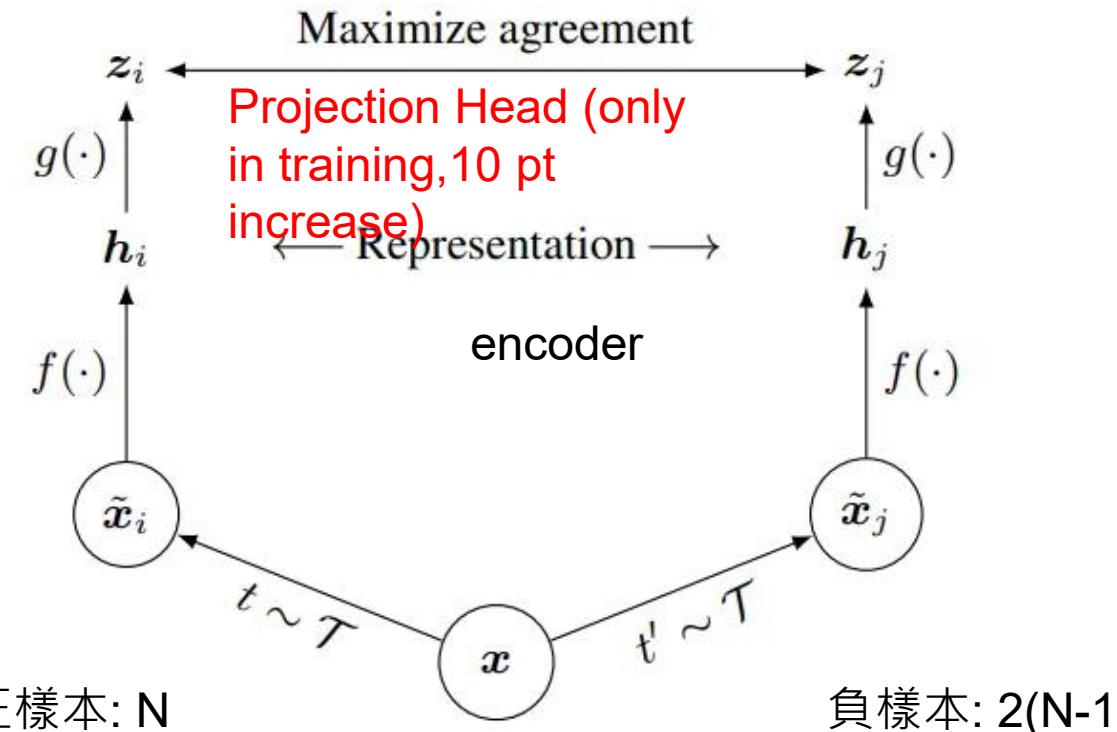


Figure 2. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder network $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss. After training is completed, we throw away the projection head $g(\cdot)$ and

Projection Head 在 SimCLR 中的作用

- 作用
 - 將編碼器 (Encoder) 學到的特徵 (h) 映射到另一個空間 (z)，以便在該空間中計算對比損失 (Contrastive Loss)
- 核心作用：過濾與資訊緩衝 (Information Buffer)
 - 對比學習的目標：
 - 對比損失強迫模型對「數據增強 (Augmentation)」保持不變性 (Invariance)。例如，一張圖片旋轉或變色後，模型仍要認出它是同一張圖。
 - 潛在問題：
 - 為了滿足這種「不變性」，模型可能會被迫丟棄一些對於下游任務（如分類）其實很有用的資訊（例如物體的顏色、具體的姿態或局部細節）。
- Projection Head 的解法：
 - Projection Head 吸收了這種「資訊丟失」。它允許 z 變得對增強操作高度不變（為了最小化 Loss）。
 - 這保護了之前的 h (Encoder 輸出)，使其能夠保留更多原始圖像的豐富特徵。
 - 實驗證明， h 的特徵包含的資訊量遠多於 z ，因此在下游任務（如 ImageNet 分類）微調時，我們使用 h ，並丟棄 z 。

- 想像 h 是你對一個人的「詳細傳記」
 - (包含身高、穿著、髮型、長相)。
- z 則是為了通過海關檢查的「身分驗證碼」。
- 為了快速驗證 (Contrastive Loss)，
 - z 必須忽略你今天穿什麼衣服或髮型有沒有亂 (不變性)。
 - 如果沒有 Projection Head，你的「詳細傳記」 h 就會被迫變成簡略的「身分驗證碼」，導致細節丟失。
 - Projection Head 就是那個負責把詳細資訊轉化為簡略驗證碼的中介，讓 h 可以保持詳細。
- Projection Head 在 SimCLR 中的作用可以歸納為：
 - **解耦 (Decoupling)**：將「對比學習的訓練目標」與「下游任務的特徵需求」分開。
 - **吸收不變性 (Absorbing Invariance)**：讓 $g()$ 去承受數據增強帶來的信息損失，從而保護 h 保留更豐富的語義特徵。

Why Momentum Encoder

- **負樣本 (Negative Samples)**
 - 作用是提供「對照組」。負樣本越多，模型就必須學會分辨越細微的差異，從而練就「火眼金睛」。
- 核心挑戰：負樣本數量與顯存的矛盾
 - 對比學習 (Contrastive Learning) 中，模型的訓練效果通常與**負樣本 (Negative Samples)** 的數量正相關。負樣本越多，模型學到的特徵越好。
 - **SimCLR 的做法 (End-to-End)**：
 - 使用超大 Batch Size (例如 4096)，同一個 Batch 內的其他圖片互為負樣本。這對 GPU 顯存要求極高。
 - **MoCo 的解法 (Queue)**：
 - 為了省顯存，MoCo 建立了一個隊列 (**Queue**) 來存儲過去多個 Batch 的特徵，當作負樣本。這樣 Batch Size 可以很小，但負樣本隊列可以很大 (例如 65536)。

- Momentum Encoder 的作用：解決「特徵過期」問題
 - 使用隊列雖然解決了數量問題，但帶來了一個新問題：特徵不一致 (**Inconsistency**)。
- 問題場景：隊列裡的特徵是來自不同時間點 (不同 **Iteration**) 的編碼器生成的。
 - 隊列頭部的特徵：可能是 1000 次迭代前的編碼器生成的。
 - 當前的 Query 特徵：是現在的編碼器生成的。
- 後果：
 - 如果編碼器更新太快（例如使用標準的梯度下降），那麼「現在的編碼器」和「1000 次前的編碼器」參數差異巨大。這就像是用「現在的匯率」去和「10 年前的物價」做比較，導致計算出的 Loss 沒有意義，模型無法收斂。
- **Momentum Encoder** 就是為了解決這個「匯率波動」問題：
 - 它強制 Key Encoder (負責生成隊列特徵的編碼器) 更新得非常非常慢，使其參數演變極其平滑。

3. 運作機制 (數學原理)

MoCo 有兩個編碼器：

1. **Query Encoder** (f_q , 參數 θ_q)：標準編碼器，使用梯度下降 (Gradient Descent) 進行更新，更新速度快。
2. **Momentum Encoder** (f_k , 參數 θ_k)：不進行梯度回傳 (No Backpropagation)，而是使用動量更新：

$$\theta_k \leftarrow m \cdot \theta_k + (1 - m) \cdot \theta_q$$

- m (**Momentum Coefficient**)：通常設得非常大 (例如 0.999)。
- 意義： θ_k 每次只吸收 0.1% 的新參數 (θ_q)，保留 99.9% 的舊參數。

這使得 Momentum Encoder 成為一個緩慢移動的平均值 (Moving Average)。雖然 Query Encoder 劇烈變化，但 Momentum Encoder 保持穩定，確保了隊列中「新特徵」與「舊特徵」之間的差異盡可能小 (具有可比性)。

方法	負樣本來源	編碼器更新方式	缺點
SimCLR	當前 Batch	梯度下降 (即時更新)	需要巨大的 Batch Size (顯存殺手)
Memory Bank	存儲在記憶體庫	(特徵固定，週期性更新)	特徵嚴重過期，不一致性高
MoCo	隊列 (Queue)	動量更新 (Momentum Update)	訓練稍慢，但能用小 Batch Size 達到極佳效果

一句話總結Momentum Encoder 的作用

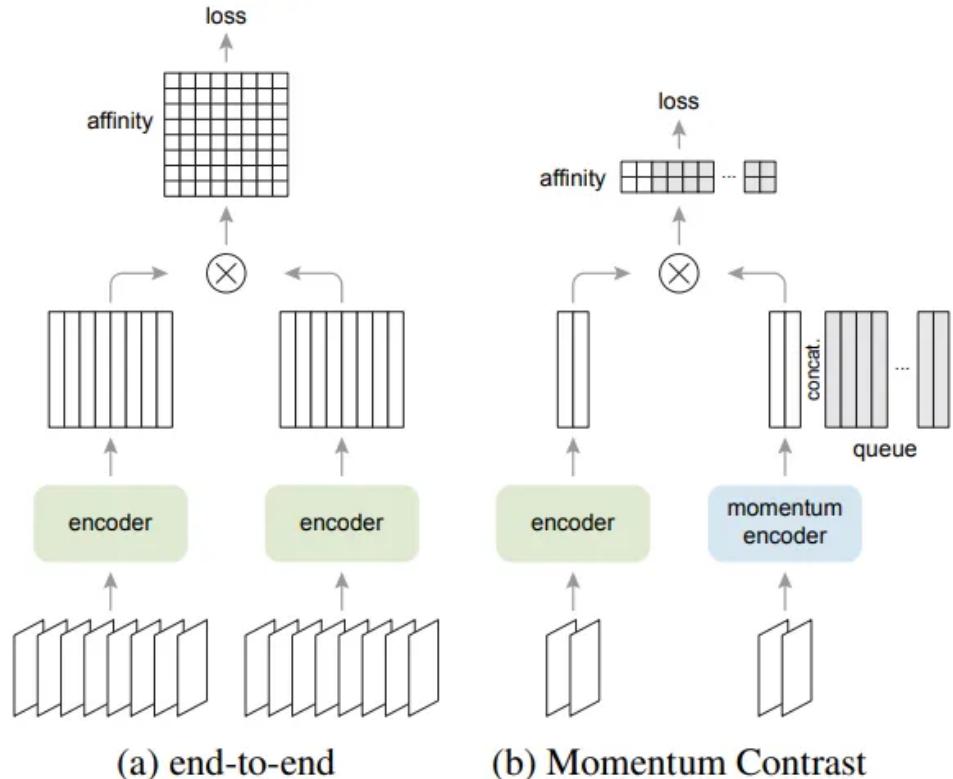
是平滑 Key Encoder 的參數變化，使得 MoCo 能夠安全地使用來自不同歷史時刻的特徵構建巨大的負樣本隊列，而不用擔心特徵分布差異過大導致訓練崩潰。

SimCLR vs MoCo

- 架構差異
 - SimCLR : SimCLR 採用端到端的學習方式，只使用一個編碼器來處理所有圖片。它不需要額外的數據結構（例如 memory bank 或隊列）來存儲負樣本，它的正負樣本都來自於同一個 mini-batch 。
 - MoCo : MoCo 使用兩個獨立的編碼器：查詢編碼器和鍵編碼器。查詢編碼器用於生成輸入圖片的特徵向量，而鍵編碼器用於生成負樣本特徵向量。為了存儲大量的負樣本特徵向量，MoCo 使用了隊列這種數據結構。
- 訓練策略差異
 - 數據增強：SimCLR 和 MoCo 都強調了數據增強在對比學習中的重要性，但 SimCLR 使用了更強大的數據增強技術，例如組合多種數據增強方法。
 - Projection Head : SimCLR 和 MoCo v2 都引入了 Projection Head，它是一個多層感知器 (MLP)，用於將特徵向量映射到一個低維空間。SimCLR v2 發現使用更深的 Projection Head 可以進一步提升模型的性能。
 - 動量編碼器：MoCo 使用動量編碼器來生成負樣本特徵向量，這確保了負樣本特徵向量的一致性和穩定性。SimCLR v2 也採用了動量編碼器，但發現它的提升幅度不如 MoCo 顯著。
 - Batch Size : SimCLR 需要使用很大的 batch size 才能取得好的效果，這對硬體資源的要求較高。MoCo 可以使用較小的 batch size 進行訓練，因此對硬體資源的要求較低。

MOCO v2

- SimCLR v1 => MOCO v2
 - 大 Batchsize、多 Epochs、更多更強的數據增強、增加一個 projection 層



case	MLP	unsup. pre-train				ImageNet acc.
		aug+	cos	epochs	batch	
MoCo v1 [6]				200	256	60.6
SimCLR [2]	✓	✓	✓	200	256	61.9
SimCLR [2]	✓	✓	✓	200	8192	66.6
MoCo v2	✓	✓	✓	200	256	67.5
<i>results of longer unsupervised training follow:</i>						
SimCLR [2]	✓	✓	✓	1000	4096	69.3
MoCo v2	✓	✓	✓	800	256	71.1

Table 2. **MoCo vs. SimCLR**: ImageNet linear classifier accuracy (**ResNet-50, 1-crop 224×224**), trained on features from unsupervised pre-training. “aug+” in SimCLR includes blur and stronger color distortion. SimCLR ablations are from Fig. 9 in [2] (we thank the authors for providing the numerical results).

Figure 1. A **batching** perspective of two optimization mechanisms for contrastive learning. Inputs are encoded into representations

case	unsup. pre-train				ImageNet acc.	VOC detection		
	MLP	aug+	cos	epochs		AP ₅₀	AP	AP ₇₅
supervised					76.5	81.3	53.5	58.8
MoCo v1				200	60.6	81.5	55.9	62.6
(a)	✓			200	66.2	82.0	56.4	62.6
(b)		✓		200	63.4	82.2	56.8	63.2
(c)	✓	✓		200	67.3	82.5	57.2	63.9
(d)	✓	✓	✓	200	67.5	82.4	57.0	63.6
(e)	✓	✓	✓	800	71.1	82.5	57.4	64.0

Table 1. **Ablation of MoCo baselines**, evaluated by ResNet-50 for (i) ImageNet linear classification, and (ii) fine-tuning VOC object detection (mean of 5 trials). “MLP”: with an MLP head; “aug+”: with extra blur augmentation; “cos”: cosine learning rate schedule.

mechanism	batch	memory / GPU	time / 200-ep.
MoCo	256	5.0G	53 hrs
end-to-end	256	7.4G	65 hrs
end-to-end	4096	93.0G [†]	n/a

Table 3. **Memory and time cost** in 8 V100 16G GPUs, implemented in PyTorch. [†]: based on our estimation.

SimCLR v2

- 換了**更大的模型**，取得更好的 performance
 - 讓微調的模型作為 Teacher 模型，可以使用 Student 小網絡來壓縮大網絡的訊息，將訓練好的大的網絡的知識，灌入小的網絡中

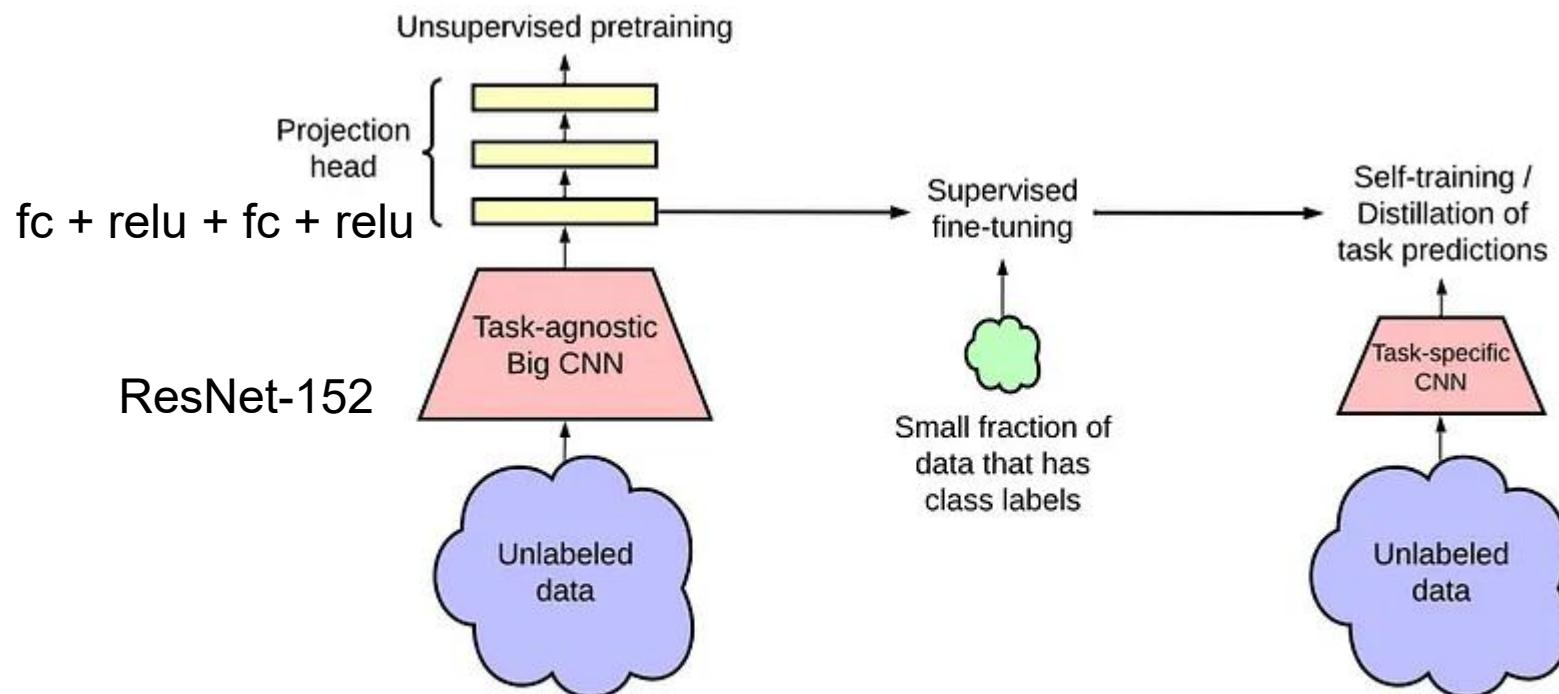


Figure 3: The proposed semi-supervised learning framework leverages unlabeled data in two ways:
(1) task-agnostic use in unsupervised pretraining, and (2) task-specific use in self-training / distillation.

simCLR v2 嘗試使用 MOCO 的動量編碼器，發現效果提升的 1%，提升的沒有很顯著，可能原因為 SimCLR v2 已經有很大的 Batchsize 了，所以不需要動量編碼器以及隊列的負樣本了

SwAV (Swap Assignment Views)

- 結合 clustering 和對比學習，學習具有視角不變性的強大特徵表示
 - SwAV 不直接使用特徵進行對比，利用聚類中心 (e.g. 3000) 來代替負樣本，語義明確有利學習，從而減少計算成本並提升訓練效率
 - 使用 Swapped Prediction 機制，利用一個視角的特徵向量和聚類中心來預測另一個視角的聚類分配向量

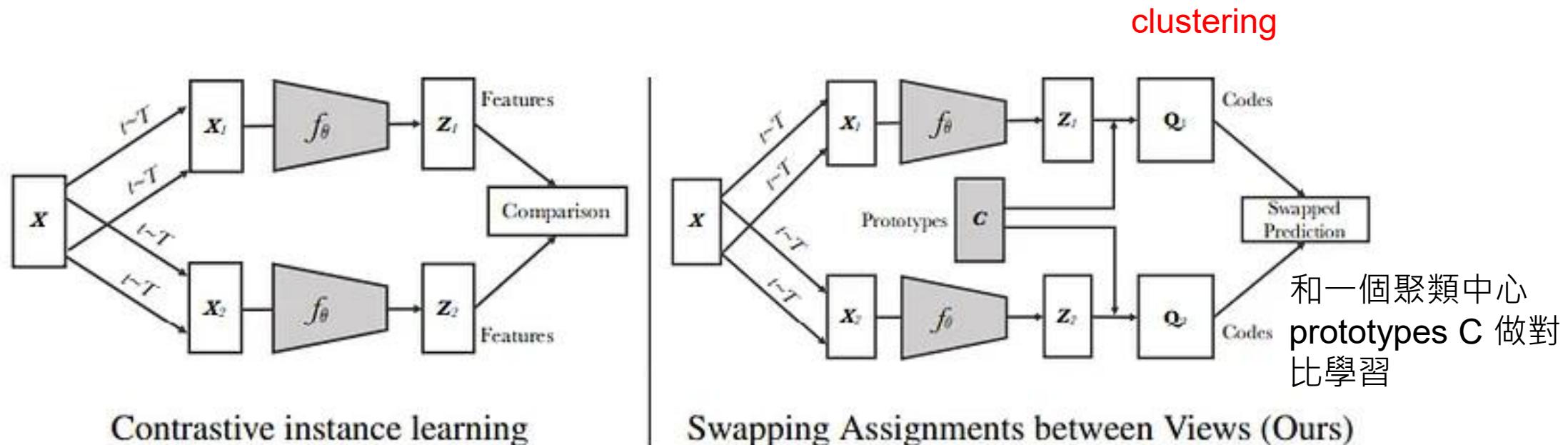


Figure 1: **Contrastive instance learning (left) vs. SwAV (right).** In contrastive learning methods

SwAV

- 核心理念是：
 - 我們不需要直接比較兩張圖片的特徵距離，而是比較它們「被歸類到哪裡」。
 - **SimCLR/MoCo (Instance Discrimination):**
 - 把每一張圖片都當作一個獨立的類別。如果數據集有 100 萬張圖，就有 100 萬個類別。這計算量很大，且語義上有些浪費（兩張不同的狗應該算同一類，但在這裡被視為負樣本）。
 - **SwAV (Clustering):** 假設數據集中存在 K 個原型 (**Prototypes**) (例如 $K=3000$)。這些原型類似於聚類中心。
 - SwAV 的目標是發現這些原型，並學習將圖片分配給它們。

運作機制：交換預測 (Swapping Assignments)

- SwAV 的名字 "Swapping Assignments between Views" 完美描述了它的訓練過程：
- **增強 (Augmentation):**
 - 對同一張圖片 x 進行兩次增強，得到視角 x_1 和 x_2 。
- **特徵提取:**
 - 通過編碼器 (ResNet) 得到特徵 z_1 和 z_2 。
- **聚類分配 (Clustering/Code):**
 - 將特徵 z_1 與所有原型向量 (Prototypes, C) 進行比對，算出它屬於哪個聚類，得到分配碼 (Code) q_1 。
 - 同理，算出 z_2 的分配碼 q_2 。
- **交換預測 (The Swap):** 這是最騷的操作。
 - 用 z_1 (視角 1 的特徵) 去預測 q_2 (視角 2 的分配碼)。
 - 用 z_2 (視角 2 的特徵) 去預測 q_1 (視角 1 的分配碼)。
- **直覺理解 :**
 - 如果 x_1 和 x_2 來自同一張圖（例如都是一隻貓），那麼 x_2 覺得自己屬於「貓類原型」， x_1 應該也要能預測出這個結果。

關鍵技術：Sinkhorn-Knopp 算法 (解決坍塌)

- 簡單地讓模型預測分類，模型很容易坍塌 (**Collapse**)：
 - 它會把所有圖片都分到同一個原型裡（例如全部歸為第 1 類），這樣 Loss 就最小。
- 為了防止這種情況，SwAV 引入了 **Sinkhorn-Knopp 算法**。
- 作用：
 - 強制均分 (**Equipartition**)。
- 效果：
 - 它限制在一個 Batch 內，所有圖片必須平均地分配到那 3000 個原型中。不能讓某個原型獨占所有圖片。這保證了原型的多樣性和特徵的豐富度。

- Multi-crop

- 以往都在一張 256×256 的圖片上用兩個 224×224 的 crop 提取兩個正樣本，但是因為 crop 過大了，所選取的 crop 都是基於全局特徵的，很多局部特徵才是非常有價值的，
- SwAV 選擇了兩個 160×160 的 crop 去提取全局特徵，選擇四個 96×96 的 crop 提取局部特徵，在計算量沒有劇增的情況下，獲取更多的正樣本。

Method	Top-1		Δ
	2x224	2x160+4x96	
Supervised	76.5	76.0	-0.5
<i>Contrastive instance approaches</i>			
SimCLR	68.2	70.6	+2.4
<i>Clustering-based approaches</i>			
SeLa-v2	67.2	71.8	+4.6
DeepCluster-v2	70.2	74.3	+4.1
SwAV	70.1	74.1	+4.0

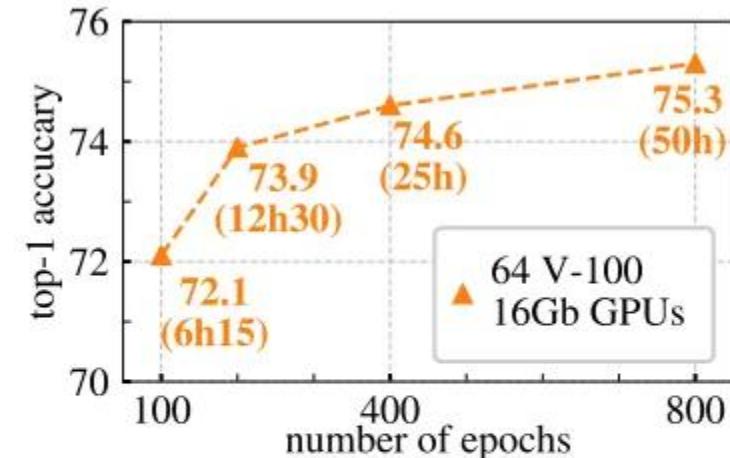


Figure 3: Top-1 accuracy on ImageNet with a linear classifier trained on top of frozen features from a ResNet-50. (left) **Comparison between clustering-based and contrastive instance methods and impact of multi-crop.** Self-supervised methods are trained for 400 epochs and supervised models for 200 epochs. (right) **Performance as a function of epochs.** We compare SwAV models trained with different number of epochs and report their running time based on our implementation.

Method	Arch.	Param.	Top1
Supervised	R50	24	76.5
Colorization [65]	R50	24	39.6
Jigsaw [46]	R50	24	45.7
NPID [58]	R50	24	54.0
BigBiGAN [15]	R50	24	56.6
LA [68]	R50	24	58.8
NPID++ [44]	R50	24	59.0
MoCo [24]	R50	24	60.6
SeLa [2]	R50	24	61.5
PIRL [44]	R50	24	63.6
CPC v2 [28]	R50	24	63.8
PCL [37]	R50	24	65.9
SimCLR [10]	R50	24	70.0
MoCov2 [11]	R50	24	71.1
SwAV	R50	24	75.3

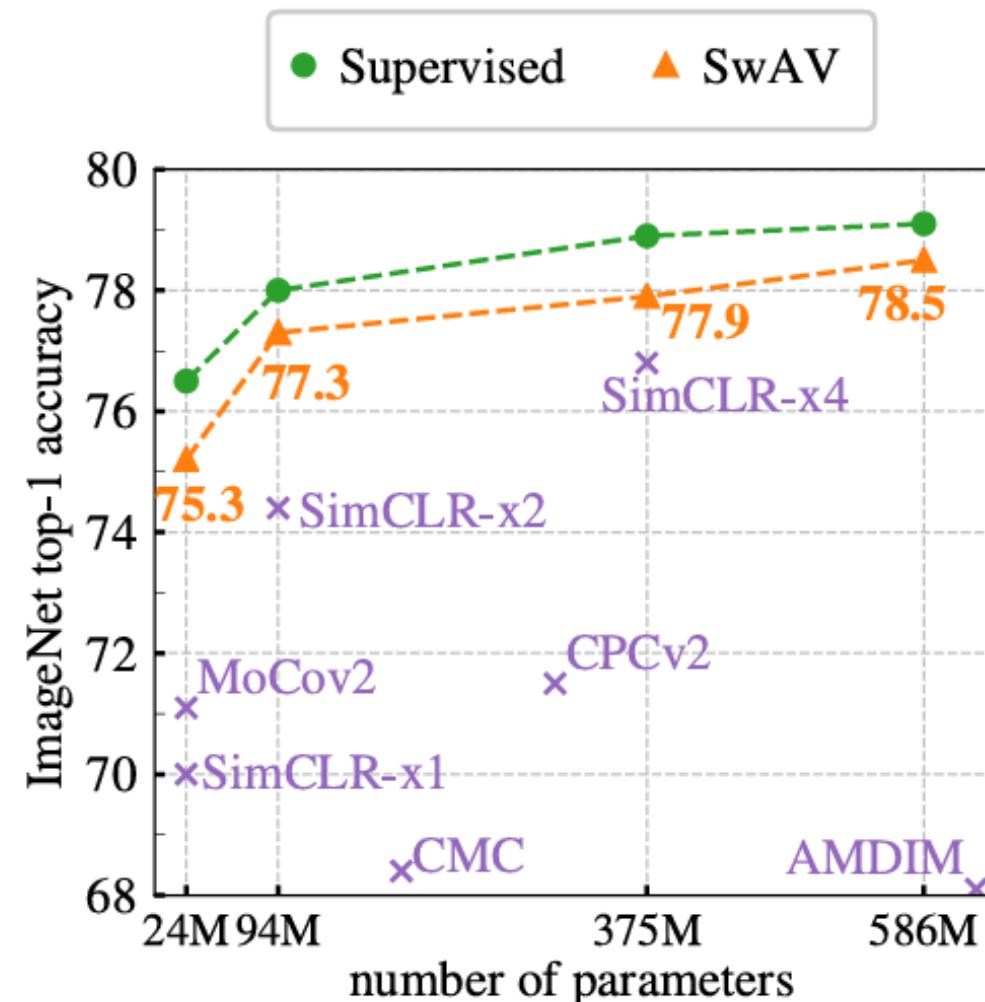
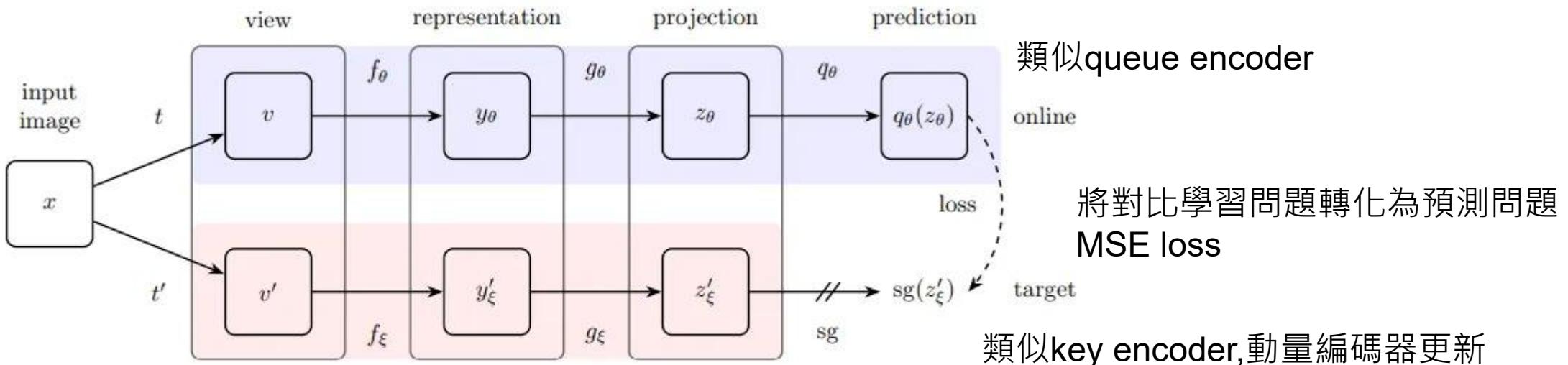


Figure 2: **Linear classification on ImageNet.** Top-1 accuracy for linear models trained on frozen features from different self-supervised methods. **(left)** Performance with a standard ResNet-50. **(right)** Performance as we multiply the width of a ResNet-50 by a factor $\times 2$, $\times 4$, and $\times 5$.

- 負樣本的必要性
 - 假設模型的輸入只有正樣本，那麼模型需要讓正樣本之間的距離盡可能地縮小，但對模型來說正樣本之間是相似的，所有正樣本之間的距離無限接近。這會導致模型學習一個捷徑解，即無論輸入是什麼，模型都返回相同的輸出
 - 此時添加負樣本可以對模型形成一個約束，讓正樣本之間的距離接近，讓負樣本之間的距離拉遠，所以添加負樣本是非常關鍵的，讓模型不會坍塌。
- SwAV
 - 已經稱得上不用負樣本了，它是和一個聚類中心 $\text{prototypes } C$ 做對比學習
- 可以不要負樣本嗎？

BYOL: Bootstrap Your Own Latent

- 架構: 提出正樣本自己和自己學習，完全沒有負樣本或者聚類中心
 - 通過預測機制來學習特徵表示，即利用一個視角的特徵來預測另一個視角的特徵
 - BYOL 採用孿生網絡架構，包含兩個具有相同網絡結構但參數不同的編碼器
 - f_θ 會隨著梯度更新而更新，而 f_ξ 則使用動量更新方式來更新
 - 整體思路是將上半部的網絡拿來預測下半部網絡的內容，讓二者盡量相似



Simsiam

- 也不需要用負樣本，更是直接把動量編碼器給拿掉了
 - 將一張圖片 X 經過兩個數據增強變為 X_1 和 X_2 ，接著經過孿生的 Encoder f ，將左半部網路得到的 Embedding 向量放入 predictor h ，拿來預測右半部網路的內容

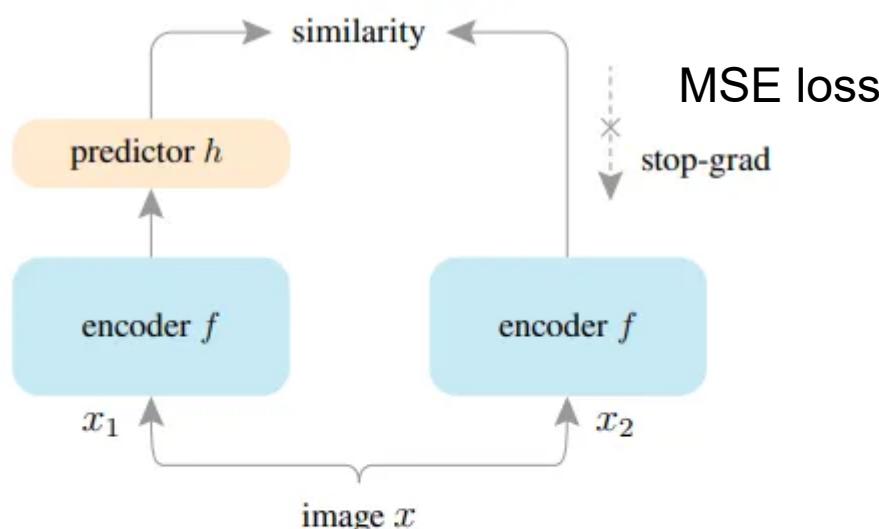
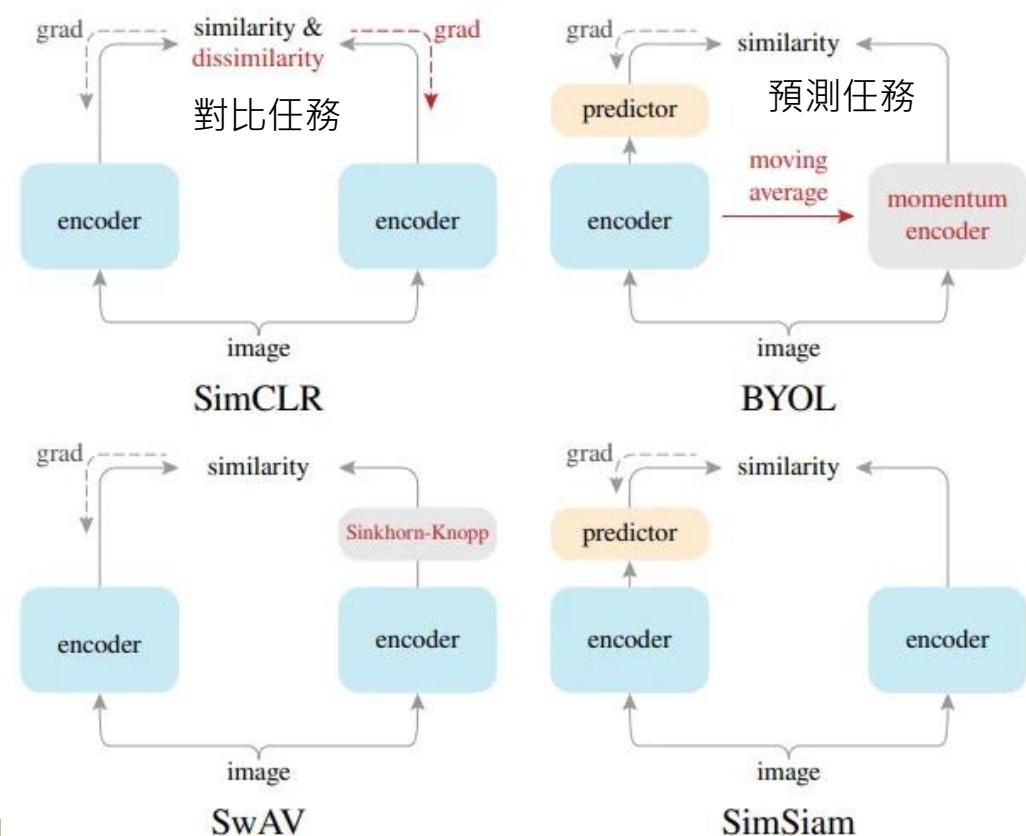


Figure 1. **SimSiam architecture.** Two augmented views of one image are processed by the same encoder network f (a backbone plus a projection MLP). Then a prediction MLP h is applied on one side, and a stop-gradient operation is applied on the other side. The model maximizes the similarity between both sides. It uses neither



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

Table 4. **Comparisons on ImageNet linear classification.** All are based on **ResNet-50** pre-trained with **two 224×224 views**. Evaluation is on a single crop. All competitors are from our reproduction, and “+” denotes *improved* reproduction vs. original papers (see supplement).

pre-train	VOC 07 detection			VOC 07+12 detection			COCO detection			COCO instance seg.		
	AP ₅₀	AP	AP ₇₅	AP ₅₀	AP	AP ₇₅	AP ₅₀	AP	AP ₇₅	AP ₅₀ ^{mask}	AP _{mask}	AP ₇₅ ^{mask}
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

Table 5. **Transfer Learning.** All unsupervised methods are based on 200-epoch pre-training in ImageNet. *VOC 07 detection*: Faster R-CNN [32] fine-tuned in VOC 2007 trainval, evaluated in VOC 2007 test; *VOC 07+12 detection*: Faster R-CNN fine-tuned in VOC 2007 trainval + 2012 train, evaluated in VOC 2007 test; *COCO detection* and *COCO instance segmentation*: Mask R-CNN [18] (1× schedule) fine-tuned in COCO 2017 train, evaluated in COCO 2017 val. All Faster/Mask R-CNN models are with the C4-backbone [13]. All VOC results are the average over 5 trials. **Bold entries** are within 0.5 below the best.

- **Stop gradient 操作的作用:**

- 在 BYOL 和 SimSiam 中，模型會使用兩個編碼器（例如 ResNet-50）來分別處理同一張圖片的不同視角（view）。其中一個編碼器被稱為 online network 或 student network，它的參數會隨著梯度更新而更新；另一個編碼器則被稱為 target network 或 teacher network，它的參數則使用動量更新的方式進行更新，並且在計算損失函數時不進行梯度回傳，這就是 stop gradient 操作的應用之處。

- **Stop gradient 操作的意義:**

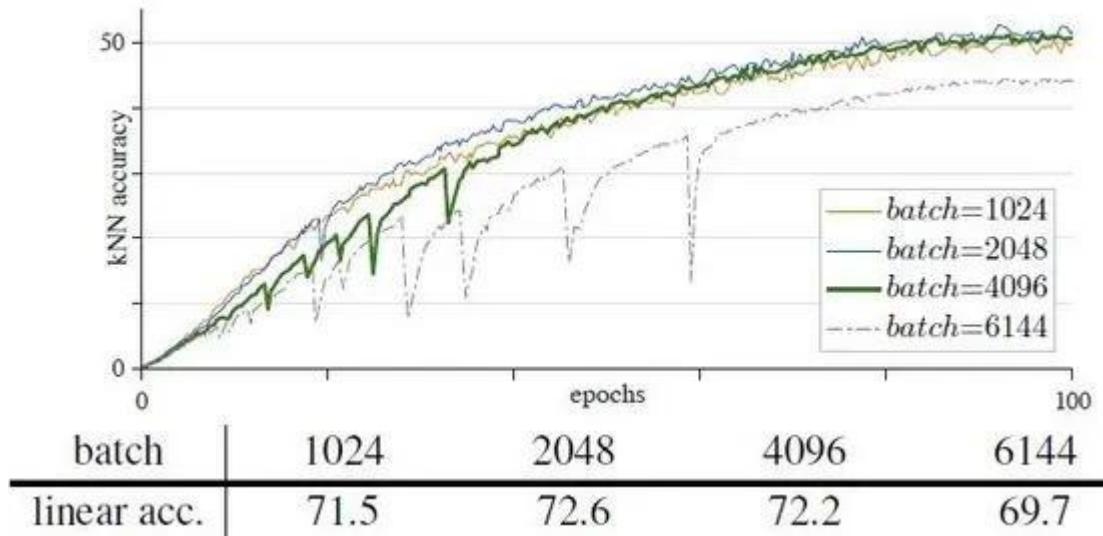
1. **防止模型坍塌:** 如果沒有 stop gradient 操作，模型很容易學到一個捷徑解，也就是對所有輸入都輸出相同的特徵表示。這是因為在沒有負樣本約束的情況下，模型只需要讓所有正樣本的特徵表示盡可能相似就可以最小化損失函數。但是，這種情況下模型並沒有學到任何有用的資訊，因為它無法區分不同的輸入。
2. **實現自蒸餾:** Stop gradient 操作可以將模型的訓練過程人為地分成兩部分，一部分是 student network 的學習過程，另一部分是 teacher network 的更新過程。這種交替更新的方式類似於 EM 演算法，可以說明模型更好地學習到資料中的結構資訊。
3. **簡化模型訓練:** Stop gradient 操作可以簡化模型的訓練過程，因為它不需要像 SimCLR 那樣使用大量的負樣本，也不需要像 MoCo 那樣使用佇列和動量編碼器。

- **Stop gradient 操作的爭議:**

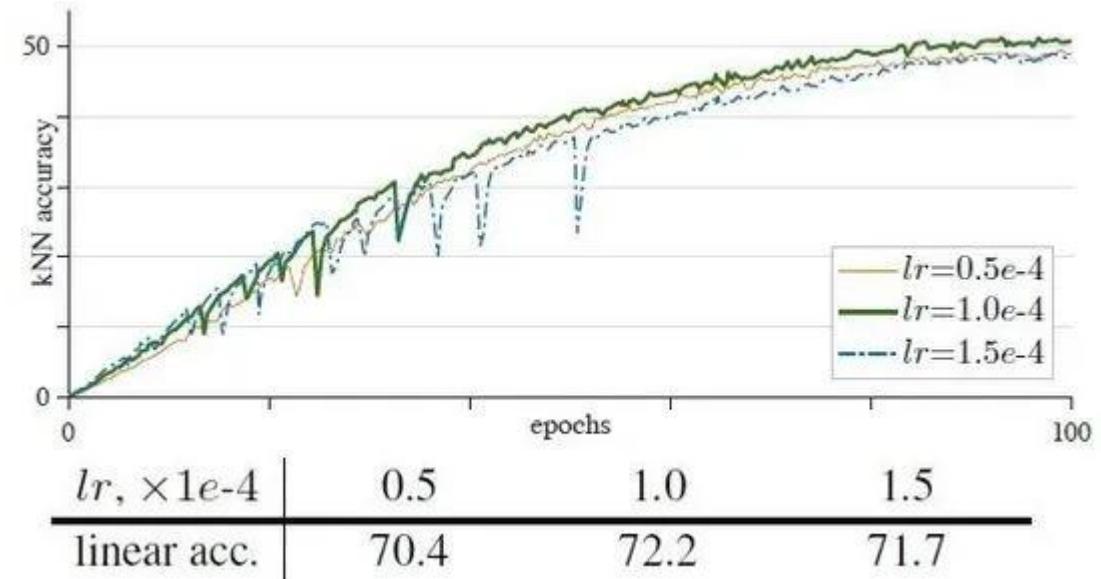
- 一些研究者認為，BYOL 中 stop gradient 操作的成功可能與 batch normalization (BN) 有關。他們認為，BN 在計算批次統計量時會洩露其他樣本的資訊，這相當於提供了一種隱式的負樣本。然而，BYOL 的作者通過進一步的實驗反駁了這個觀點，他們證明了即使在沒有 BN 的情況下，BYOL 仍然可以正常訓練，並取得良好的效果。

MOCO v3 (MOCO v2+BYOL => ViT)

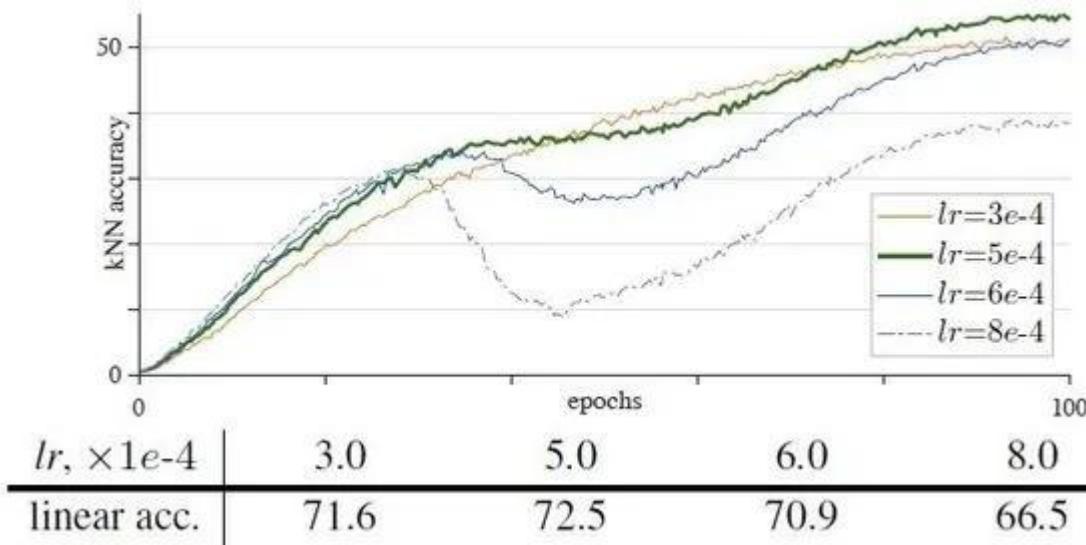
- 當 MOCO v3 的 Encoder = Vision Transformer 時，
 - 訓練會變得不穩定，造成的結果並不是訓練無法收斂，而是 Performance 會輕微的下降



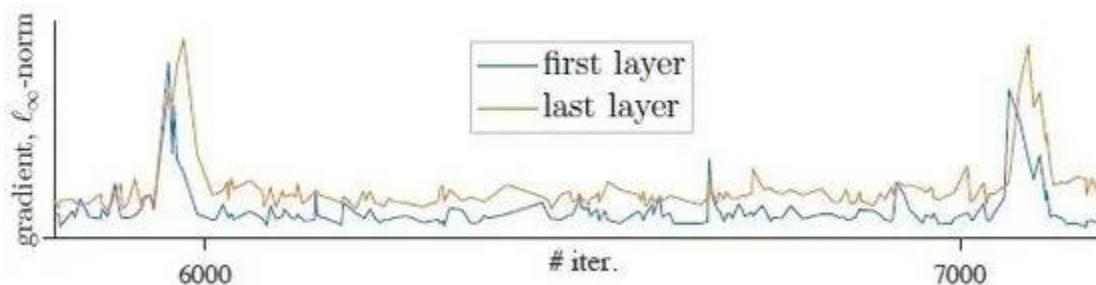
當 Batch Size=4096 時，曲線出現了 Dip 現象。當 Batch Size=6144 時，曲線的 Dip 現象更明顯了。這種不穩定的現象導致精度下降。



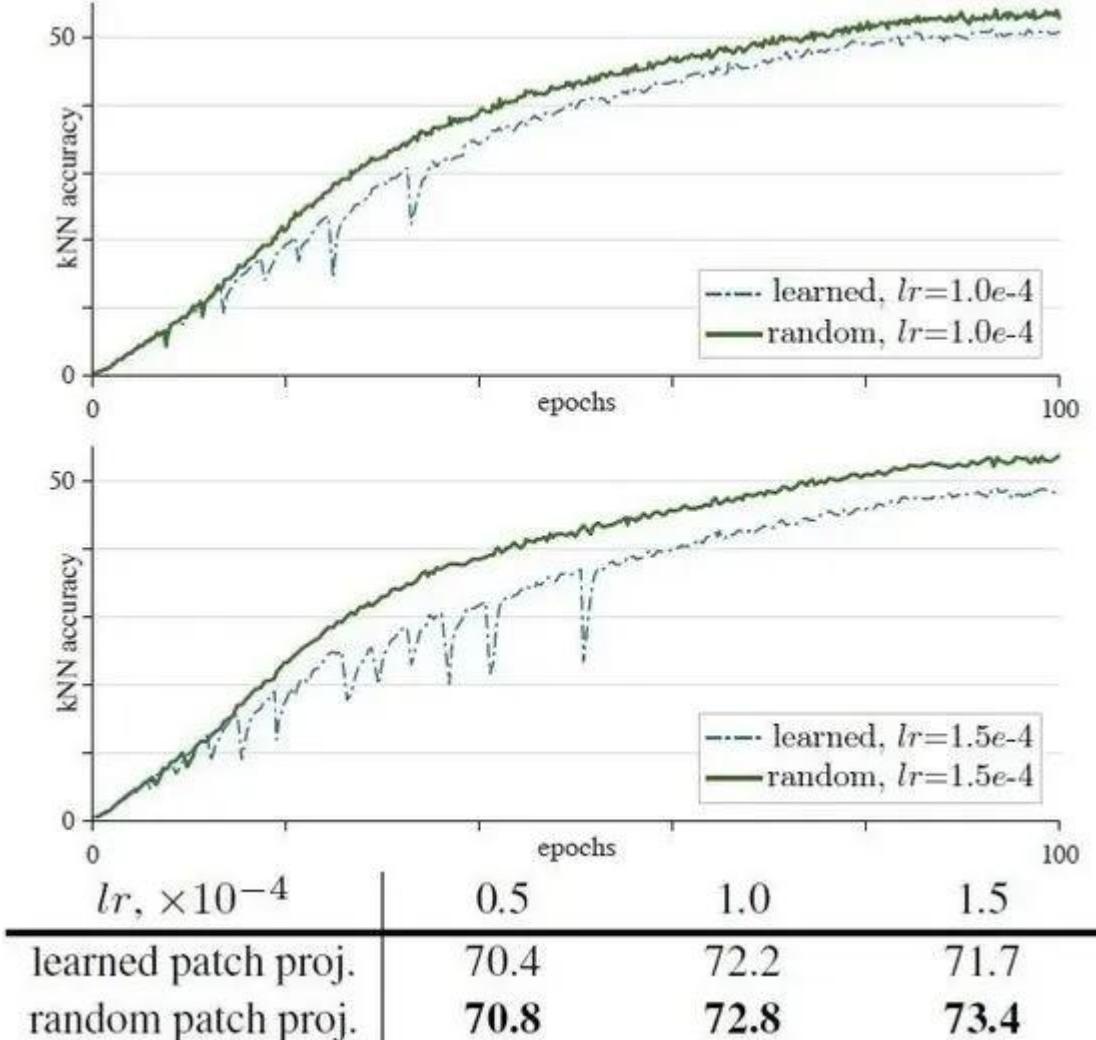
當 Learning Rate = 0.5e-4 時，曲線出現了 Dip 現象。當 Learning Rate = 1.5e-4 時，曲線的 Dip 現象更明顯了。



當 Optimizer = LARS 時當 Learning Rate = 5×10^{-4} 時，
LARS 的 Performance 可以超過 AdamW，而且曲線是平滑的，沒有 Dip 現象。



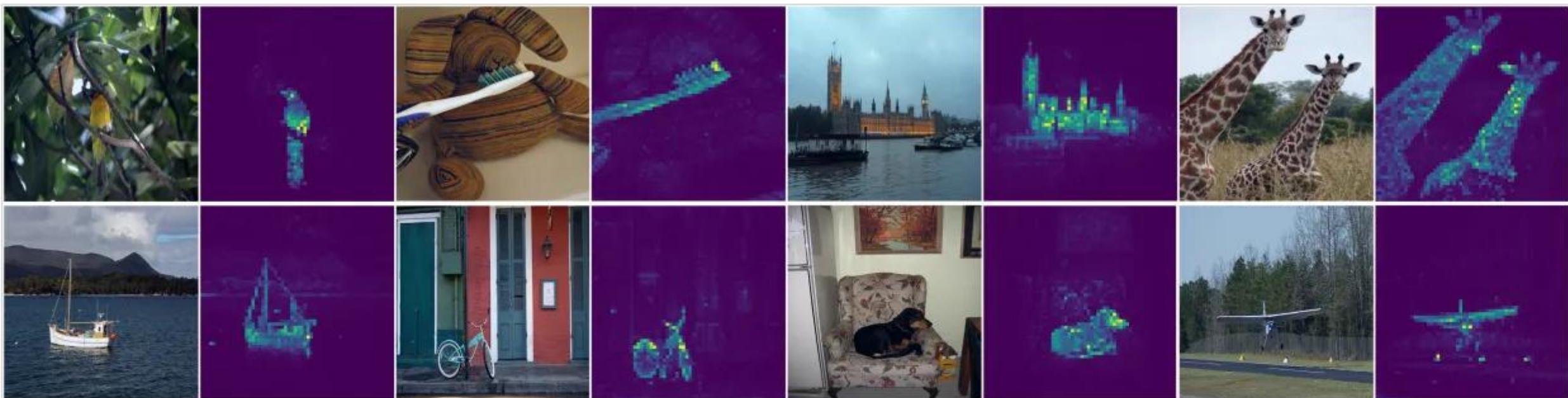
導致訓練出現不穩定的跟第 1 層梯度暴漲有關



凍結 Patch Embedding 層的參數，都能夠提升訓練的穩定性

DINO (Self-distillation with no labels)

- 使用 8x8 patches 的 Vi-T Self-Supervised Learning 的 Attention Map。
 - Self-Supervised Learning，還是可以把注意力都學在這些物體上，模型非常準確的抓住每個物體的輪廓，這個效果甚至可以直接對這個物體作語義分割。



為何後來可以只採用正樣本

- 沒有負樣本
 - (沒有人告訴模型「什麼不是哈士奇」)，模型最簡單的偷懶方法就是**「坍塌 (Collapse)」**：把所有圖片都編碼成一模一樣的向量（例如全部輸出 0 或 1）。這樣不管圖片怎麼變，它們的距離永遠是 0，Loss 完美為 0。
- **BYOL (Bootstrap Your Own Latent)** 和 **SimSiam** 等方法證明了
 - 只要設計巧妙的「不對稱結構 (Asymmetry)」，就不需要負樣本，模型也不會坍塌。
 - 核心魔法在於：**不要讓兩個分支同時更新**。

解析這些方法是如何「騙」過模型，讓它無法偷懶的

- 核心障眼法：Stop Gradient (停止梯度)
 - SimCLR (有負樣本) 中，
 - 兩張圖片通過完全一樣的 Encoder，參數是共享且同時更新的。
- **SimSiam / BYOL (無負樣本)** 中，
 - 引入了一個關鍵機制：**Stop Gradient (停止梯度)**。
 - 路徑 A (Student/Online)：正常計算 Loss，可以回傳梯度更新參數。
 - 路徑 B (Teacher/Target)：雖然也有算特徵，但切斷梯度回傳，參數被視為「常數」。
 - **這改變了遊戲規則**：
 - 原本是：「我們兩個（A 和 B）一起商量，找個最簡單的向量（例如全 0）來讓距離最小。」-> 導致坍塌。
 - 現在變成：「B 暫時不動（被固定住），A 你要負責移動去靠近 B。」
 - 因為 B 是一個隨機初始化的複雜特徵（且不斷在變動），A 為了靠近它，就必須學習去解析圖片的內容，而不能簡單地輸出 0（因為 B 不是 0）。這就逼迫模型真正去學習特徵。

進階機制：BYOL 與 SimSiam 的實作

- 兩種方法雖然都只用正樣本，但防止坍塌的手段略有不同
- **BYOL (Bootstrap Your Own Latent)**
 - 策略：使用 **Momentum Encoder** (動量編碼器) 作為目標。
 - 機制：Teacher 網路的參數不是透過梯度更新的，而是 Student 網路參數的「滑動平均 (Moving Average)」。
 - 效果：Teacher 變得很穩重（更新慢），Student 變得很靈活。Student 拼命追趕 Teacher 的視角。因為 Teacher 一直在變（雖然慢），Student 永遠追不到一個固定的「坍塌點」。

- **SimSiam (Simple Siamese)**

- 策略：更加極簡，連 Momentum Encoder 都不用。直接用同一個網路，但一邊加了 **Stop Gradient**。
- 關鍵組件：**Predictor (預測頭)**。
- 機制：
 - 一邊是 Encoder 輸出 z (視為目標，Stop Gradient)。
 - 另一邊是 Encoder + Predictor 輸出 p 。
 - 任務是讓 p 去預測 z 。
- 為什麼有效？
 - SimSiam 的作者認為這類似於 **EM 算法 (Expectation-Maximization)**。我們把 z 當作暫時的「偽標籤 (Pseudo-label)」，然後優化 p 去擬合它。因為這個過程是不對稱的，模型不會瞬間找到坍塌解。

為什麼只用正樣本更好？

- 如果負樣本那麼有用，為什麼還要費盡心機去掉它？
 - 省去 **Batch Size** 的限制：
 - SimCLR 需要巨大的 Batch Size (4096+) 才能湊夠負樣本。BYOL/SimSiam 對 Batch Size 不敏感，普通的 GPU (Batch Size 64/128) 也能訓練得很好。
 - 避免「錯誤的負樣本」：
 - 在 ImageNet 中，隨機抽的負樣本其實很有可能和正樣本是同一類（例如兩張圖都是狗，但被當作負樣本推開）。只用正樣本就沒有這個邏輯矛盾。
 - 對數據增強 (**Augmentation**) 更魯棒：
 - 實驗發現 BYOL 對圖像顏色的扭曲程度不像 SimCLR 那麼敏感。
- **總結**
 - **SimCLR (有負樣本)**：靠負樣本的**排斥力**撐開特徵空間，防止所有特徵縮成一點。
 - **BYOL/SimSiam (無負樣本)**：靠結構的**不對稱性 (Stop Gradient / Predictor)** 來製造一個「動態的追逐遊戲」。
 - 模型無法偷懶，因為目標 (Target) 總是在變，或者被強制固定不讓動，所以 Student 只能老老實實地去學習如何從圖片中提取特徵來匹配目標。
 - 簡單來說：**有負樣本是「推開異類」**，**無負樣本是「單向追隨」**。兩者都能達到讓模型學會特徵的目的。

Data Centric

Position, 2015

Jigsaw, 2017

Rotation, 2018

Prediction

RNNLM, 1997

word2v, 2013

Cutout, 2015

BERT, 2018

Contrastive

APC, 2019

audio2v, 2019

Mock, 2020

TERA, 2020

Contrastive

InfoNCE, 2017

SimCLR, 2020

MoCov2, 2020

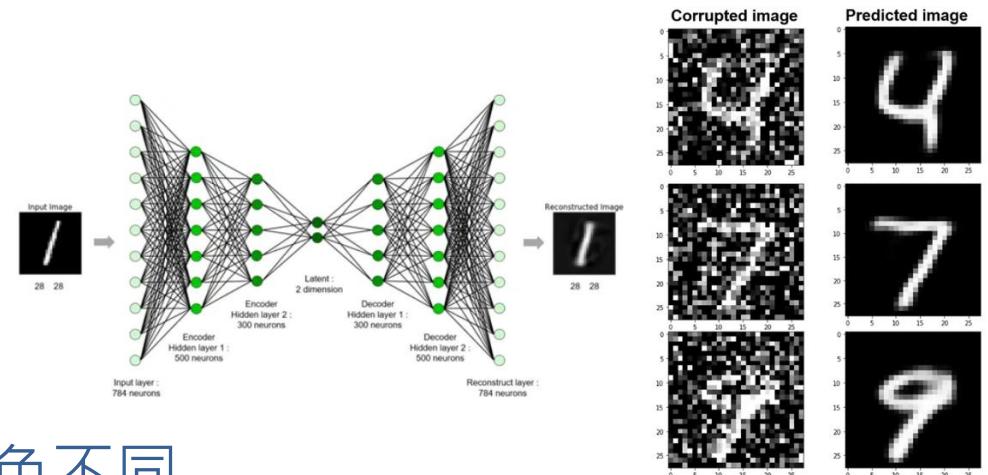
BYOL, 2020

SimSiam, 2020

BERT AND MAE (2021)

Masked Autoencoder (MAE) (CV 版BERT)

- Masked autoencoder
 - 是一種更通用的去噪自編碼器(Denoised Autoencoder)
 - 對NLP (BERT) 效果很好，但vision MAE發展為何還是落後於 NLP?
- 原因
 - CV 和 NLP 主流架構不同
 - ViT前，CV: CNN, NLP: transformer
 - 語言和圖片 (視頻) 的資訊密度不同
 - 語言:高度語意資訊
 - 圖片:空間上是高度冗餘 (可透過內插還原)
 - Decoder部分在 CV 和 NLP 中充當的角色不同
 - CV 領域，Decoder 作用是重建影像，Decoder輸出的語意(訊息)級別是低階的。
 - NLP 領域，Decoder 作用是重建單詞，Decoder輸出的是高階、富含資訊的語義級別



- Large random masked patch

- 這種策略在很大程度上減少了冗餘，並創造了一個具有挑戰性的自監督任務，該任務需要超越低級圖像統計的整體理解

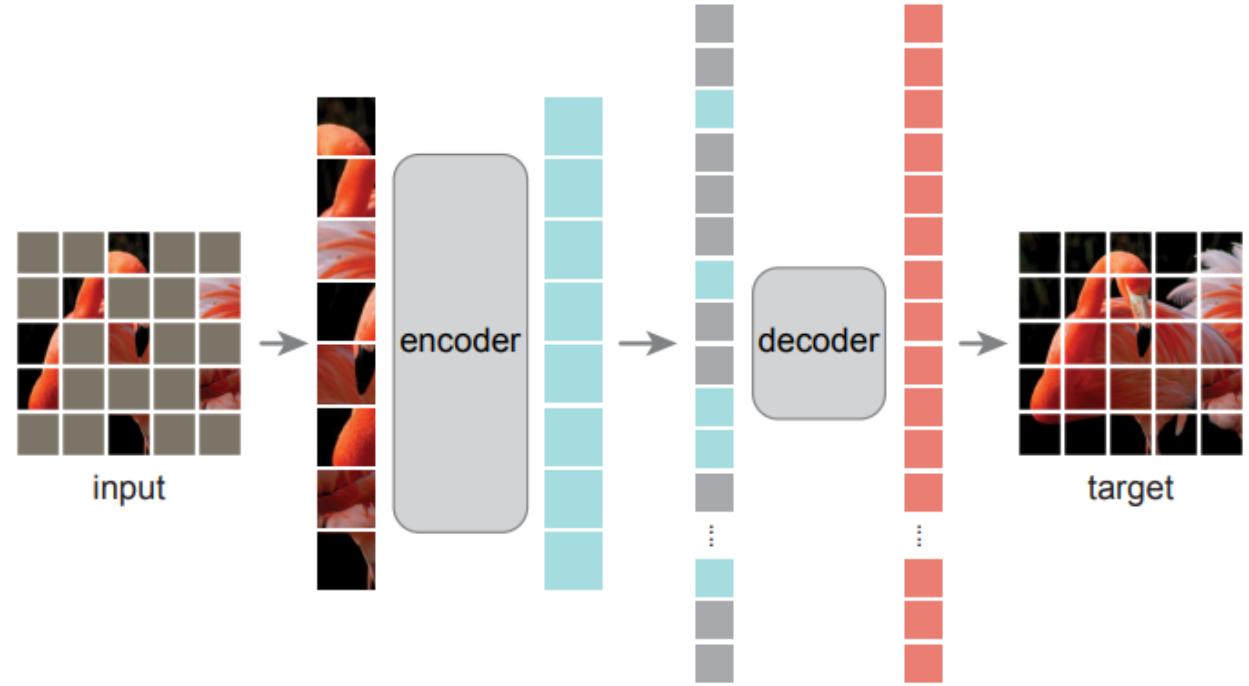
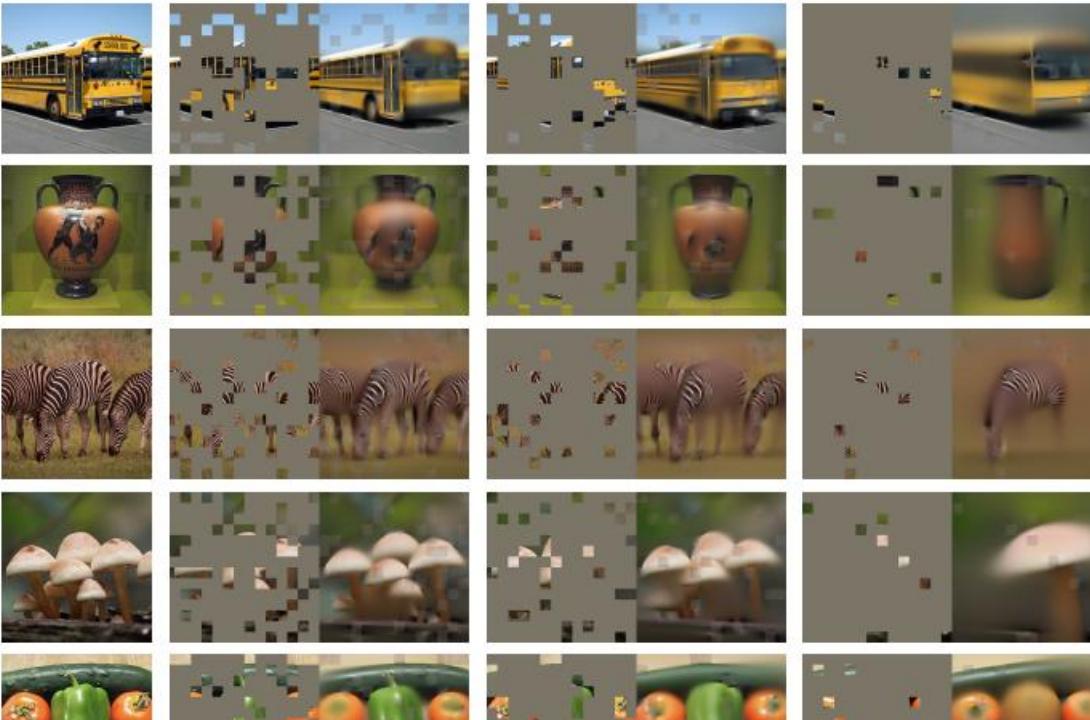


Figure 1. **Our MAE architecture.** During pre-training, a large random subset of image patches (e.g., 75%) is masked out. The encoder is applied to the small subset of *visible patches*. Mask tokens are introduced *after* the encoder, and the full set of encoded patches and mask tokens is processed by a small decoder that reconstructs the original image in pixels. After pre-training, the decoder is discarded and the encoder is applied to uncorrupted images (full sets of patches) for recognition tasks.

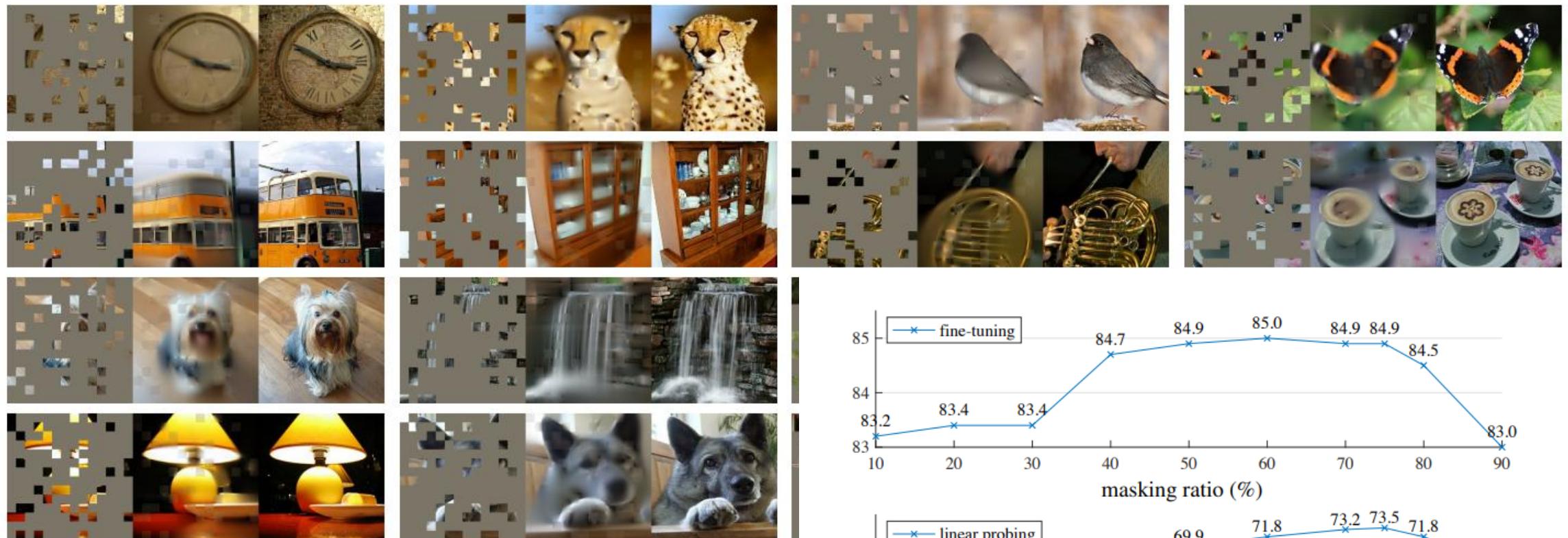


Figure 2. Example results on ImageNet *validation* images. For each image, we show the model’s prediction (middle), and the ground-truth (right). The masking ratio is 80%, leaving 20% of the image visible.

[†]As no loss is computed on visible patches, the model output on visible patches is not used to improve visual quality. We intentionally opt not to do this, so we can

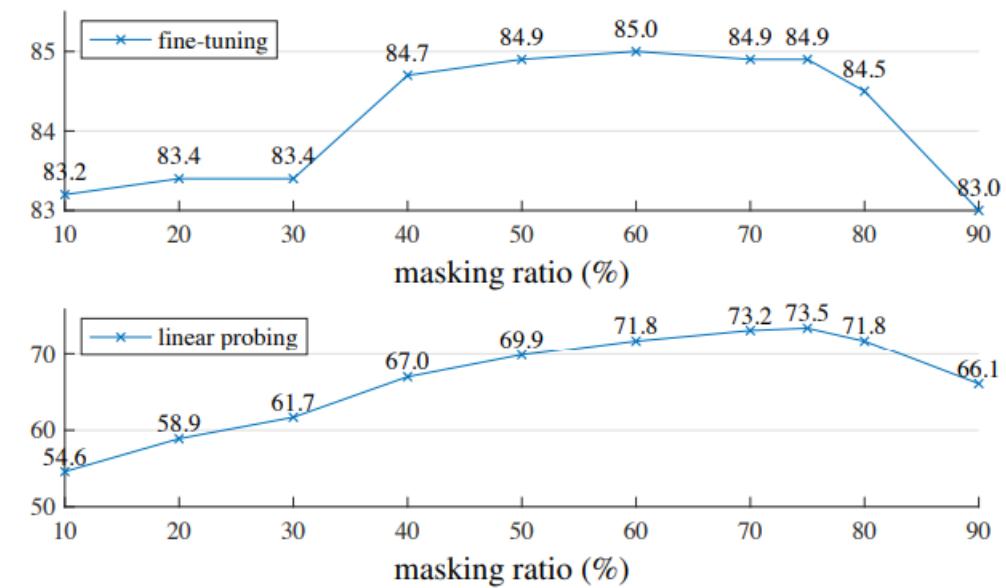


Figure 5. **Masking ratio.** A high masking ratio (75%) works well for both fine-tuning (top) and linear probing (bottom). The y-axes are ImageNet-1K validation accuracy (%) in all plots in this paper.

method	pre-train data	ViT-B	ViT-L	ViT-H	ViT-H ₄₄₈
scratch, our impl.	-	82.3	82.6	83.1	-
DINO [5]	IN1K	82.8	-	-	-
MoCo v3 [9]	IN1K	83.2	84.1	-	-
BEiT [2]	IN1K+DALLE	83.2	85.2	-	-
MAE	IN1K	<u>83.6</u>	<u>85.9</u>	<u>86.9</u>	87.8

Table 3. Comparisons with previous results on ImageNet-1K. The pre-training data is the ImageNet-1K training set (except the tokenizer in BEiT was pre-trained on 250M DALLE data [50]). All self-supervised methods are evaluated by end-to-end fine-tuning. The ViT models are B/16, L/16, H/14 [16]. The best for each column is underlined. All results are on an image size of 224, except for ViT-H with an extra result on 448. Here our MAE reconstructs normalized pixels and is pre-trained for 1600 epochs.

self-supervised pre-training on the ImageNet-1K (IN1K) [13] training set. Then we do supervised training to evaluate the representations with (i) end-to-end fine-tuning or (ii) linear probing

method	pre-train data	AP ^{box}		AP ^{mask}	
		ViT-B	ViT-L	ViT-B	ViT-L
supervised	IN1K w/ labels	47.9	49.3	42.9	43.9
MoCo v3	IN1K	47.9	49.3	42.7	44.0
BEiT	IN1K+DALLE	49.8	53.3	44.4	47.1
MAE	IN1K	50.3	53.3	44.9	47.2

Table 4. COCO object detection and segmentation using a ViT Mask R-CNN baseline. All entries are based on our implementation. Self-supervised entries use IN1K data *without* labels. Mask AP follows a similar trend as box AP.

method	pre-train data	ViT-B	ViT-L
supervised	IN1K w/ labels	47.4	49.9
MoCo v3	IN1K	47.3	49.1
BEiT	IN1K+DALLE	47.1	53.3
MAE	IN1K	48.1	53.6

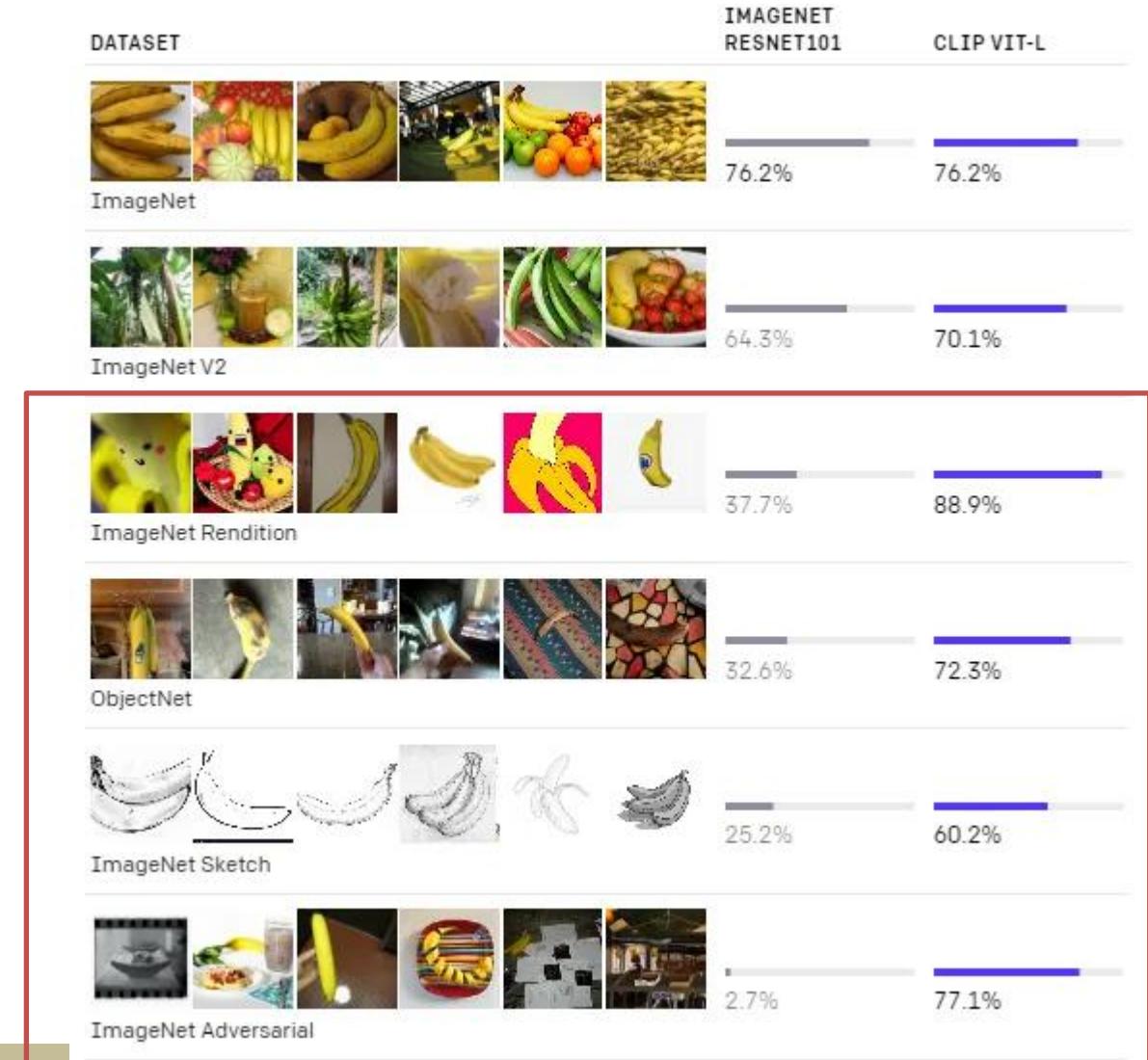
Table 5. ADE20K semantic segmentation (mIoU) using UperNet. BEiT results are reproduced using the official code. Other entries are based on our implementation. Self-supervised entries use IN1K data *without* labels.

MULTIMODEL SELF-SUPERVISED LEARNING

(將的「圖像 VS. 圖像」對比，換成了「圖像 VS. 文本」
對比)

CLIP: Learning Transferable Visual Models From Natural Language Supervision

- 傳統作法問題
 - 訓練和預測通常使用一組固定的預定義物件類別，例如 ImageNet。
 - 難擴展，不夠robust
- How
 - 預訓練任務是在一個包含4億（**圖像，文本**）對的數據集上，從頭開始預測應該學習哪個標題以獲得圖像特徵表示。
 - 預訓練後，自然語言被用來參照學習到的視覺概念（或描述新概念），實現模型對下游任務的zero-shot transfer

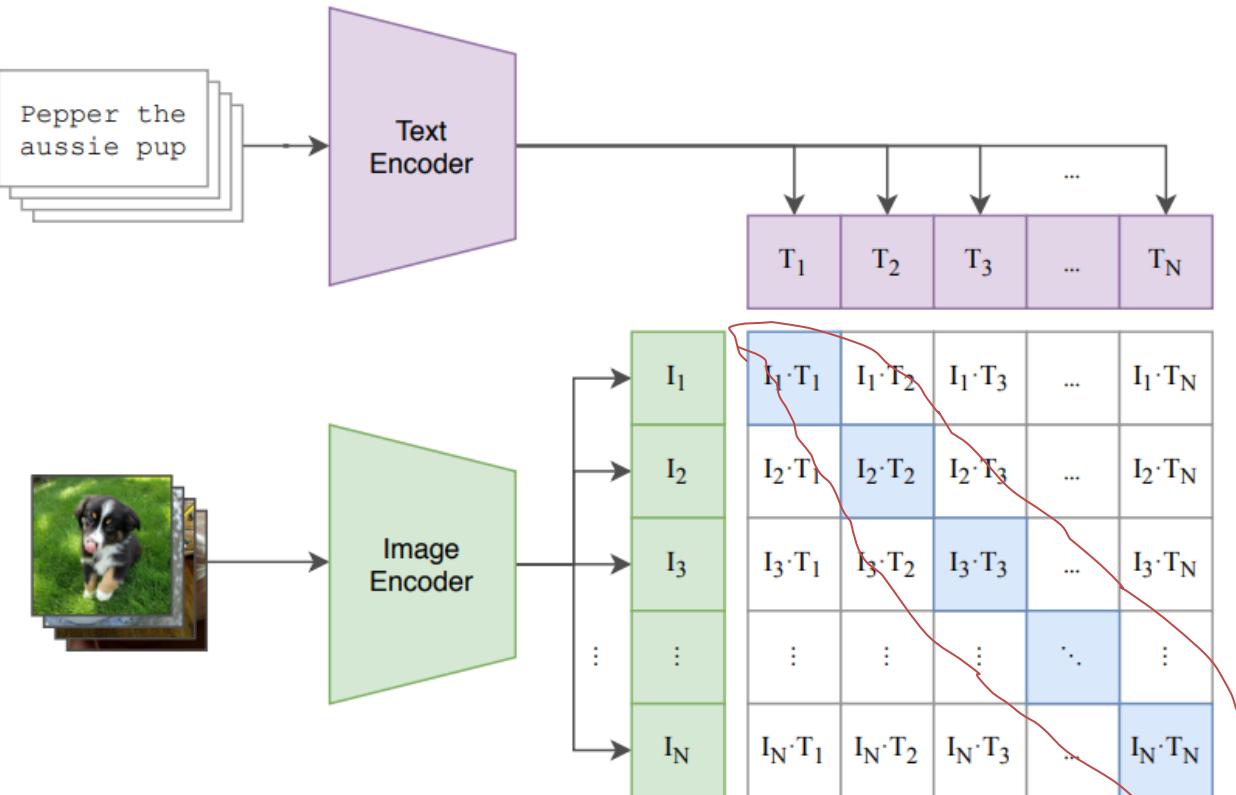


- (text, image) pair, WebImageText (WIT) Dataset
 - 從自然語言中學習相對於標準的眾包標註圖像分類方式更容易擴展，因為它不需要標註。
 - 相較於大多數的無監督或自監督學習方法，它還有一個重要優勢，那就是它不"僅僅"學習一種表示，而且還將該影像特徵表示與語言相連接，這使得靈活的zero-shot轉移成為可能。
- 效果
 - 在 ImageNet 資料集上，CLIP 達到了 76.2% 的準確率。
 - 與使用 ImageNet 訓練資料進行監督式訓練的 ResNet-50 模型的準確率相當，但 CLIP 在此過程中完全沒有使用任何 ImageNet 訓練集中的圖片
 - 但它的準確率仍然落後於目前 ImageNet 上表現最佳的模型，例如 Noisy Student 和 Vision Transformer。這些模型的準確率已經達到 88% 甚至 90% 以上，而 CLIP 僅為 76.2%。作者估計，如果要縮小這個差距，需要將 CLIP 的訓練規模擴大 1000 倍，這對目前的硬體條件來說是難以實現的

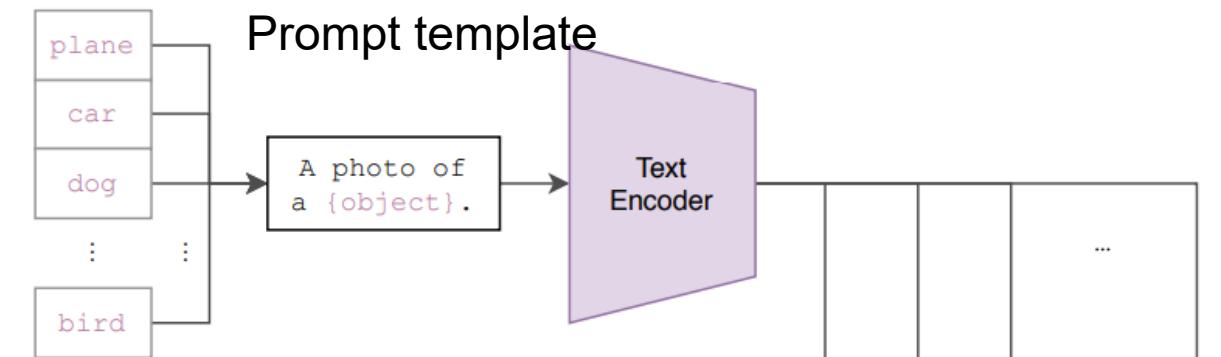
好處

- 擺脫固定類別標籤的限制
 - CLIP 模型則利用自然語言作為監督信號，學習圖像和文本之間的語義關聯，從而擺脫了固定類別標籤的限制，可以靈活地進行 zero-shot 遷移學習，識別任何可以用文本描述的物體類別。
- 強大的 Zero-Shot 遷移能力
 - 在 ImageNet 上達到了 76.2% 的準確率，這與使用 ImageNet 訓練資料進行監督式訓練的 ResNet-50 模型的準確率相當，但 CLIP 完全沒有使用任何 ImageNet 訓練集中的圖片。
- 更接近人類的感知方式:
 - CLIP 模型通過學習圖像和文本之間的語義關聯，使其在某些任務上的表現更接近人類的感知方式。例如，對於 CLIP 模型來說，難分類的類別對人類來說也很難。
- 更高的數據利用效率
- 更強大的泛化能力

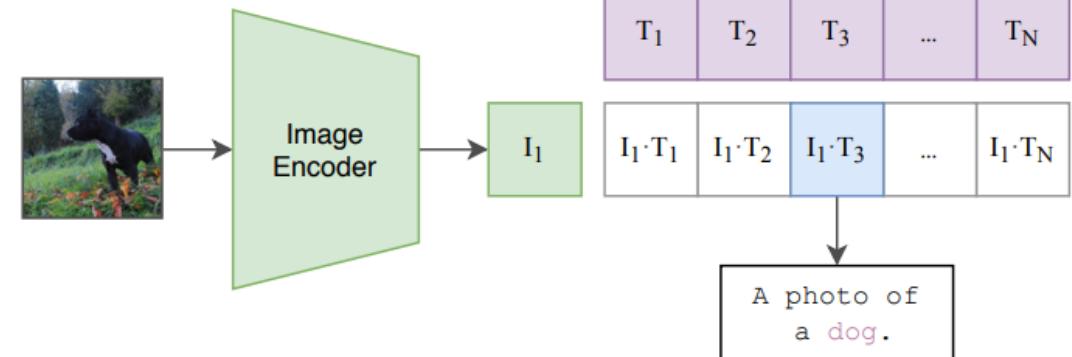
(1) Contrastive pre-training



(2) Create dataset classifier from label text



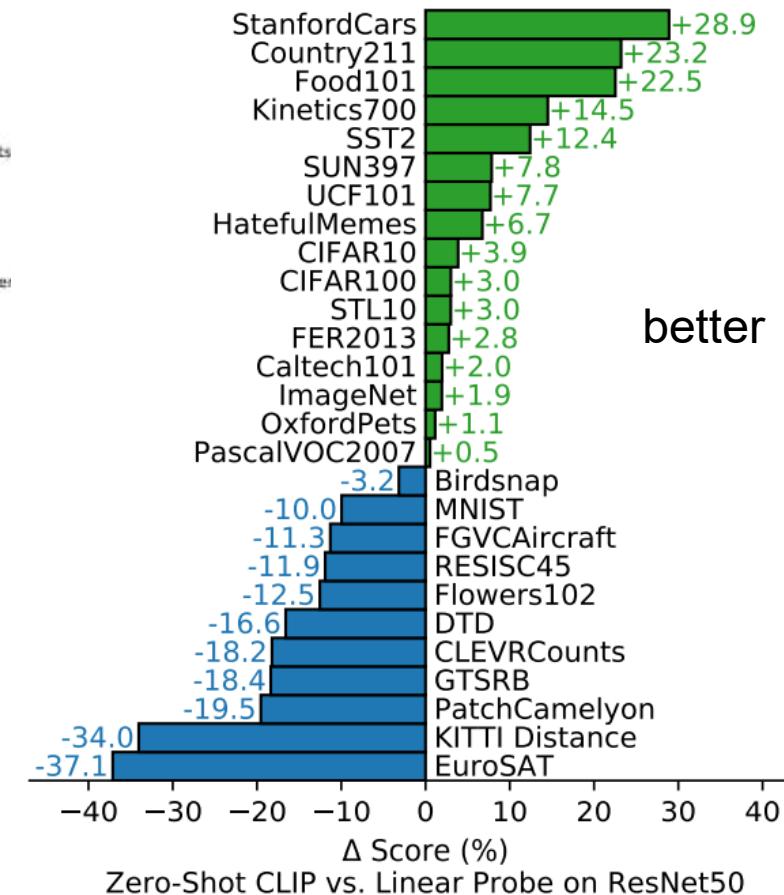
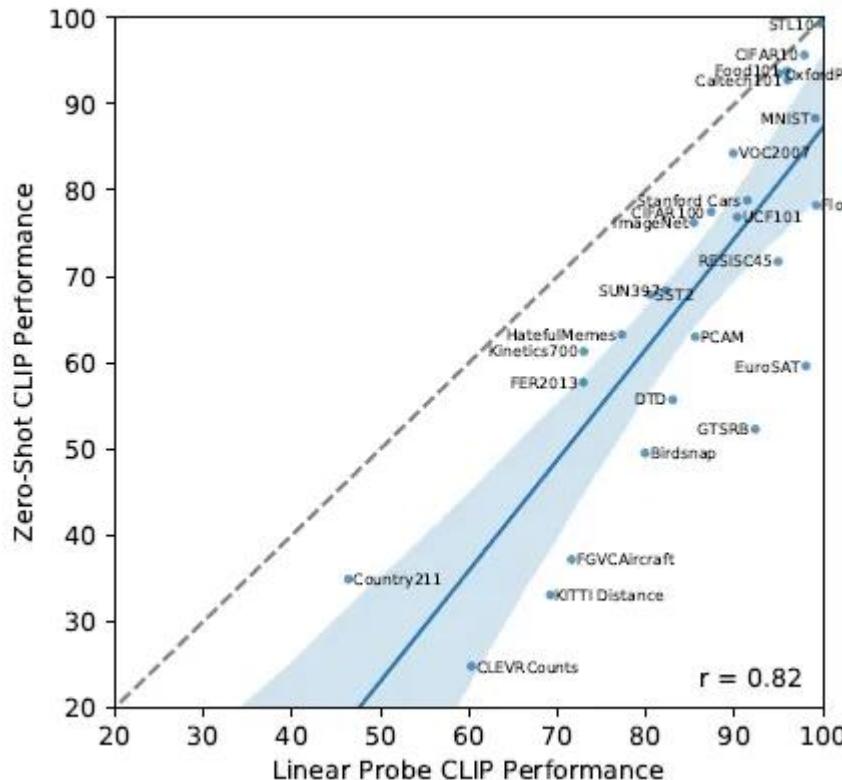
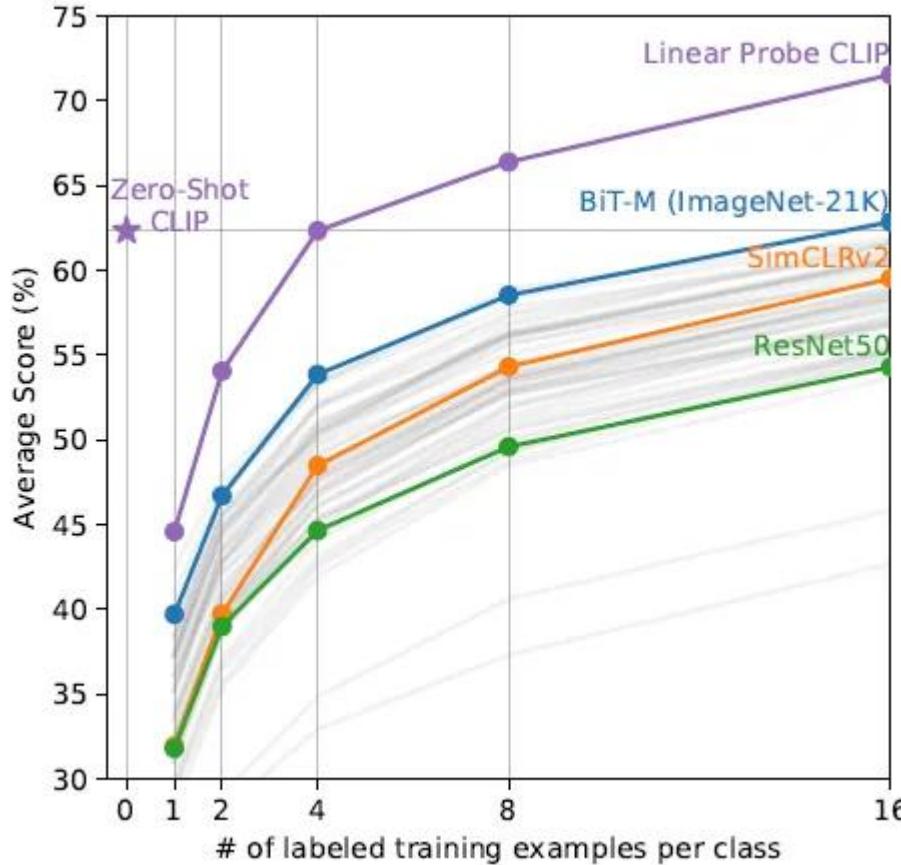
(3) Use for zero-shot prediction



用NLP來做監督訊號

CLIP同時訓練一個圖像編碼器和一個文本編碼器，以預測一批 (image, text) 訓練樣本的正確配對。在測試時，學習後的文本編碼器通過嵌入目標數據集類別的名稱或描述，合成了一個zero-shot線性分類器

- Training Objective
 - The objective of CLIP is to align text embeddings and image embeddings for correct pairs. It achieved this by **maximizing the cosine similarity** between the **correct pair of image and text embeddings** and minimising the cosine similarity between incorrect image and text embeddings.
 - The image and text encoder are trained from scratch without using any pre-weights.
 - The embeddings generated by the image and text encoder are linear projected to match their dimension
 - For a dataset of N (image, text Paris) there are $N \times N$ possible (image, text) pairs. Out of these $N \times N$ pairs N pairs are correct and the rest $N \times N - N$ pairs are incorrect.
 - The model is trained to maximize the cosine similarity of the image and text embeddings of the N correct pairs and minimize the cosine similarity of $N \times N - N$ pairs.
 - Cross entropy loss over similarity score is used as a loss function.
- During inference time
 - the embedding of image and text description of all possible pairs is obtained through the trained image and text encoder and a similarity score is calculated which indicates the relevance of text description concerning image



Zero-shot CLIP 優於少數樣本的線性探測。

Zero-shot 性能與線性探測性能相關，但仍然大多數情況下並非最優。

Zero-shot CLIP is competitive with a fully supervised baseline

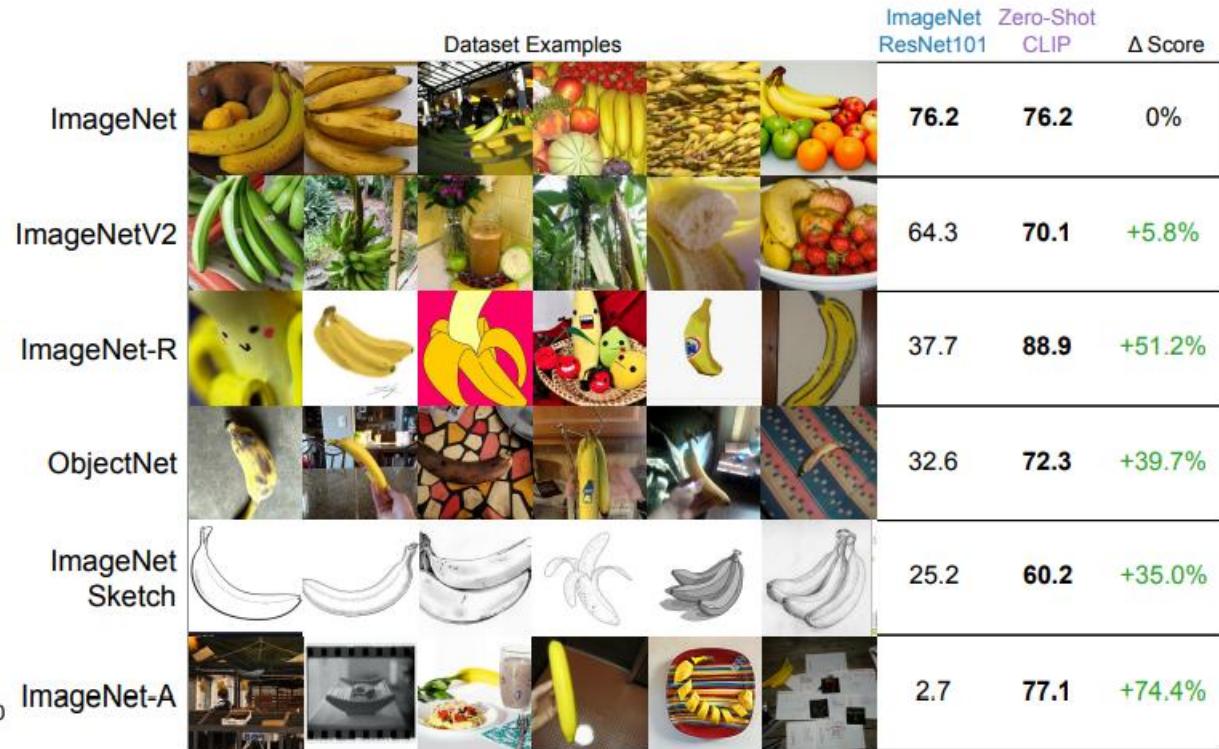
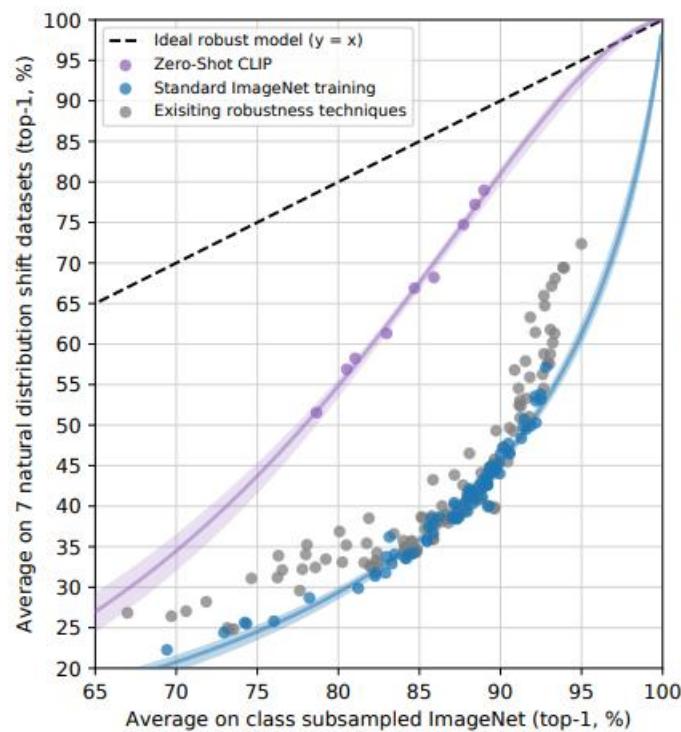


Figure 13. Zero-shot CLIP is much more robust to distribution shift than standard ImageNet models. (Left) An ideal robust model (dashed line) performs equally well on the ImageNet distribution and on other natural image distributions. Zero-shot CLIP models shrink this “robustness gap” by up to 75%. Linear fits on logit transformed values are shown with bootstrap estimated 95% confidence intervals. (Right) Visualizing distribution shift for bananas, a class shared across 5 of the 7 natural distribution shift datasets. The performance of the best zero-shot CLIP model, ViT-L/14@336px, is compared with a model that has the same performance on the ImageNet validation set, ResNet-101.

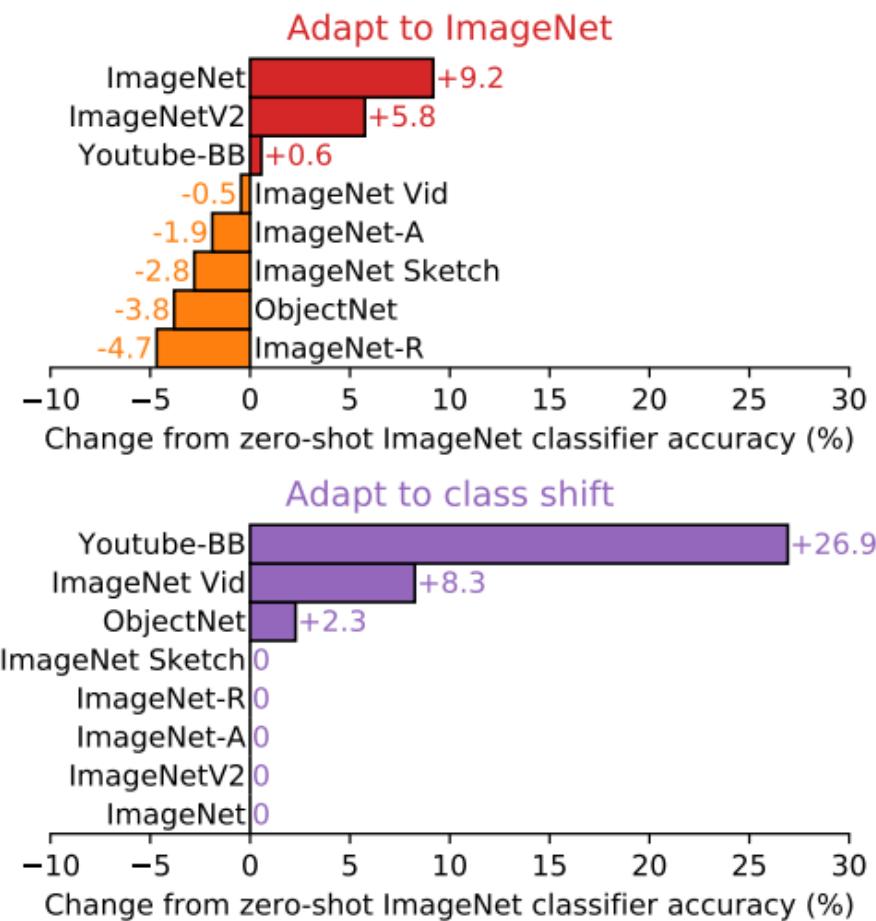
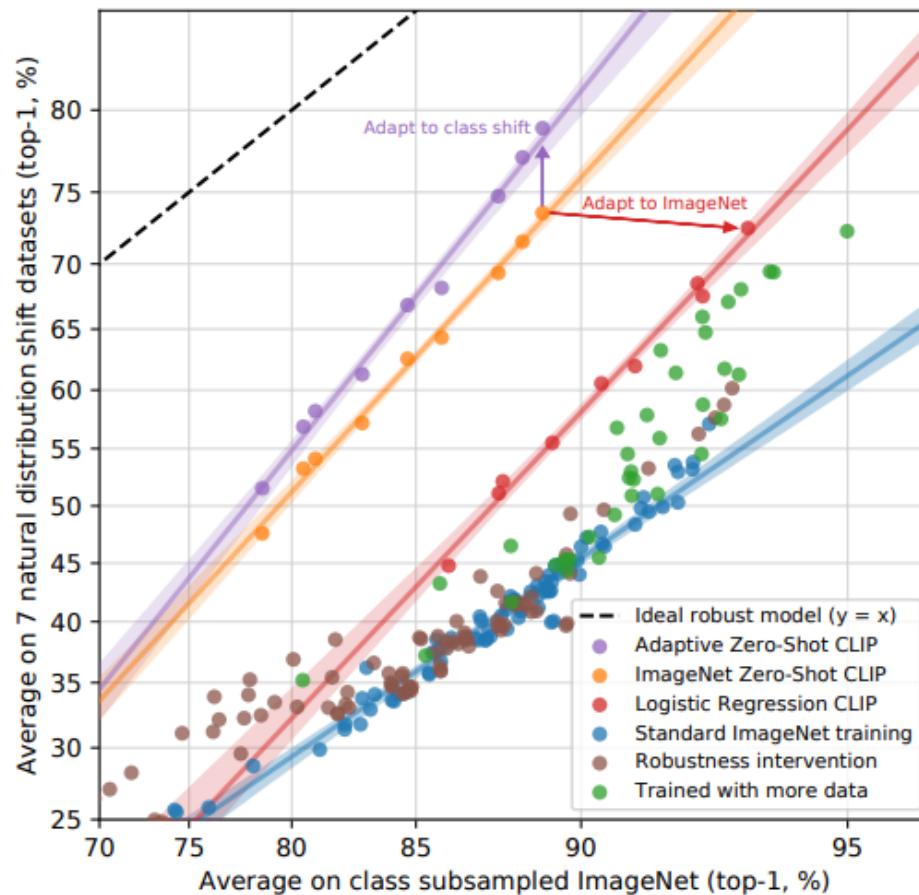


Figure 14. While supervised adaptation to ImageNet increases ImageNet accuracy by 9.2%, it slightly reduces average robustness. (Left) Customizing zero-shot CLIP to each dataset improves robustness compared to using a single static zero-shot ImageNet classifier and pooling predictions across similar classes as in Taori et al. (2020). CLIP models adapted to ImageNet have similar effective robustness as the best prior ImageNet models. (Right) Details of per dataset changes in accuracy for the two robustness interventions. Adapting to ImageNet increases accuracy on ImageNetV2 noticeably but trades off accuracy on several other distributions. Dataset specific zero-shot classifiers can improve accuracy by a large amount but are limited to only a few datasets that include classes which don't perfectly align with ImageNet categories.

Key Applications and Uses of CLIP in Real-World Scenarios

- Image generation model:
 - The SOTA image generation model from text like **Dalle-3** and **Midjourney** uses the CLIP model to generate image embeddings from the input text embeddings to generate images consistent with the text.
- Image Segmentation model:
 - The popular **SAM (Segment anything model)** from meta uses CLIP to understand user prompts and generate segments from images based on user prompt.
- Image Captioning:
 - CLIP is used to get the best caption for the image.

- Content moderation:
 - social media sites employ CLIP to analyze images that can be harmful or do not comply with their policy and filter out them.
- Semantic Retrieval:
 - Text-to-image or image text searches are possible with CLIP embeddings.
- Image Search:
 - CLIP can be utilized to find images corresponding to a specific text query.
- Visual Question Answering (VQA):
 - CLIP has the ability to respond to queries regarding the visual elements of an image using a given text input.

為什麼 CLIP 很重要？

- CLIP 改變了遊戲規則，主要體現在兩點：
- **語義理解 (Semantic Understanding)**：
 - SimCLR 知道兩張狗的照片「很像」，但它不知道那是「狗」。CLIP 知道那是「狗」，甚至知道那是「黃金獵犬」，因為文字編碼器賦予了圖像語義。
- **生成模型的導航員**：
 - 在 Stable Diffusion 或 DALL-E 中，CLIP 扮演了「評審」的角色。生成器畫出一張圖，CLIP 負責判斷：「這張圖跟用戶輸入的文字描述像不像？」這使得 Text-to-Image 生成成為可能。

Segment Anything (SAM)

Universal segmentation model

