Department of Computing

Imperial College of Science, Technology and Medicine

Self-supervised Learning

This week's paper is [1]

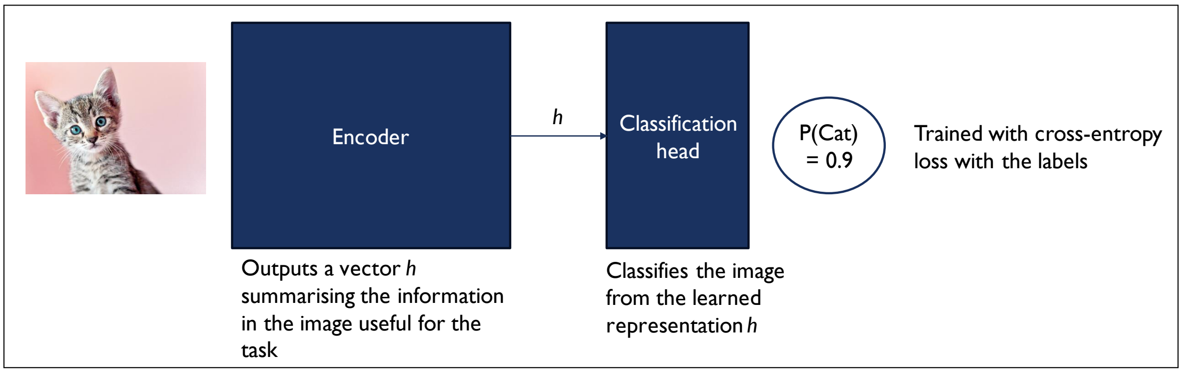
Author: Anton Zhitomirsky

Contents

1 Motivation

Supervised classifiers are very data hungry. This is a problem where labels are very expensive to acquire, but we have a lot of unlabelled images avialable (e.g. healthcare). The question is, could we learn to understand images without any labels?

1.1 Standard supervised classification pipeline



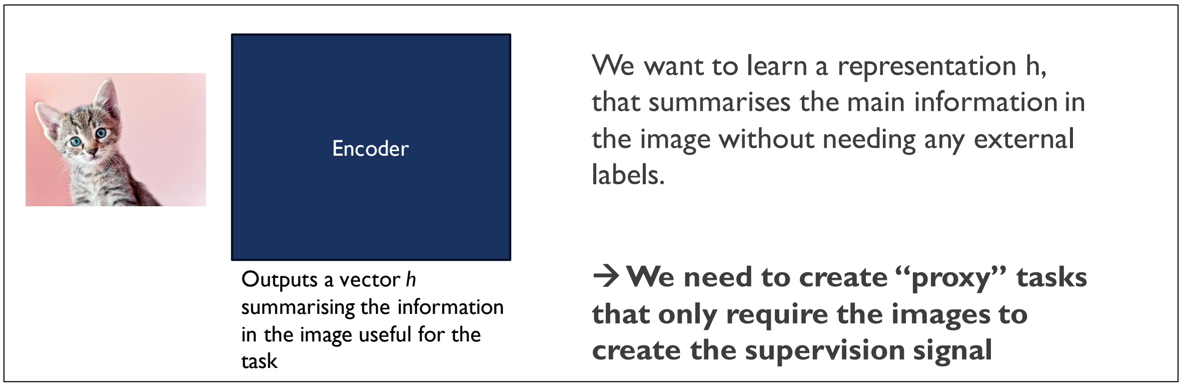
Trained with cross-entropy loss with the labels

Outputs a vector *h* summarising the information in the image useful for the task

Classifies the image from the learned representation *h*

The question remains, 'can we also learn useful image representation h without labels?'

1.2 Self-supervised learning classification pipeline



Outputs a vector *h* summarising the information in the image useful for the task

We want to learn a representation h, that summarises the main information in the image without needing any external labels.

→ We need to create "proxy" tasks that only require the images to create the supervision signal

We chop off the classificatio head becuase we don't ahve a label, so there is no classificaiton task, yet we still want to output a vector which summarises an image which still has all the important information to be used for classification as a downstream task. We require an objective for the network to strive for. Therefore we create proxys

1.2.1 Proxy

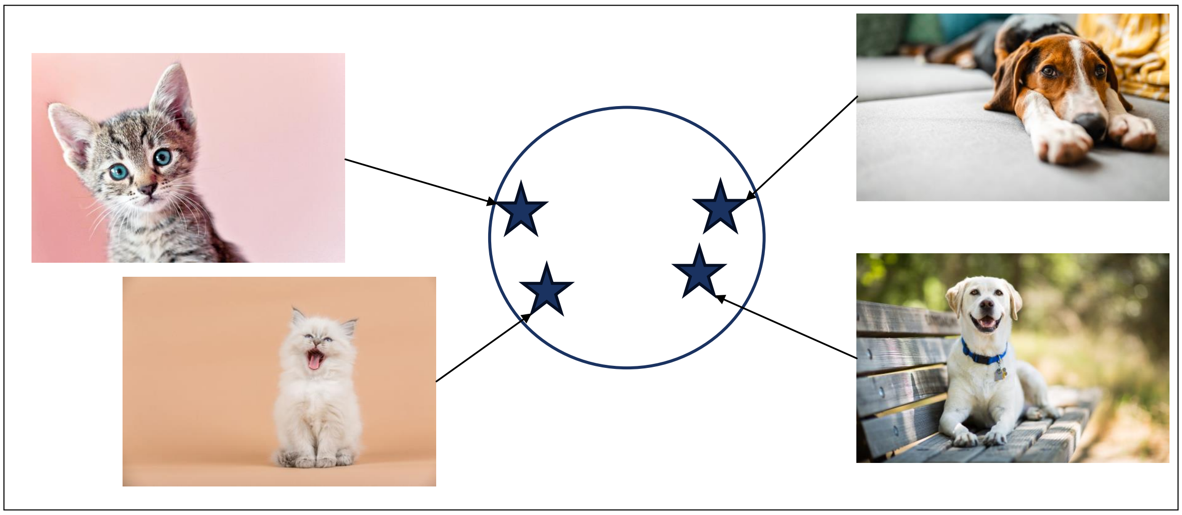
How can we create synthetic tasks that will get the network to learn useful representation?

- **Contrastive-based learning**: teaching the network to recognise meaningful pairs of images (Section 2).
- **Generative-based learning**: teaching the network to reconstruct an image from a corrupted version (Section 3)
- **Joint-Embedding Prediction**: a mix of both (Section 4).

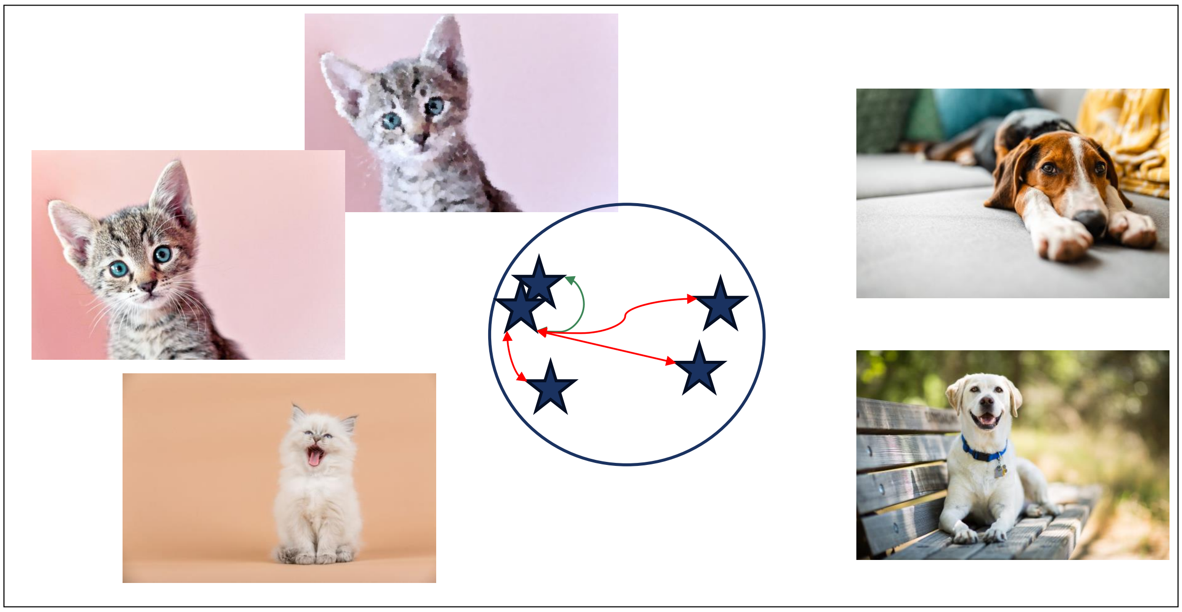
2 Contrastive-based Learning — SIMCLR

2.1 Main Idea

A meaningful representation should put similar images closer to each other.



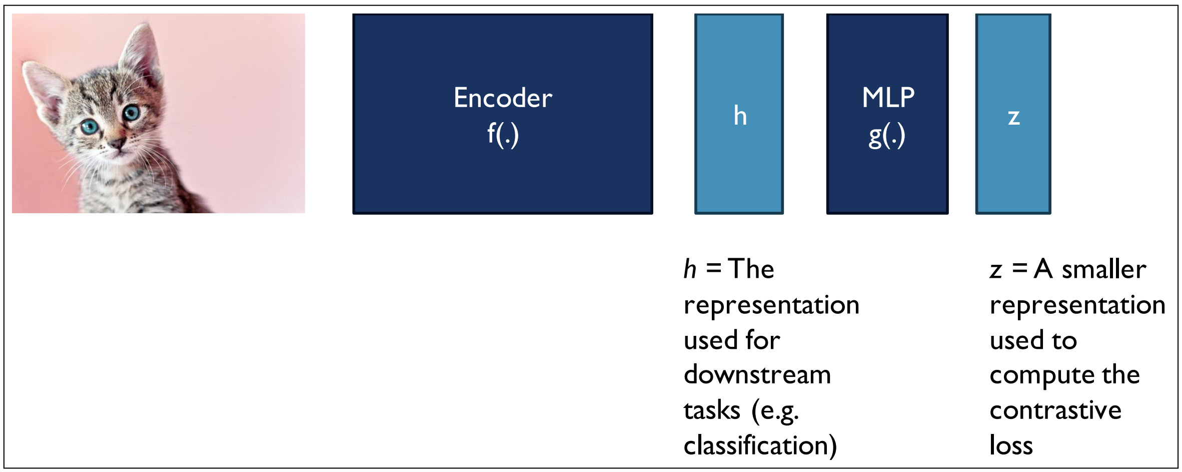
However, without labels I don't know which images are dogs or cats so I cannot use this information as supervision signal!



The extra cat here is a blurred version of a cat already assigned to the circular space. Therefore, this perturbed versino of the same image should have a similar representation to the original image compared to all other images.

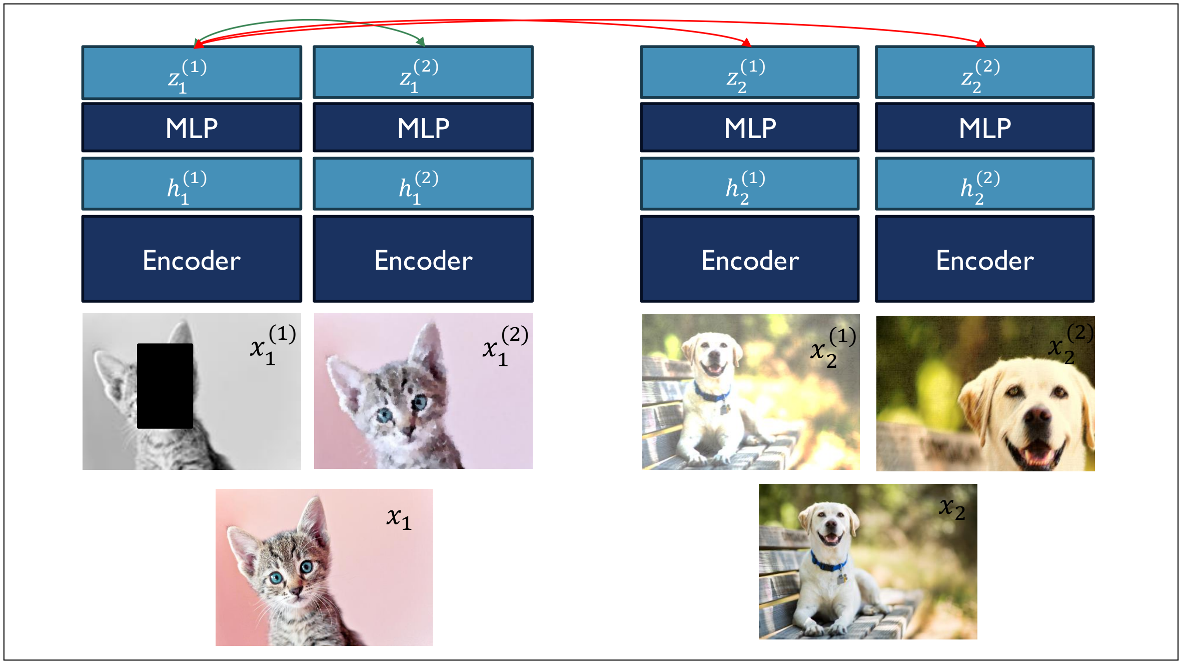
This is the main idea of contrastive-based learning: create multiple versions of the same image and theach the network to recognise the correct pair. The paper is at [1].

2.2 Main Components

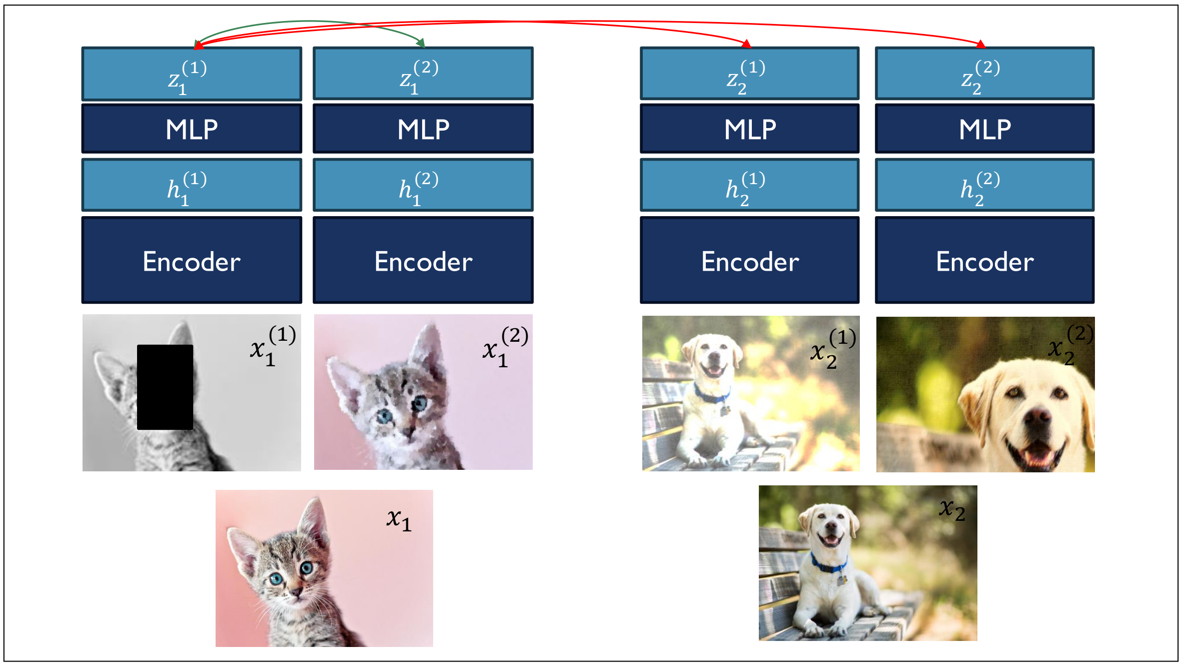

Encoder
f(.)
h
MLP
g(.)
z
h = The representation used for downstream tasks (e.g. classification)
z = A smaller representation used to compute the contrastive loss

1. We are trying to pretrain the encoder $f(\cdot)$ that we are trying to train
2. This encoder outputs a representation h
3. Add small multi-layer perception to make the dimensions smaller. This is because in high dimensional space, comparing similarities is harder, and not necessarily robust. This projector translates from vectors of size 2048 to 128 for example.

2.3 Walkthrough



Start with two images $x_1 \wedge x_2$ and apply augmentations to them to produce (for example) two views. Then pass these images through the network. We want the embeddings of the two views of the cat to be close to each other, and at the same time have them far away from the dogs.



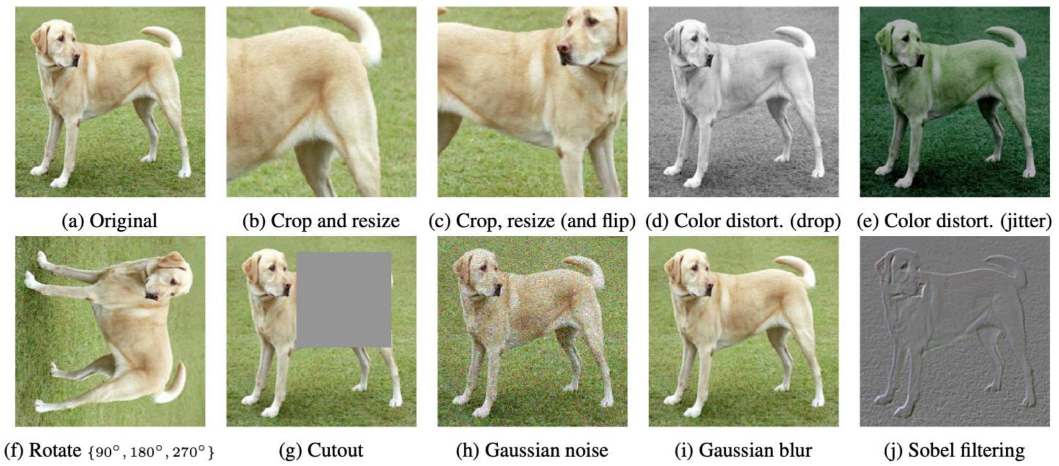
This is done for both of the views

2.4 Augmentation Pipeline

The augmentation you choose has very bix consequences on what your model will actually learn. For example, an augmentation that removes colour will implicity teach the network that you should not care about the colour, only about the concept (shapes and textures).

We require the augmentation pipeline to:

- Needs to reflect what information the model should diregard and what it should focus on
- Needs to be hard enough, otherwise trivial information is learnt.



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

(f) Rotate $\{90°, 180°, 270°\}$ (g) Cutout (h) Gaussian noise (i) Gaussian blur (j) Sobel filtering

Figure 4. Illustrations of the studied data augmentation operators. Each augmentation can transform data stochastically with some internal parameters (e.g. rotation degree, noise level). Note that we *only* test these operators in ablation, the *augmentation policy used to train our models* only includes *random crop (with flip and resize)*, *color distortion*, and *Gaussian blur*. (Original image cc-by: Von.grzanka)

Source: Chen, Ting, et al. "A simple framework for contrastive learning of visual representations."

These augmentation are very heavy, yet in the paper it is shown that this is what makes the model work

2.5 Loss Function — NT-XNet loss

2.5.1 Similarity Measure

The problem they try to solve is 'how can we attract positive pairs while repelling negative pairs'. This is the **normalised temperature contrastive loss** algorithm.

$$sim(\boldsymbol{u}, \boldsymbol{v}) = \frac{\boldsymbol{u}^\top \boldsymbol{v}}{||\boldsymbol{u}|| \cdot ||\boldsymbol{v}||} \tag{2.5.1}$$

This is essentially the cosine similarity which is the angle between the two vectors.

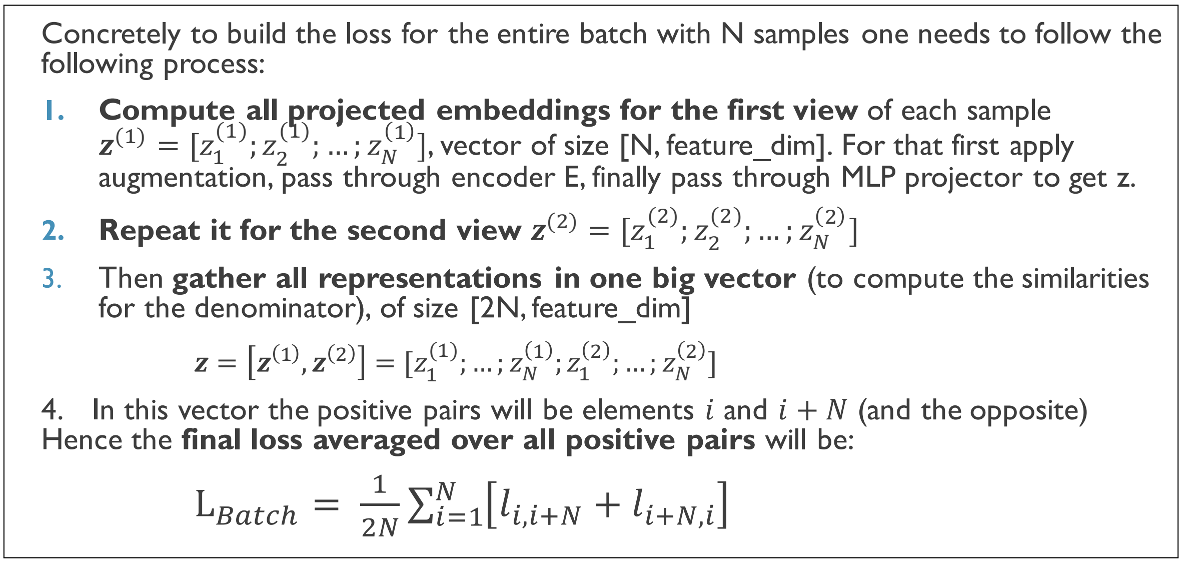
2.5.2 Loss

For each positive pair (i, j) we have the following loss:

$$\ell_{i,j} = -\log \frac{\exp(sim(\boldsymbol{z}_i, \boldsymbol{z}_j)/\tau)}{\sum_{k=1}^{SN} \mathbf{1}_{[k \neq i]} \exp(sim(\boldsymbol{z}_i, \boldsymbol{z}_k)/\tau)} \tag{2.5.2}$$

1. We want to minimise the loss, therefore we want to maximise the logarithm (since there is a leading $-$ in the loss function)
2. Therefore, we need to maximise the numerator and minimise the denominator.
3. **numerator**: We want to maximise the similarity (attract the embeddings we want together). I.e. maximise the nominator.
4. **denominator**: Measure similairty between the first image and all other images in the batch. I.e. minimise the denominator.
5. τ is the temperature, which controls how much to penalise hard negatives (negative pairs wrongly mapped close to each other). A low temperature penalises them more.

The final loss is computed across all positive pairs, both (i, j) and (j, i), in amini-batch, by averaging all $l_{i,j}$.



Concretely to build the loss for the entire batch with N samples one needs to follow the following process:

1. **Compute all projected embeddings for the first view** of each sample $\boldsymbol{z}^{(1)} = [z_1^{(1)}; z_2^{(1)}; \dots; z_N^{(1)}]$, vector of size [N, feature_dim]. For that first apply augmentation, pass through encoder E, finally pass through MLP projector to get z.
2. **Repeat it for the second view** $\boldsymbol{z}^{(2)} = [z_1^{(2)}; z_2^{(2)}; \dots; z_N^{(2)}]$
3. Then **gather all representations in one big vector** (to compute the similarities for the denominator), of size [2N, feature_dim]

$$\boldsymbol{z} = [\boldsymbol{z}^{(1)}, \boldsymbol{z}^{(2)}] = [z_1^{(1)}; \dots; z_N^{(1)}; z_1^{(2)}; \dots; z_N^{(2)}]$$

4. In this vector the positive pairs will be elements i and $i + N$ (and the opposite)

Hence the **final loss averaged over all positive pairs** will be:

$$\mathrm{L}_{Batch} = \frac{1}{2N} \sum_{i=1}^{N} \left[l_{i,i+N} + l_{i+N,i} \right]$$

2.5.3 Triplet Loss

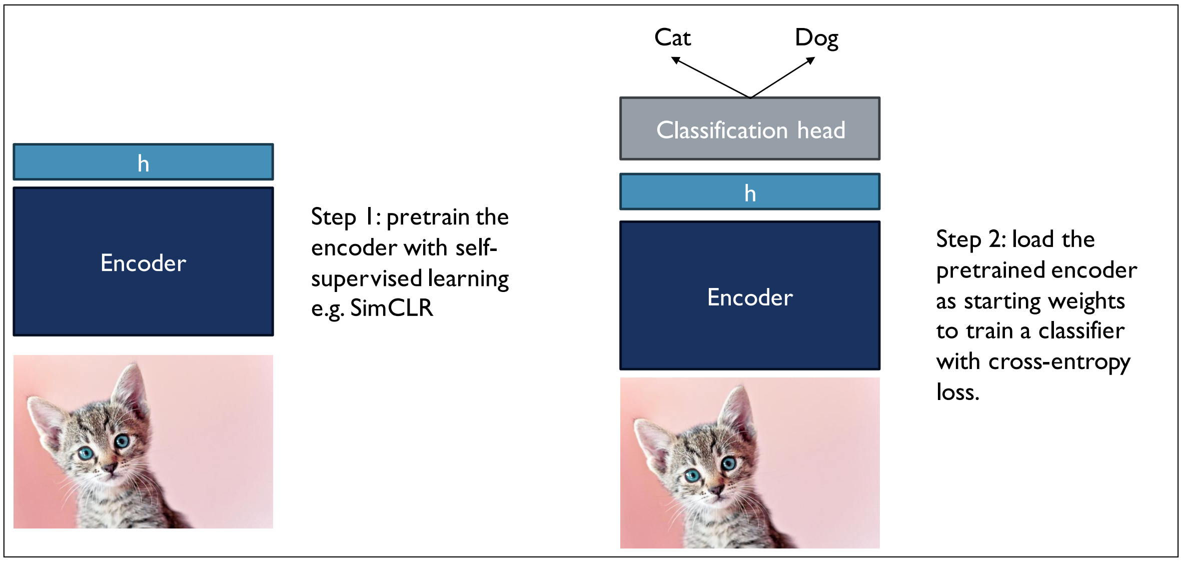
$$\mathcal{L}_{triplet}(\boldsymbol{x}, \boldsymbol{x}^+, \boldsymbol{x}^-) = \sum_{x \in \mathcal{X}} \max(0, ||f(x) - f(x^+)||_2^2 - ||f(x) - f(x^-)||_2^2 + \epsilon) \tag{2.5.3}$$

- x is one image
- x^+ is the corresponding positive
- x^- is a randomly selected negative image
- ϵ is the marign parmeter controlling how far the negatives should be compared to the positive pairs (temperature τ plays this role in NT-Xent loss)

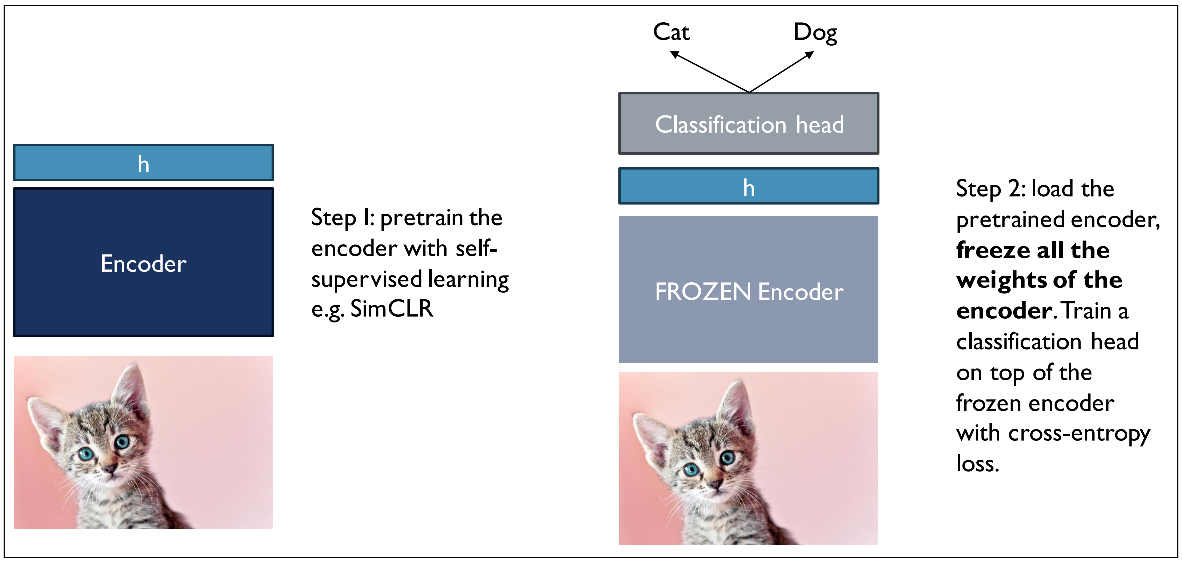
Here you want a similar thing: you wat the positive paris to be closer than the distance between the negatives (plus the margin). This qnautity is negative so the loss is zero. “The positive pairs should be closer to eachother by at least ϵ margin compared to the distance to any negative pair”

2.6 Evaluation of Self-supervised Learning

2.6.1 Finetuning


h
Encoder
Step 1: pretrain the encoder with self-supervised learning e.g. SimCLR
Cat
Dog
Classification head
h
Encoder
Step 2: load the pretrained encoder as starting weights to train a classifier with cross-entropy loss.

2.6.2 Linear evaluation


Cat
Dog
Classification head
h
Encoder
Step I: pretrain the encoder with self-supervised learning e.g. SimCLR
h
FROZEN Encoder
Step 2: load the pretrained encoder, **freeze all the weights of the encoder**. Train a classification head on top of the frozen encoder with cross-entropy loss.

“you can only base yourself on what you learned during the pre-training. So it cna be a good way to check what you’re actually learning during the pretraining as opposed to if you do for fine tuning you might modify the encoder” If you have few labels, then sometimes its better to keep the forzen encoder and just rtain the classificaiton label layer

2.7 BatchSize

To learn good representation, it is crucial to use very large batch sizes with SimCLR.

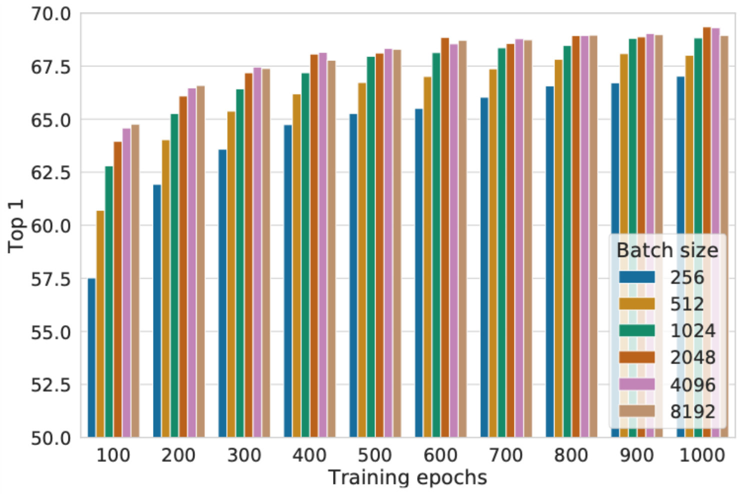


Figure 9. Linear evaluation models (ResNet-50) trained with different batch size and epochs. Each bar is a single run from scratch.[10]

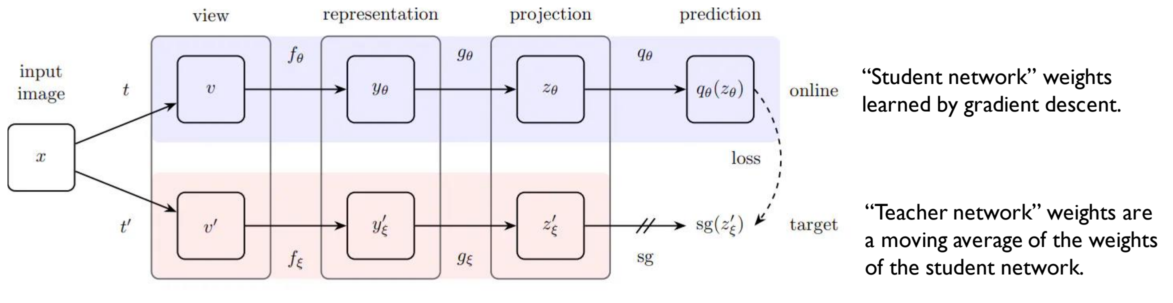
Linear evaluation — Key take-away; the more you pre-train the model, the better it gets. Also, bigger batch sizes learn a better representation of the image, the more images there are in your batch, the harder the task is.

2.8 Follow-up work

SimCLR requires very large batch size to provide good representation, this means very high computational requirements (expensive, difficult to train).

2.8.1 Boostrap your won latent

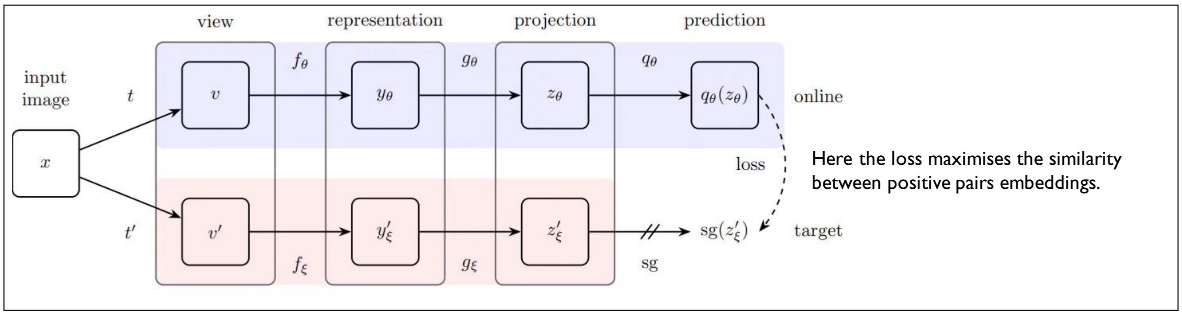
BYOL removes the need to use negative pairs in the loss function, instead one just optimises the similarity on the positive pairs → less need for bigger batch sizes. For that they need to introduce a new architecture to avoid learning trivial embeddings (otherwise all converge to the same point).



“Student network” weights learned by gradient descent.

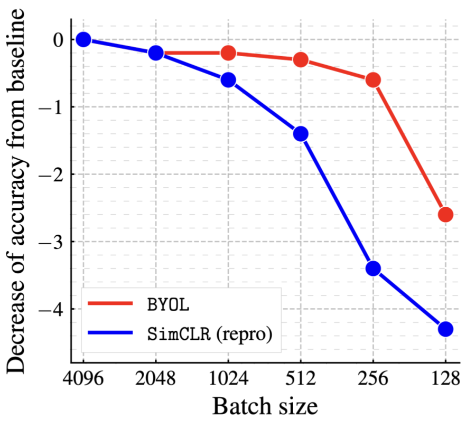
“Teacher network” weights are a moving average of the weights of the student network.

Where previously you would pass your two images into the same path, here we have them go itno two separate paths (because they might obtain the trivial solution). The student and teacher (as described above) the weights will not be the same so they won’t return the same representation



Here the loss maximises the similarity between positive pairs embeddings.

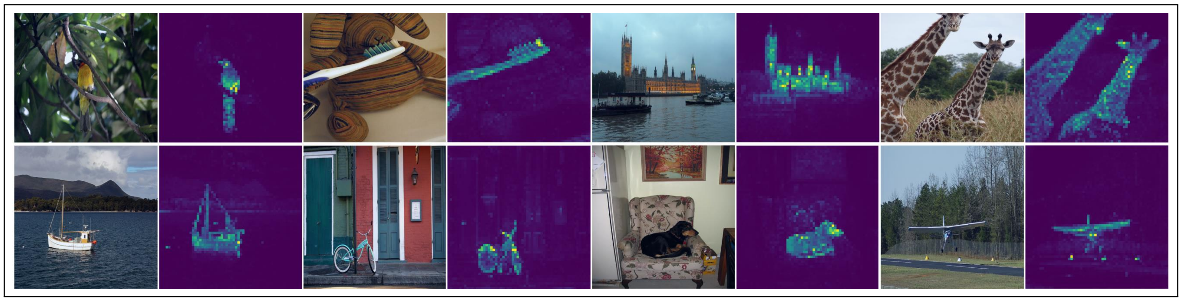
“To further avoid the risk of still learning the trivial solution where these two just end up converging to the same point, they add yet another MLP (prediction) and try to match the output of online and target. You’re trying to say that the output of the student network after the prediciton should be the same as the output of the target network without the prediction”



BYOL is **more robust to smaller batch size** than SimCLR.

2.8.2 DINO

Very similar to training BYOl, but using visial transformers as encoders. "Becuase they use vision transfoemers they have access to this attention layer and os they can visualise the attention amp of the network when they create the self-supervised encoding, and by visualising these attention maps they can generate self-supervised segmentation maps".



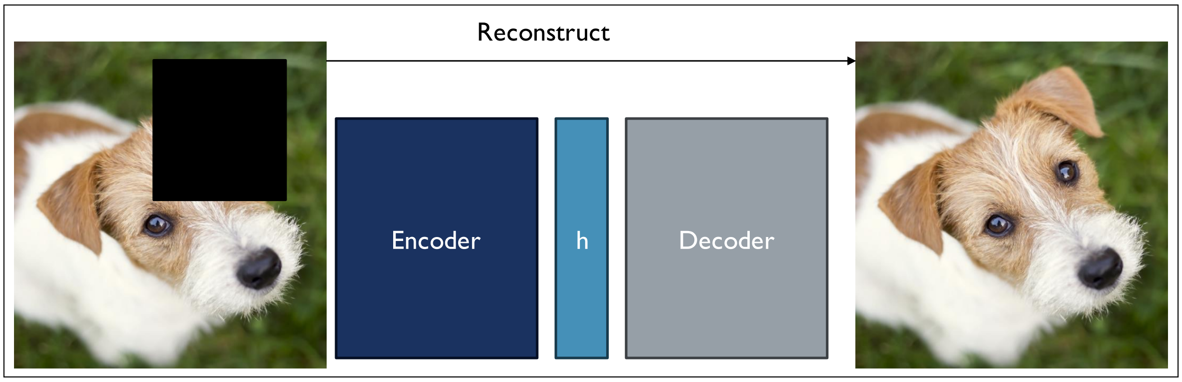
2.9 Conclusion

Contrastive learning is one pretext task that has proven very useful in self-supervised learning, but its not the only way. In particular, it has some drawbacks such as large batch sizes, design of the augmentation pipeline etc.

3 Generative-based Learning

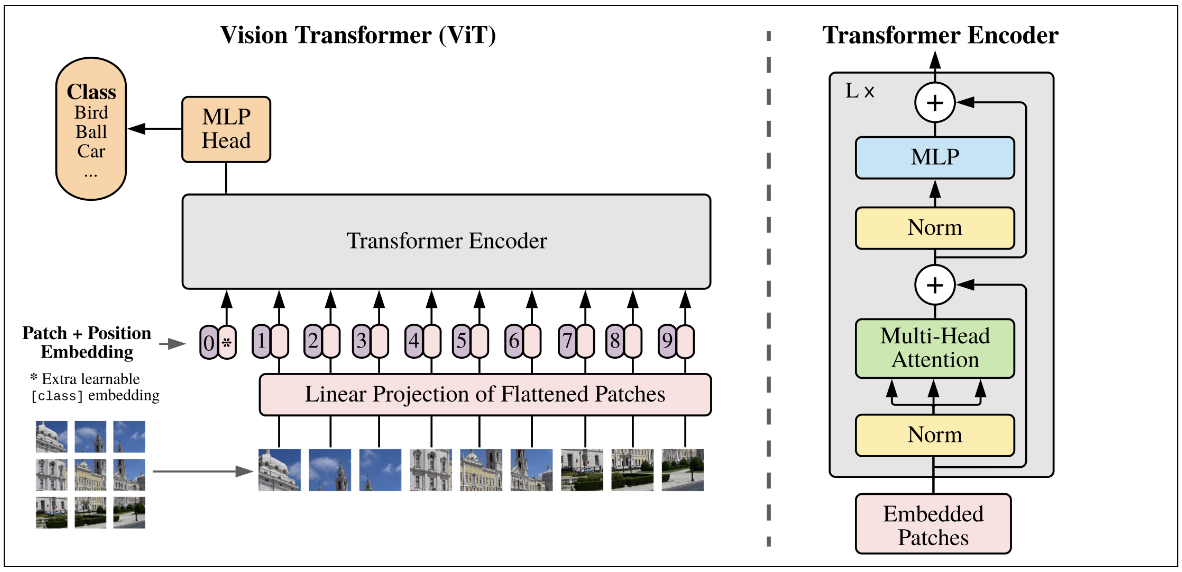
3.1 Main-Idea

Another way to use the image as supervision signal is to teach the network to reconstruct the original image from a masked image.



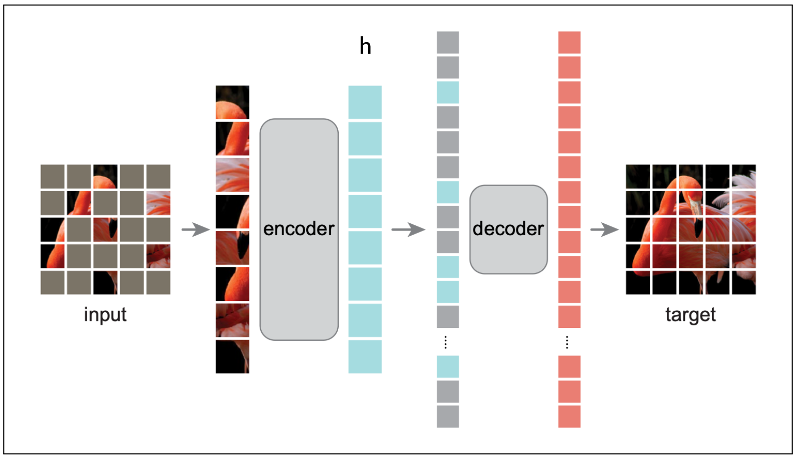
Instead of using pairs, constructing pairs to teach the network to learn semantics, they teach the network to reconstruct the original image from a masked image.

3.1.1 Vision transformers (refresher)

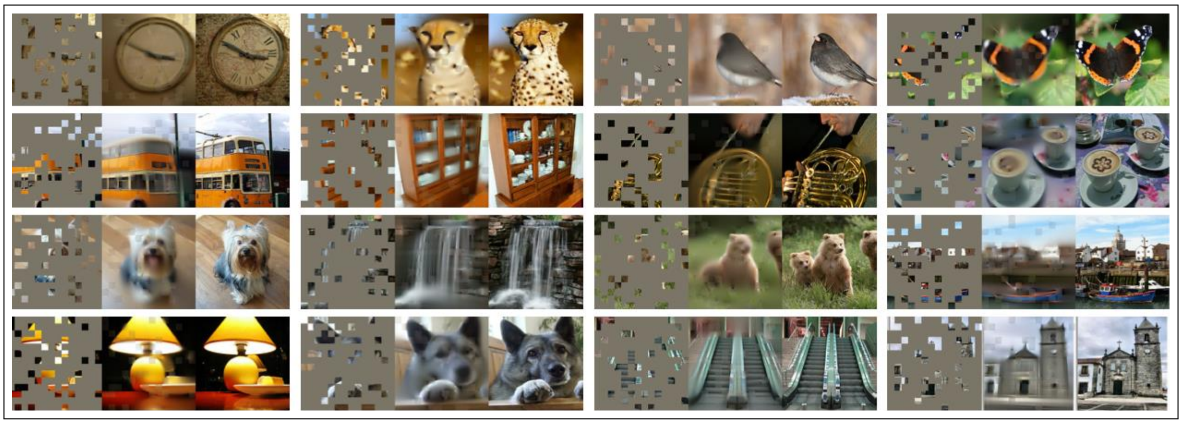


“They start with one image, they cut it in different non-overlapping patchesthen they pass it through some embedding and then they get it into a transformer encoder, which ahs some attention layers, and the attention layers update the representation of that patch using the information from the other patches, and progressively it gets updated and then it learns useful representaiton of each of these single patches.”

3.2 Masked Auto-encoders are scalable vision learners



They divide the image in multiple subpatches and randomly mask some of the patches. After taking the patches they have, they feed it through the encoder which is a vision transformer. The encoder gets a representation for each patch and then you fill in the blanks iwth some mask token you learn during the process. The decoder then predicts each patch.
The encoder here is drawn bigger becuase most of the computation is done in the encoder, and the decoder is pretty small. The goal is to put most of the details in the encoder (after training the decoder is thrown away).



The reconstruction of the detail is pretty bad, but this is because the size of the decoder is smaller (and we throw away the decoder after training)

3.2.1 Loss

$$MSE = \sum(\hat{x}_i - x_i)^2, \quad \text{where } i \text{ is the pixel index} \tag{3.2.1}$$

3.2.2 Scalability

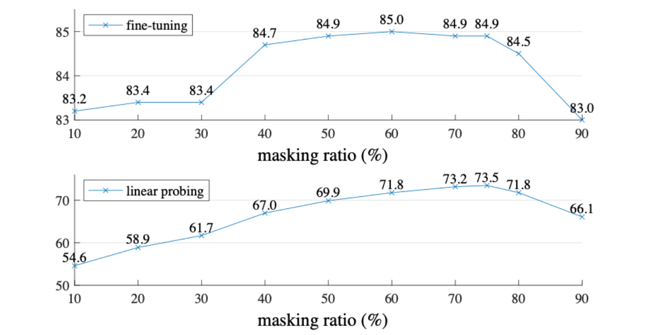


Figure 5. **Masking ratio**. A high masking ratio (75%) works well for both fine-tuning (top) and linear probing (bottom). The y-axes are ImageNet-1K validation accuracy (%) in all plots in this paper.

The best result they got was approximately 75% mask ratio

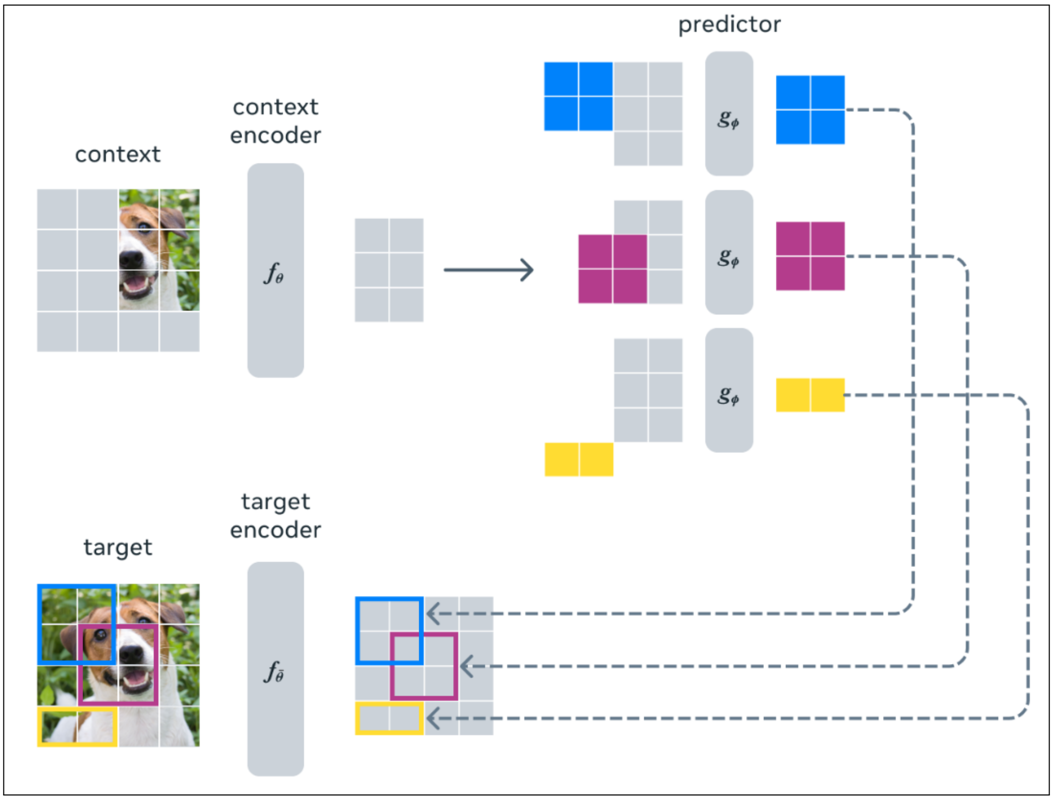
3.3 Conclusion

MAE avoids some of the drawbacks of CL, however, training a full reconstruction model is qute expensive. Maybe its not necessarily to full ylearn a perfect decoder model since we throw it away after training?

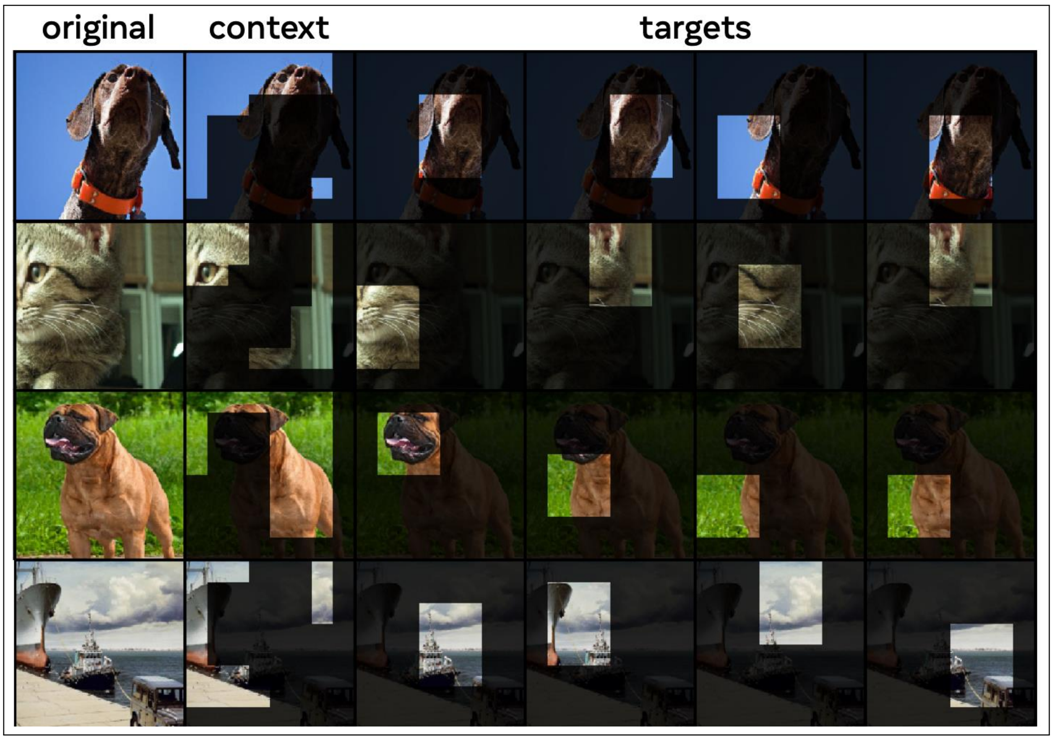
4 Joint-Embedding Prediction

4.1 i-Jepa

Idea is to combine strength of CL and MAE approaches.



Some patches are masked, and we pass on the available patches to the context encoder. At the same time, we sample a region of the blue window, and pass it thorugh the target embedding. Repeat this for several regions. The goal of this paradigm is that instead of predicitng the missing image region, we are going to predict the encoded representation because the goal of the encoders is to summarise information, so instead of wasting compute reconstructing ever single pixel in the image, we are going to reconstruct the important information which there should be if the network is trained summarised.

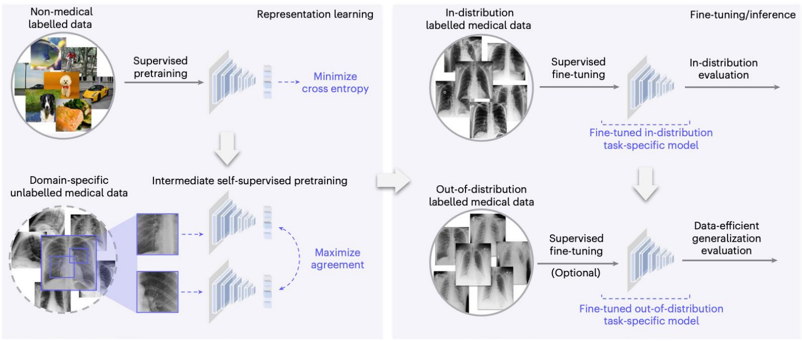


4.1.1 Loss

The loss is simply the average L_2 distance between the predicted patch-level representations $\hat{s}_y(i)$ and the target patch-level representation $s_y(i)$ i.e.,

$$\frac{1}{M}\sum_{i=1}^{N} D(\hat{s}_y(i), s_y(i)) = \frac{1}{M}\sum_{i=1}^{M}\sum_{j\in B_i} ||\hat{s}_{y_j} - s_{y_j}||_2^2. \tag{4.1.1}$$

5 Examples in Healthcare



Non-medical labelled data
Representation learning
Supervised pretraining
Minimize cross entropy
Domain-specific unlabelled medical data
Intermediate self-supervised pretraining
Maximize agreement
In-distribution labelled medical data
Fine-tuning/inference
Supervised fine-tuning
In-distribution evaluation
Fine-tuned in-distribution task-specific model
Out-of-distribution labelled medical data
Supervised fine-tuning
(Optional)
Data-efficient generalization evaluation
Fine-tuned out-of-distribution task-specific model

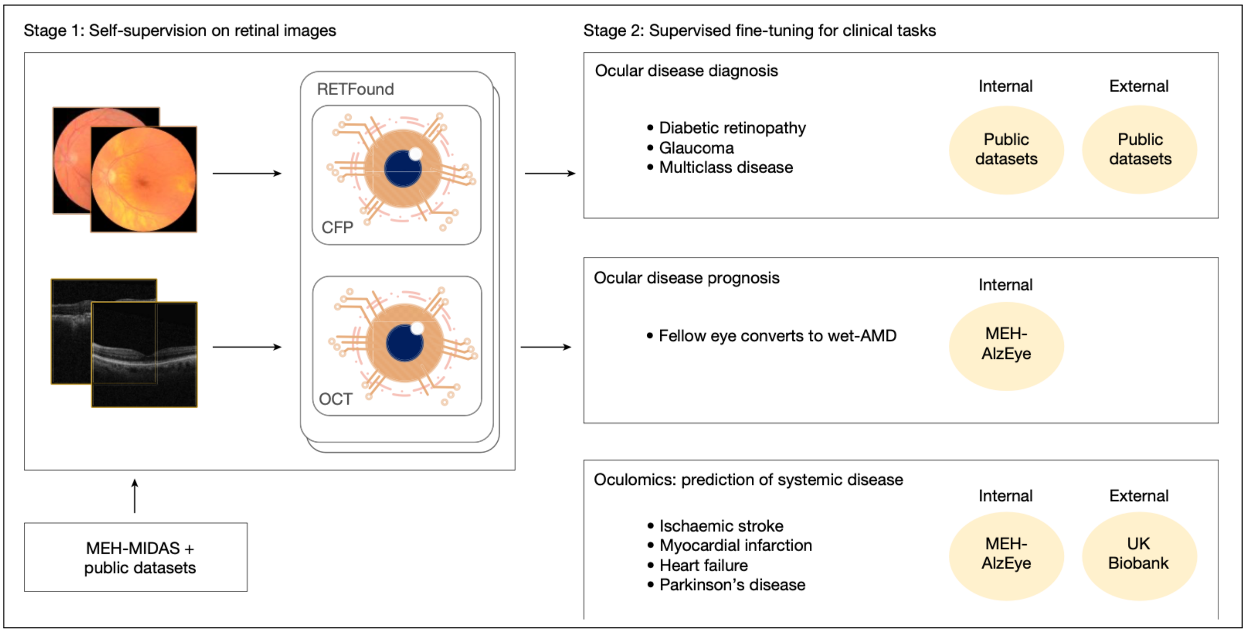
1 | Overview of the REMEDIS approach for developing robust and efficient for medical imaging. REMEDIS starts with representations initialized using e-scale natural-image pretraining following the BiT method[52]. We then adapt model to the medical domain using intermediate contrastive self-supervised learning without using any labelled medical data. Finally, we fine-tune the model to specific downstream medical-imaging ML tasks. We evaluate the ML model in both ID and OOD settings to establish the data-efficient generalization performance of the model.

The authors show that pretraining models with SSL:

→ improves performance overall

→ Improves model generalisation to external deployment settings

1. “Start by pretraining it on ImageNet (better performance than from scratch)”
2. “then they run SIMCLR from these weights. The goal is to go from a very generic encoder that has no idea baout medical languages to specialise it to learn on these unlablled chest x-rays”
3. “then they will use this on downstream tasks”



Half a million scans were trained which didn’t have labels, and saw that pre-training improved performance than from scratch.

References

[1] Ting Chen et al. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020. arXiv: 2002.05709 [cs.LG].