Representation learning

Deep Learning @ HSE

Plan

- Examples
- Differences to other tasks
- Classification
- Metric learning
- Self-supervised learning
- CLIP

Retrieval using learned representations

[Krizhevsky et al. NIPS12]

Verification problems in vision

Key question: do two photos show the same object/subject? (*verification)*

Face recognition datasets (e.g. *MSRA-CF*):

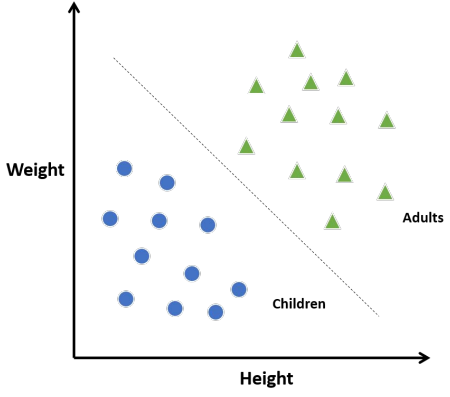
Re-identification datasets (e.g. *ViPER*):

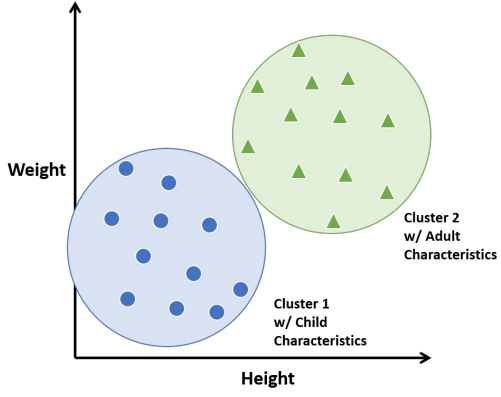
Class is important

Examples

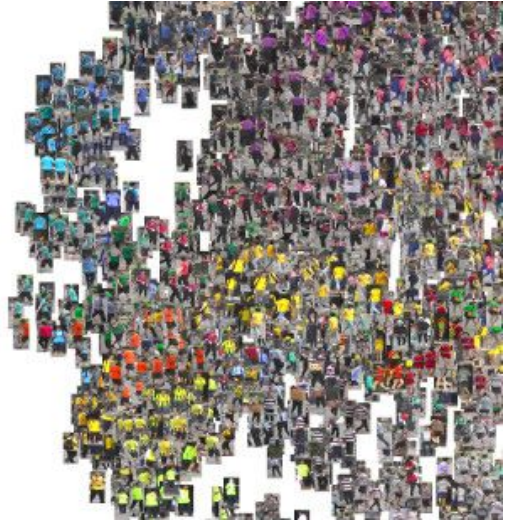
- Retrieve similar to request
- few-shot classification
  - Rare classes
  - Enhance time-to-market
  - Non-DS can use as wrapped instrument
- Build strong embedder
- Identification
  - Faces
  - Logos
  - Calligraphy
- Re-Identification

Classification vs Clustering





Classification vs Clustering



Verification vs Classification

Key question: do two photos show the same object/subject? (*verification)*

- System must be able to handle unseen "classes"
- During training classes can be numerous, small-sized, imbalanced, etc.

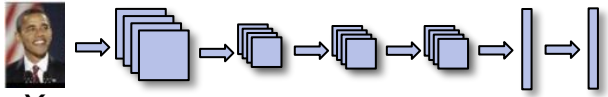
Datasets

Many classes, few objects pre class

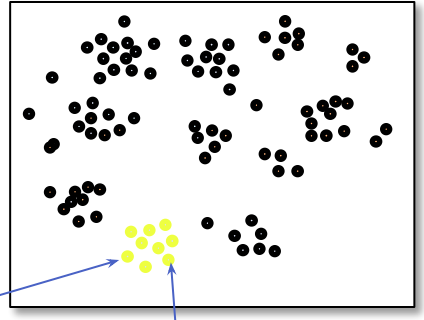
- Omniglot
- Person ReID
- LFW

Datasets. ReID

Verification as embedding learning

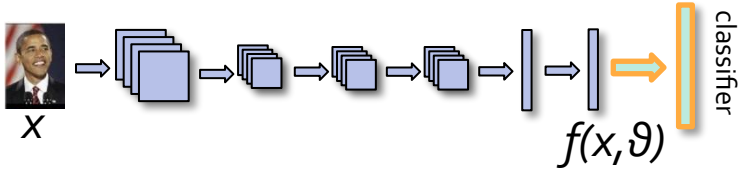


Semantic space:



NB: always normalize your descriptors!

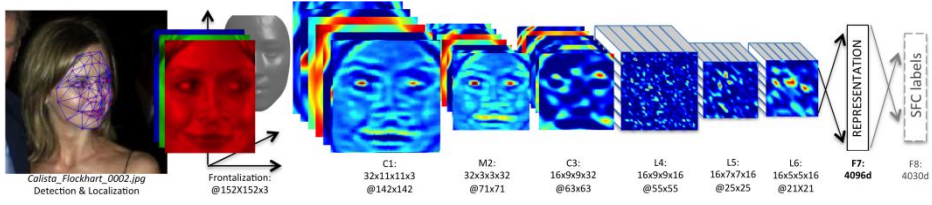
Approach 1: classification-based



- Same idea as "Train on ImageNet, use for retrieval"
- The bigger the classification dataset, the better is the performance
- Training-time classes can be seen as prototypes for test-time classes

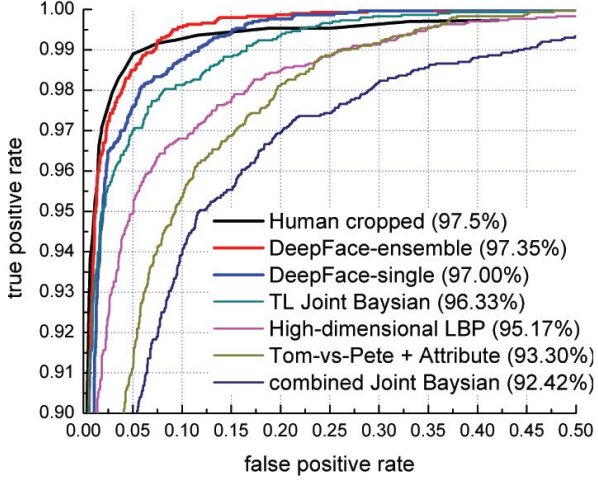
Face verification: “Deep face”

[Taigman et al. 2014]



- *Classification* network trained on 4030 people x ~1000 images.
- Target problem: *verification* (same vs different)

Face verification: “Deep face”



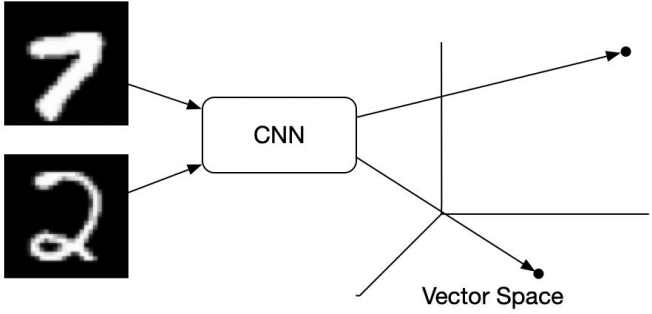
Different ConvNets combined using SVM-learned weights on validation set

[Taigman et al. 2014]

From classification to metric learning

- Classification requires class labels
- Metric learning requires only same-different labels
- Classification dataset can be trivially converted to metric learning dataset

Contrastive Loss



Contrastive Loss

$$y_{\text{pred}} = \text{dist}(x_i, x_j)$$

$$y_{\text{target}} = \begin{cases} 1, \text{ if } c_i == c_j, \\ 0, \text{otherwise} \end{cases}$$

Contrastive Loss

$$y_{\text{pred}} = \text{dist}(x_i, x_j)$$

$$y_{\text{target}} = \begin{cases} 1, \text{ if } c_i == c_j, \\ 0, \text{otherwise} \end{cases}$$

$$L = y_{\text{target}} y_{\text{pred}} - (1 - y_{\text{target}}) y_{\text{pred}}$$

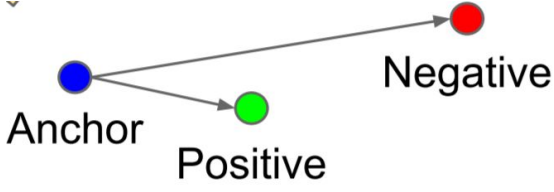
Contrastive Loss

$$y_{\text{pred}} = \text{dist}(x_i, x_j)$$

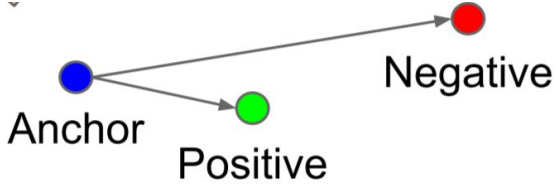
$$y_{\text{target}} = \begin{cases} 1, \text{ if } c_i == c_j, \\ 0, \text{otherwise} \end{cases}$$

$$L = y_{\text{target}} y_{\text{pred}} + (1 - y_{\text{target}}) \max(0, m - y_{\text{pred}})$$

Triplet Loss

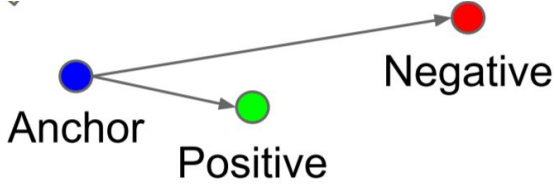


Triplet Loss



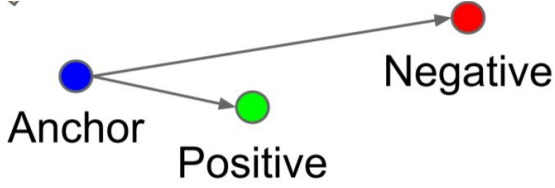
$$L = \text{dist}(x_a, x_p) - \text{dist}(x_a, x_n)$$

Triplet Loss



$$L = \text{dist}(x_a, x_p) + \max(0, m - \text{dist}(x_a, x_n))$$

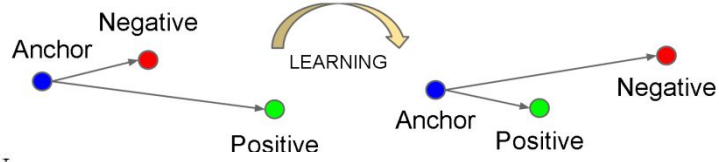
Triplet Loss + Classification



$$L = L_{\text{triplet}} + L_{\text{CE}}$$

Google “FaceNet”

[Schroff et al. CVPR15]



Simple **triplet loss**: $$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$$

- Use large mini-batches (1800, 40 images for several classes + lots of random)
- Take all positives from the batch
- Mine “*semi-hard*” negatives

$$\|f(x_i^a) - f(x_i^p)\|_2^2 < \|f(x_i^a) - f(x_i^n)\|_2^2$$

Google “FaceNet” results

[Schroff et al. CVPR15]

- Upto 99.63% on LFW (human is ~97%)

Performance vs training data:

#images	VAL
2.6M	76.3%
26M	85.1%
52M	85.1%
260M	86.2%

Sampling

- All Triplet
- Hard Triplet
- Hard Cluster Triplet
- And others

All Sampling

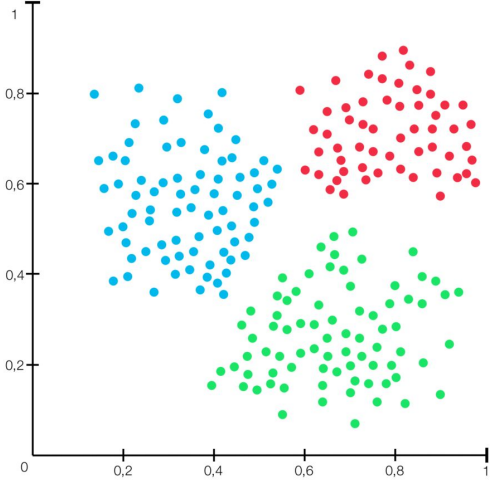
$$\mathcal{L}_{\text{BA}}(\theta; X) = \overbrace{\sum_{i=1}^{P}\sum_{a=1}^{K}}^{\text{all anchors}} \overbrace{\sum_{\substack{p=1 \\ p\neq a}}^{K}}^{\text{all pos.}} \overbrace{\sum_{j=1}^{P}\sum_{\substack{n=1 \\ j\neq i}}^{K}}^{\text{all negatives}} \left[m + d^{i,a,p}_{j,a,n} \right]_+, \quad (6)$$

$$d^{i,a,p}_{j,a,n} = D\left(f_\theta(x^i_a), f_\theta(x^i_p)\right) - D\left(f_\theta(x^i_a), f_\theta(x^j_n)\right).$$

Hard Sampling

$$\mathcal{L}_{\text{BH}}(\theta; X) = \overbrace{\sum_{i=1}^{P} \sum_{a=1}^{K}}^{\text{all anchors}} \Big[m + \overbrace{\max_{p=1\ldots K} D\left(f_\theta(x_a^i), f_\theta(x_p^i)\right)}^{\text{hardest positive}} - \underbrace{\min_{\substack{j=1\ldots P \\ n=1\ldots K \\ j \neq i}} D\left(f_\theta(x_a^i), f_\theta(x_n^j)\right)}_{\text{hardest negative}} \Big]_+, \quad (5)$$

Hard Cluster Sampling



$$d_i^{inter} = \min_{\forall i_d \in P, i_d \neq i} ||f_i^m - f_{i_d}^m||_2^2$$

$$d_i^{intra} = \max_K ||f(x) - f_i^m||_2^2$$

$$Lb_c = \sum_i^P \max((d_i^{intra} - d_i^{inter} + \alpha), 0)$$

Self-supervised representation learning

- (Pre) trains representations without labels
- Especially useful for new domains with few labels (e.g. medical, robotics)
- Meta-idea 1: bottleneck learning (reduction to compression)
- Meta-idea 2: predictive learning (reduction to classification)
- Meta-idea 3: reduction to metric learning

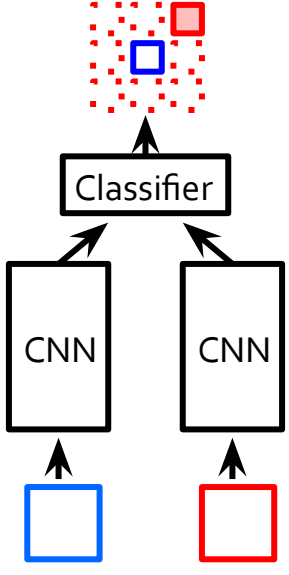
Self-supervised representation learning

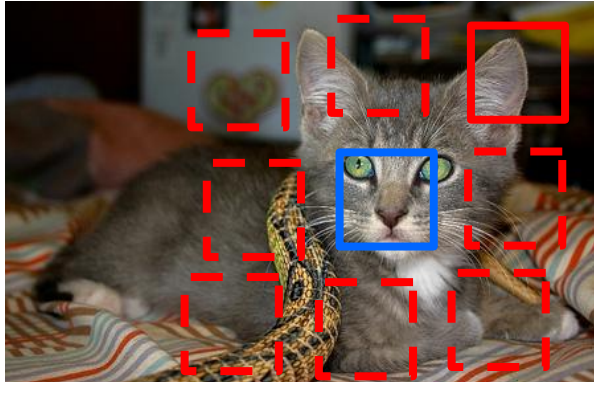
- (Pre) trains representations without labels
- Especially useful for new domains with few labels (e.g. medical, robotics)
- Meta-idea 1: bottleneck learning (reduction to compression)
- **Meta-idea 2: predictive learning (reduction to classification)**
- Meta-idea 3: reduction to metric learning

Predictive self-supervised learning

- Given an element predict *nearby* elements (e.g. next, previous, adjacent, etc.)
- Does not require annotated data ("self-supervised")
- Particularly prominent in NLP (e.g. BERT), but also applicable to computer vision

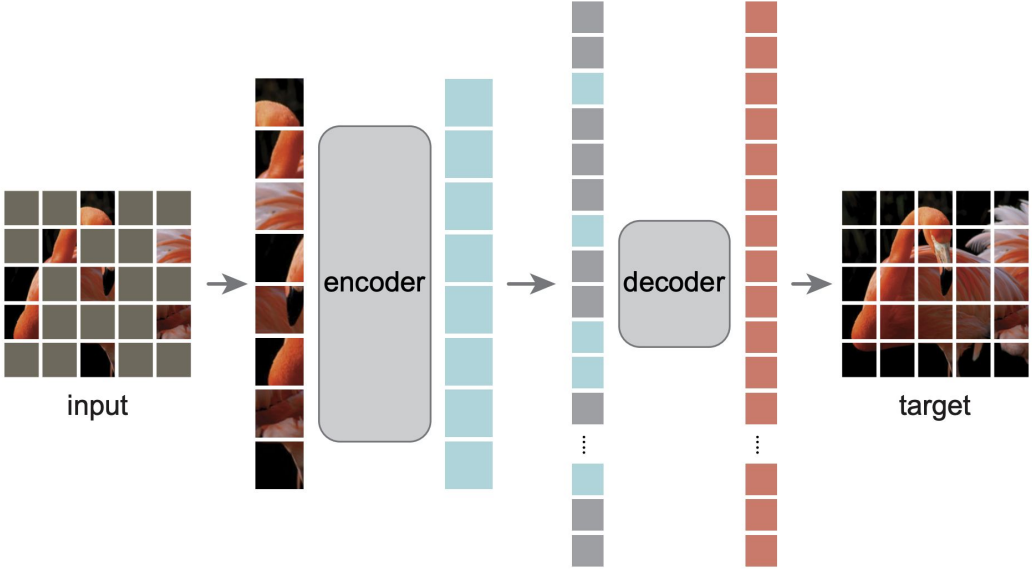
Predictive learning for still images



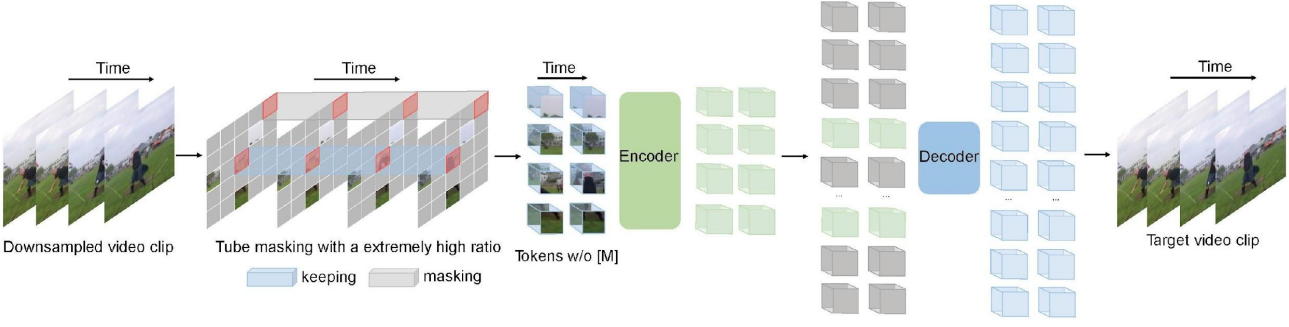


[Doersch et al. ICCV15]

Predictive learning for still images



Predictive learning for still images



Self-supervised representation learning

- (Pre) trains representations without labels
- Especially useful for new domains with few labels (e.g. medical, robotics)
- Meta-idea 1: bottleneck learning (reduction to compression)
- Meta-idea 2: predictive learning (reduction to classification)
- **Meta-idea 3: reduction to metric learning**

SimCLR self-supervised learning

(a) Original (b) Crop and resize (c) Crop, resize (and flip) (d) Color distort. (drop) (e) Color distort. (jitter)

- Each batch consists of multiple pairs of matching images
- Each pair is the two copies of the same image with varying crop, color distortion, Gaussian blur
- We want the network to map each pair to two nearby points that are separated from remaining images
- Loss:

$$\ell_{i,j}^{\text{NT-Xent}} = -\log \frac{\exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)}$$

- Requires large enough batch to work well

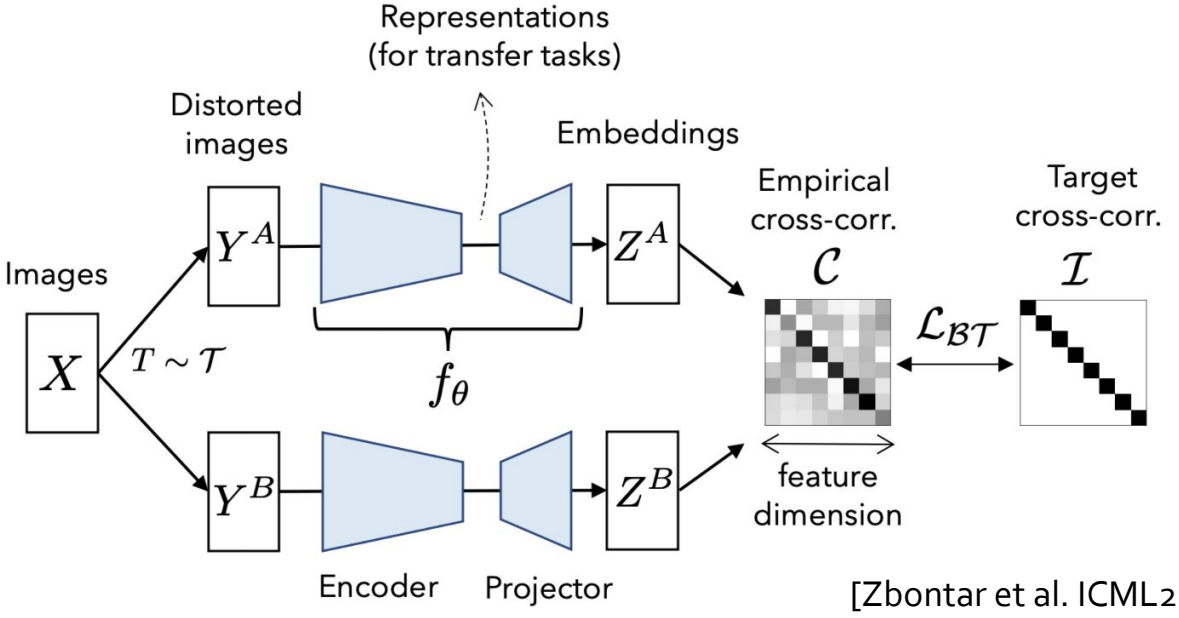
[Chen et al. ICML20] [Chen et al. NeurIPS20]

SimCLR self-supervised learning

https://ai.googleblog.com/2020/04/advancing-self-supervised-and-semi.html

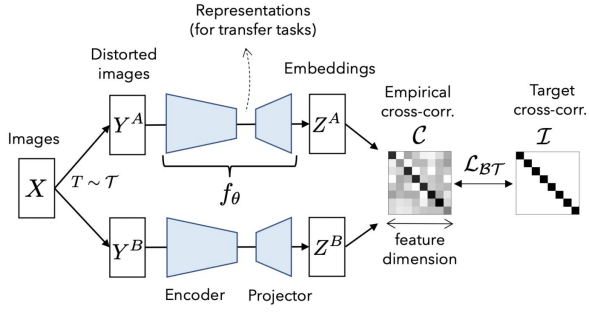
[Chen et al. ICML20]
[Chen et al. NeurIPS20]

Barlow Twins



[Zbontar et al. ICML21]

Barlow Twins

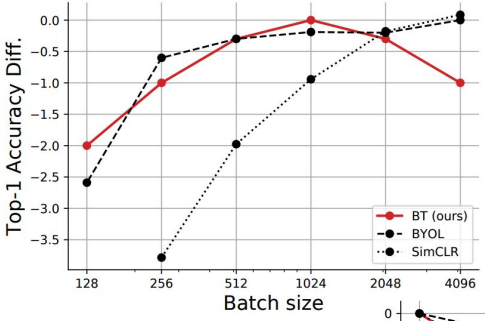


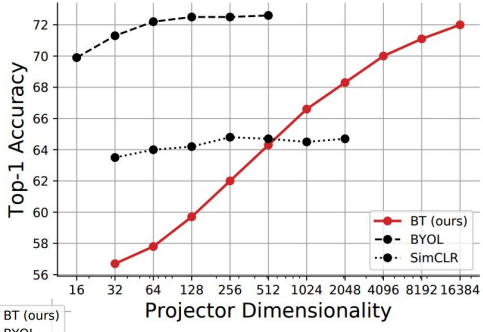
$$\mathcal{L}_{\mathcal{BT}} \triangleq \underbrace{\sum_i (1 - \mathcal{C}_{ii})^2}_{\text{invariance term}} + \lambda \underbrace{\sum_i \sum_{j \neq i} \mathcal{C}_{ij}{}^2}_{\text{redundancy reduction term}}$$

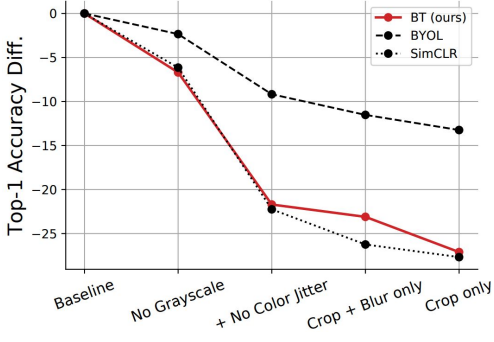
$$\mathcal{C}_{ij} \triangleq \frac{\sum_b z^A_{b,i} z^B_{b,j}}{\sqrt{\sum_b \left(z^A_{b,i}\right)^2} \sqrt{\sum_b \left(z^B_{b,j}\right)^2}}$$

[Zbontar et al. ICML21]

Barlow Twins







[Zbontar et al. ICML21]

Barlow Twins

Linear classification:

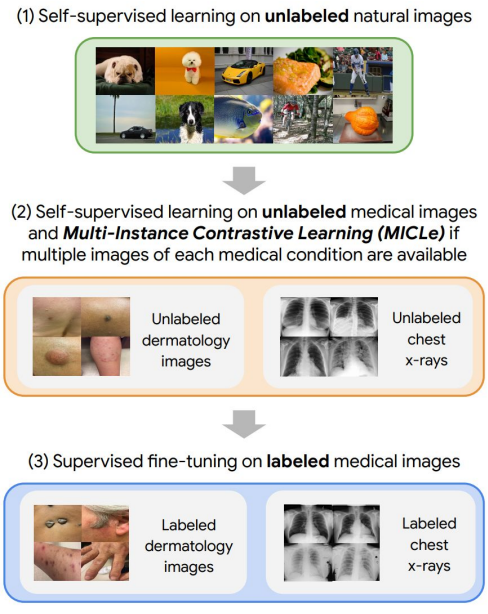
Method	Places-205	VOC07	iNat18
Supervised	53.2	87.5	46.7
SimCLR	52.5	85.5	37.2
MoCo-v2	51.8	86.4	38.6
SwAV (w/o multi-crop)	52.8	86.4	39.5
SwAV	56.7	88.9	48.6
BYOL	54.0	86.6	47.6
BARLOW TWINS (ours)	54.1	86.2	46.5

Detection / instance segmentation:

Method	VOC07+12 det			COCO det			COCO instance seg		
	AP_{all}	AP_{50}	AP_{75}	AP^{bb}	AP^{bb}_{50}	AP^{bb}_{75}	AP^{mk}	AP^{mk}_{50}	AP^{mk}_{75}
Sup.	53.5	81.3	58.8	38.2	58.2	41.2	33.3	54.7	35.2
MoCo-v2	**57.4**	82.5	**64.0**	**39.3**	58.9	**42.5**	**34.4**	55.8	36.5
SwAV	56.1	**82.6**	62.7	38.4	58.6	41.3	33.8	55.2	35.9
SimSiam	57	82.4	63.7	39.2	**59.3**	42.1	**34.4**	**56.0**	**36.7**
BT (ours)	56.8	**82.6**	63.4	39.2	59.0	**42.5**	34.3	**56.0**	36.5

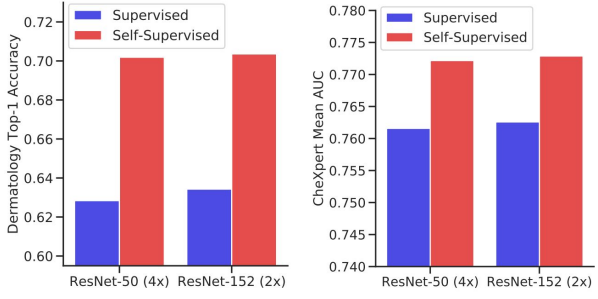
[Zbontar et al. ICML21]

Medical imaging case



15 000 train cases (1-6 images each), 454 000 unlabeled, 27 classes

67 000 train images, 112 000 unlabeled, 5 labels



Blue = pretrain on ImNet

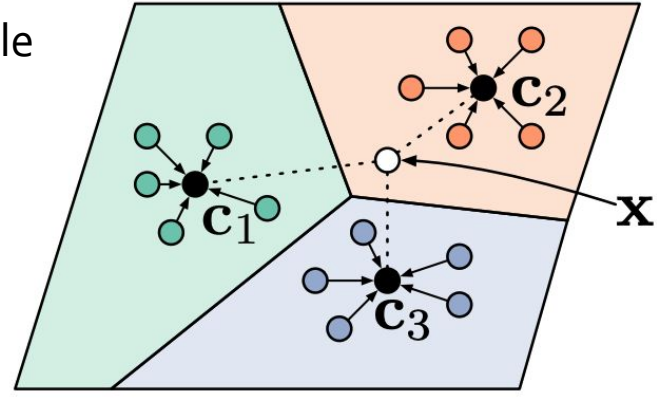
Red = unsupervised pretrain on ImNet+Target domain

[Azizi et al. ArXiv 21]

Prototypical networks

Class is represented by a single point in the mean space:

$$\mathbf{c}_k = \frac{1}{|S_k|} \sum_{(\mathbf{x}_i, y_i) \in S_k} f_{\phi}(\mathbf{x}_i)$$

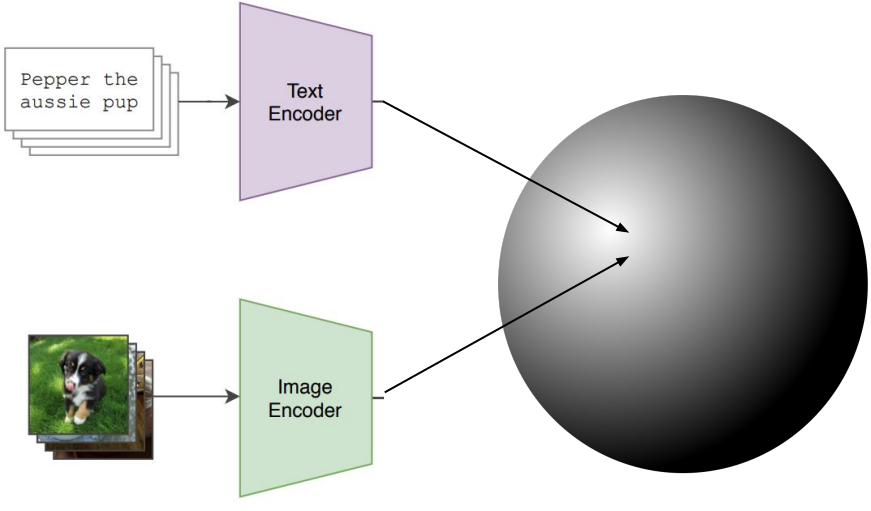


Classifying new examples:

$$p_{\phi}(y = k \mid \mathbf{x}) = \frac{\exp(-d(f_{\phi}(\mathbf{x}), \mathbf{c}_k))}{\sum_{k'} \exp(-d(f_{\phi}(\mathbf{x}), \mathbf{c}_{k'}))}$$

[Snell et al. ICLR17]

CLIP: connecting images and language



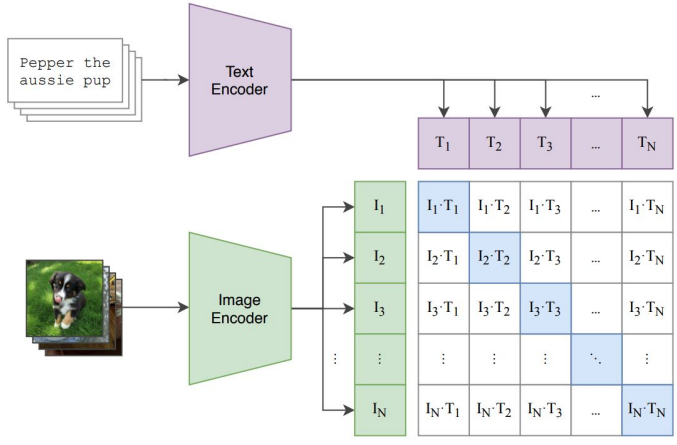
Joint space for images and text

....

....

[Radford et al. ICML21]

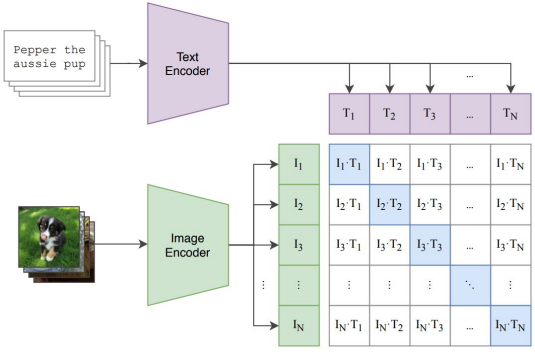
CLIP: connecting images and language




```
# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

[Radford et al. ICML21]

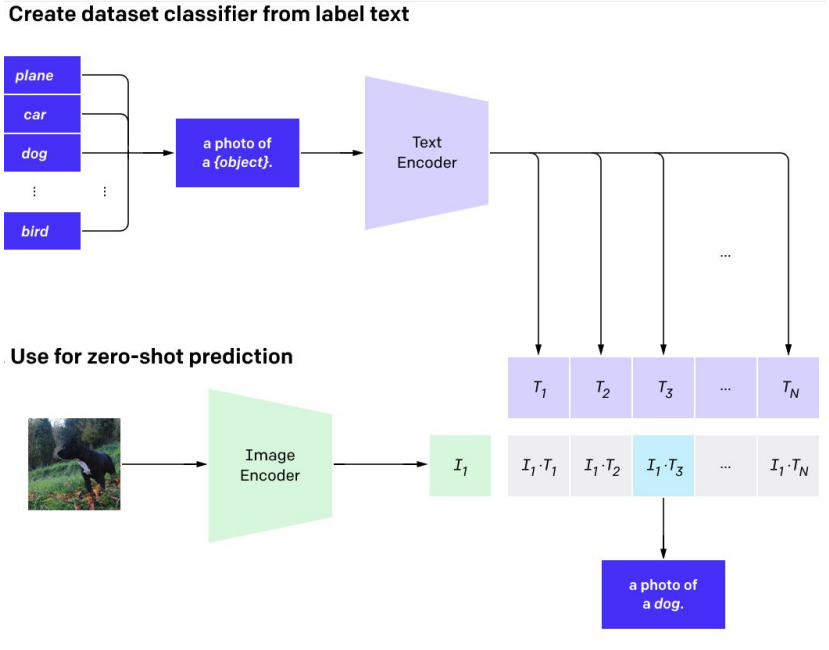
CLIP: connecting images and language



- Text encoder = Text Transformer
- Multiple architectures tried for Image encoder
- No pretraining
- Trained on 400 mln pairs from the Internet for 500,000 text queries

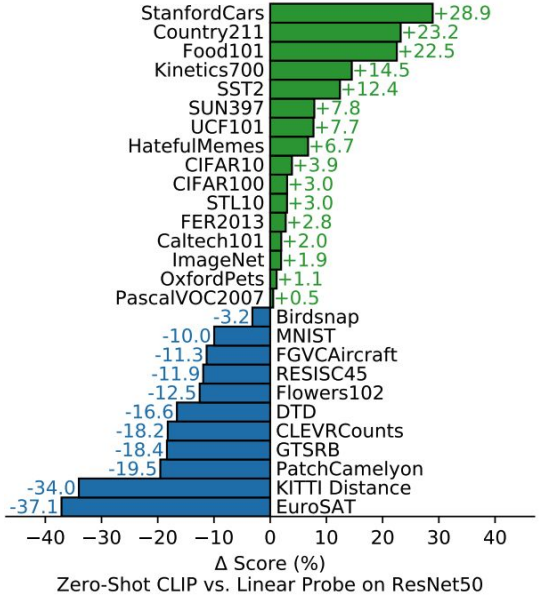
[Radford et al. ICML21]

CLIP: zero-shot evaluation



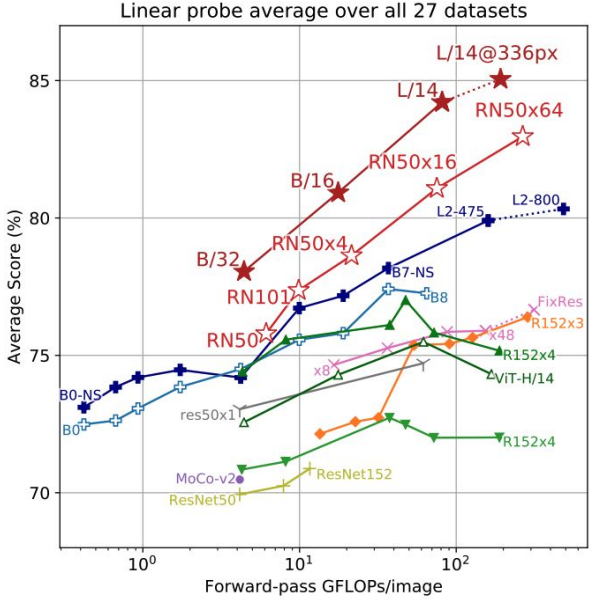
[Radford et al. ICML21]

CLIP: zero-shot evaluation



[Radford et al. ICML21]

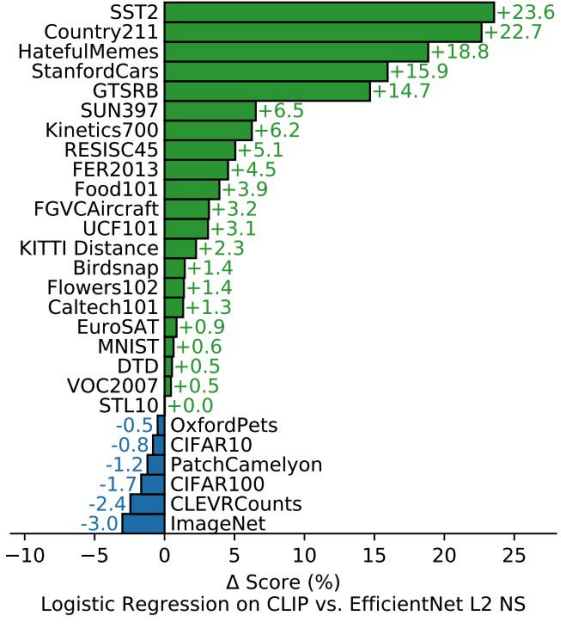
CLIP: linear-probe evaluation



CLIP-ViT
CLIP-ResNet
EfficientNet-NoisyStudent
EfficientNet
Instagram-pretrained
SimCLRv2
BYOL
MoCo
ViT (ImageNet-21k)
BiT-M
BiT-S
ResNet

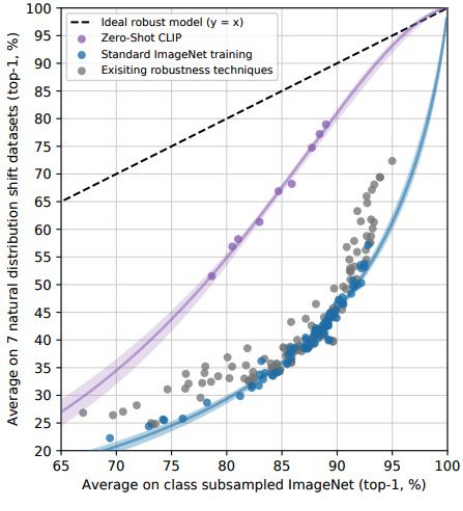
[Radford et al. ICML21]

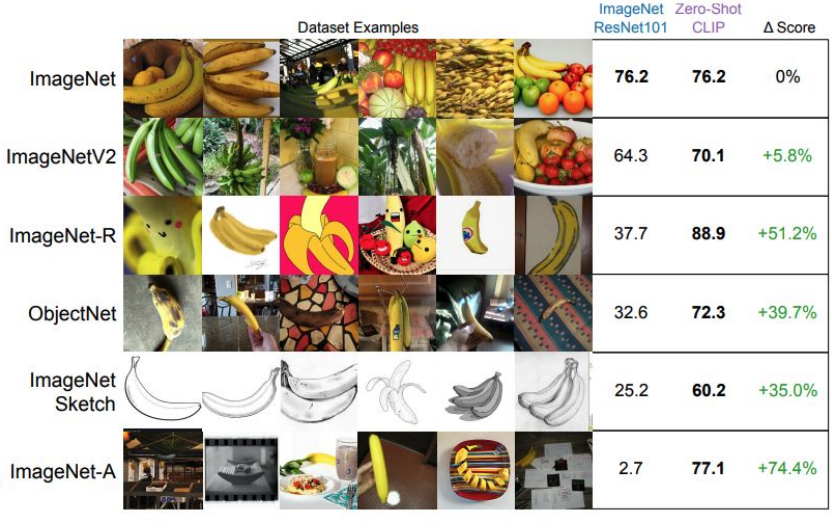
CLIP: linear-probe evaluation



[Radford et al. ICML21]

CLIP: robustness to distribution shifts





[Radford et al. ICML21]