Vision (and Language) Pretraining

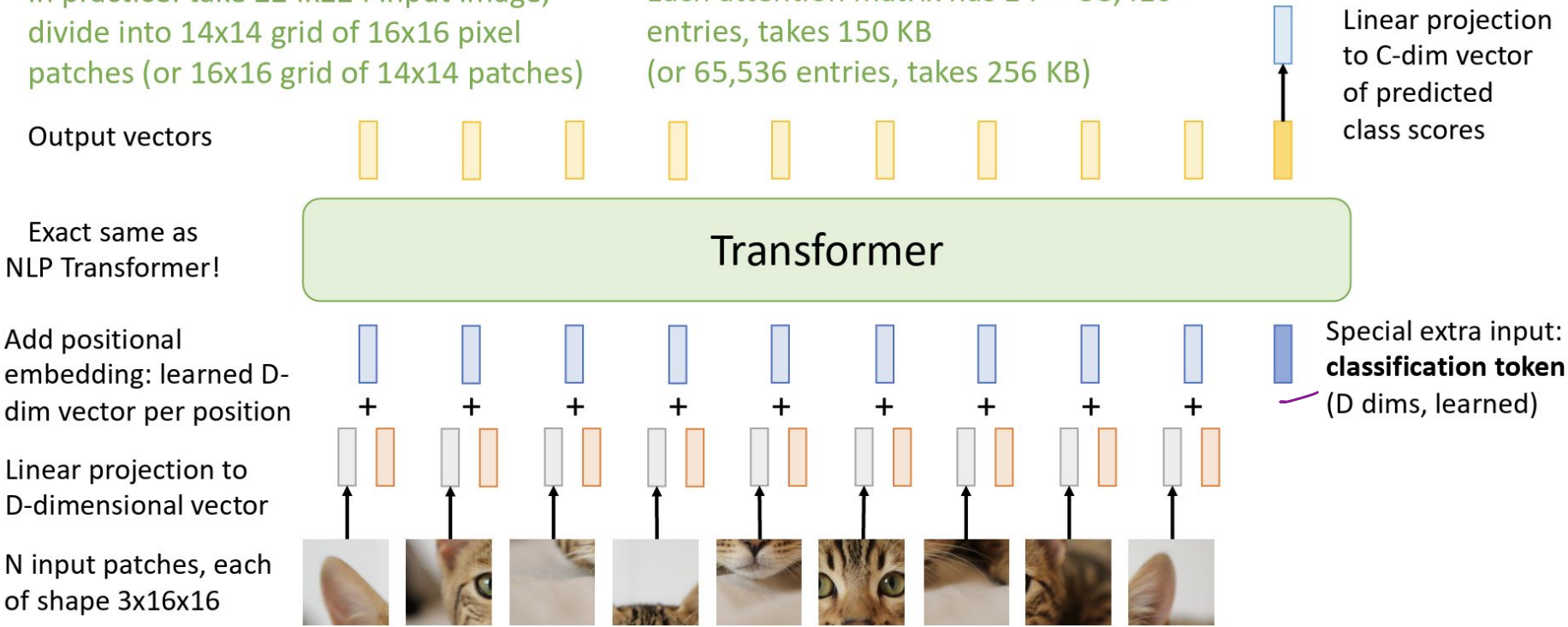
CS60010

**Various slides adapted from CS231n*

What about self-supervision for images?

In practice: take 224x224 input image, divide into 14x14 grid of 16x16 pixel patches (or 16x16 grid of 14x14 patches)

Each attention matrix has $14^4 = 38{,}416$ entries, takes 150 KB
(or 65,536 entries, takes 256 KB)



Is this not self-supervision?

Self-supervised pretext tasks

Example: learn to predict image transformations / complete corrupted images

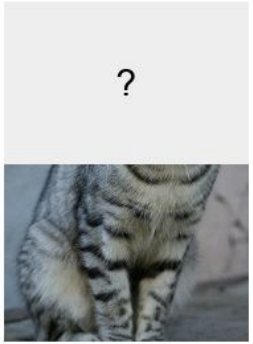
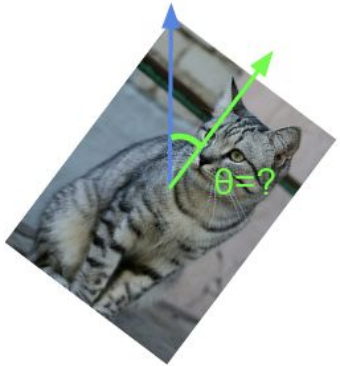


image completion



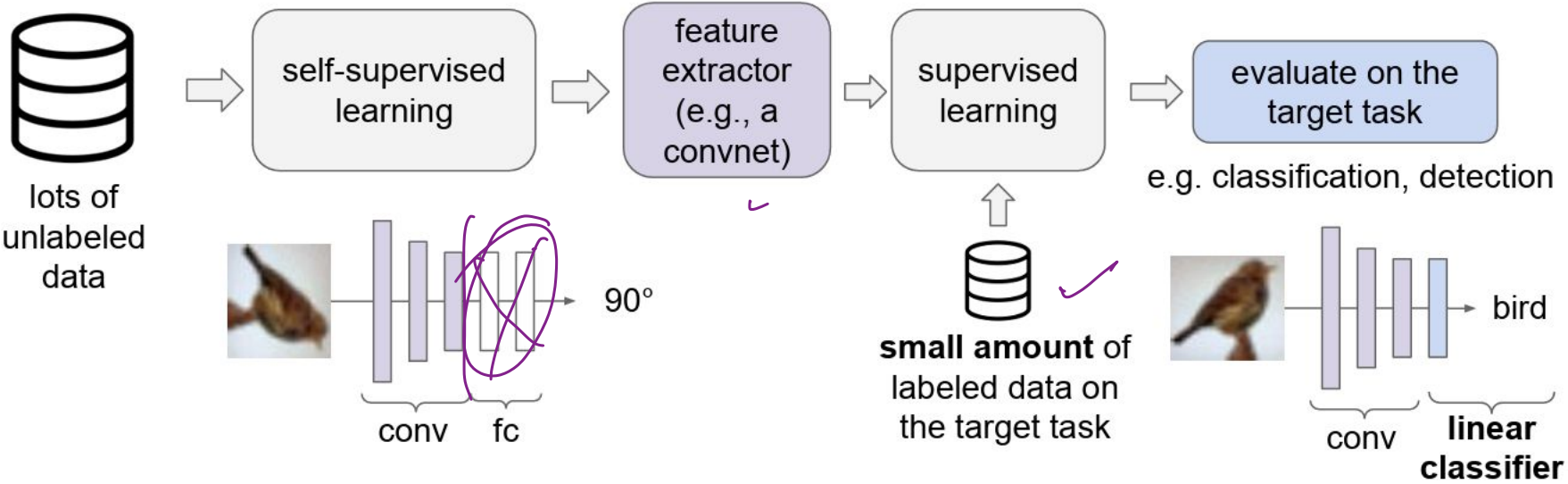
rotation prediction

“jigsaw puzzle”

colorization

1. Solving the pretext tasks allow the model to learn good features.
2. We can automatically generate labels for the pretext tasks.

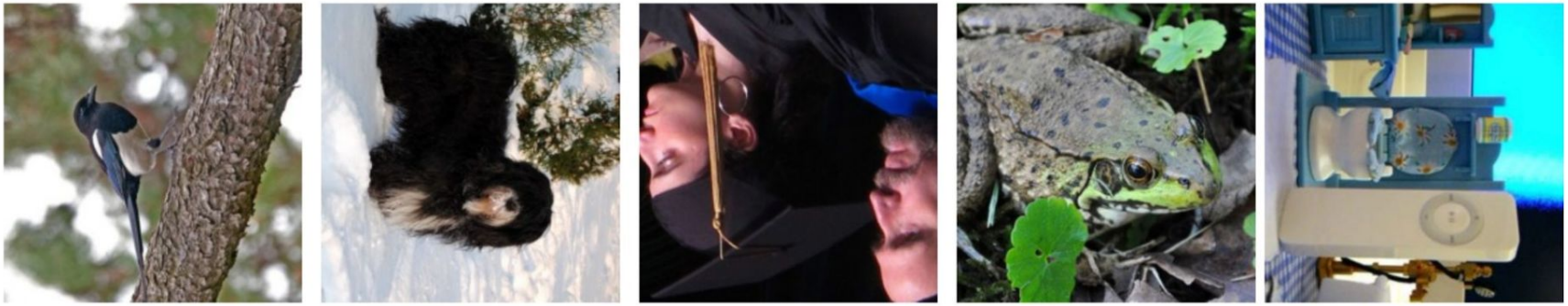
How to evaluate a self-supervised learning method?



1. Learn good feature extractors from self-supervised pretext tasks, e.g., predicting image rotations

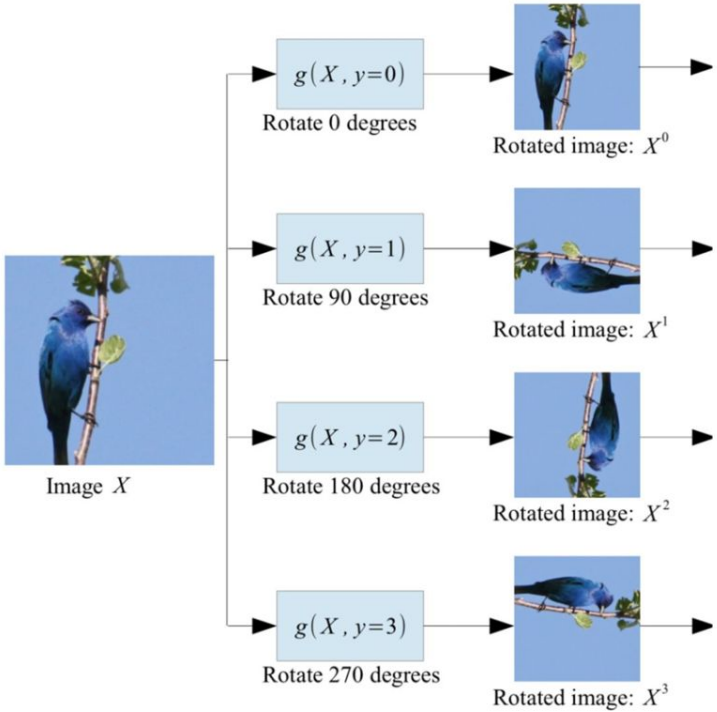
2. Attach a shallow network on the feature extractor; train the shallow network on the target task with small amount of labeled data

Pretext task: predict rotations



Hypothesis: a model could recognize the correct rotation of an object only if it has the "visual commonsense" of what the object should look like unperturbed.

Pretext task: predict rotations

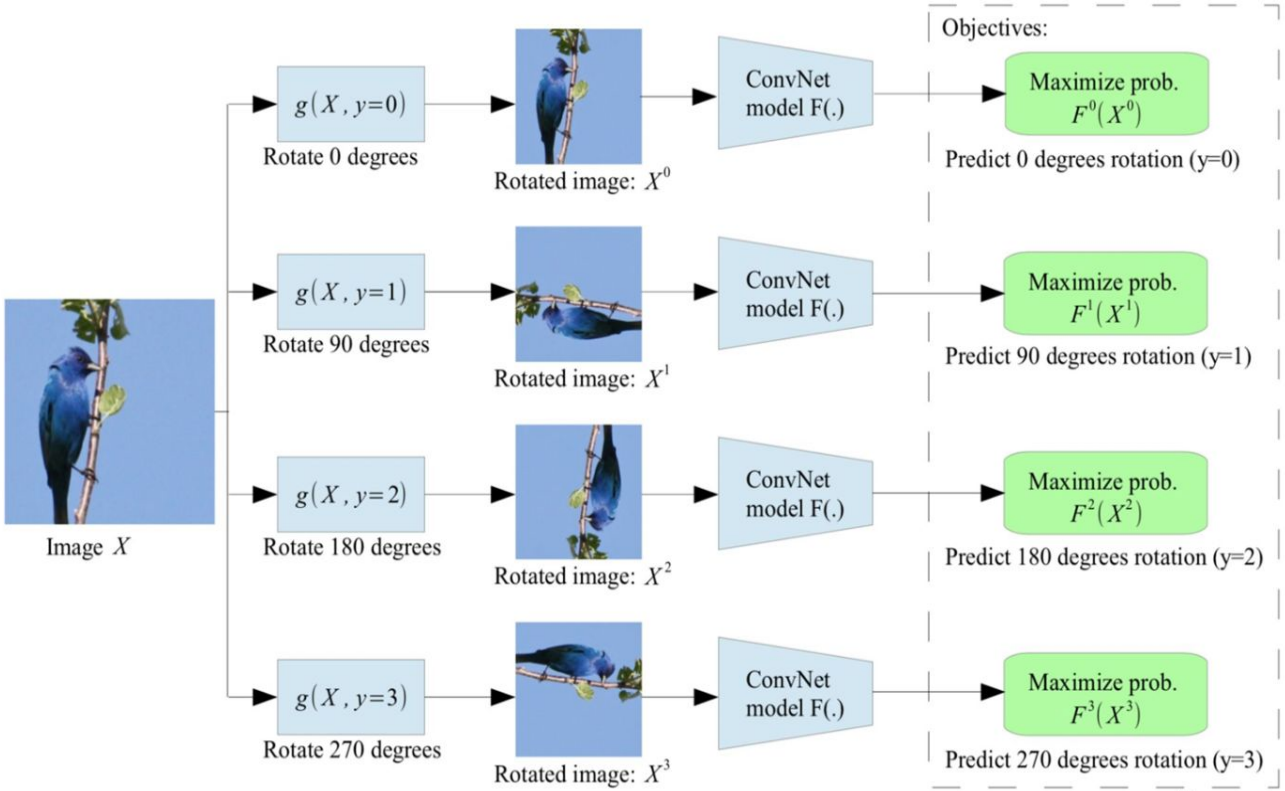


Self-supervised learning by rotating the entire input images.

The model learns to predict which rotation is applied (4-way classification)

(Image source: Gidaris et al. 2018)

Pretext task: predict rotations

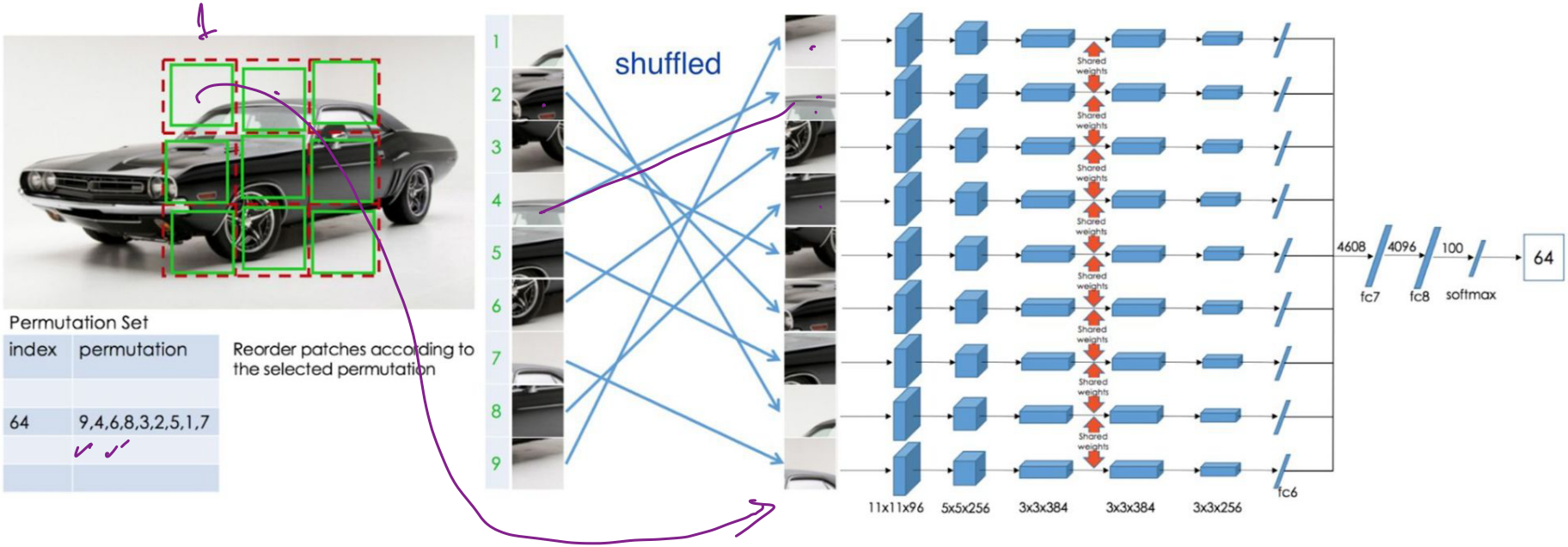


Self-supervised learning by rotating the entire input images.

The model learns to predict which rotation is applied (4-way classification)

(Image source: Gidaris et al. 2018)

Pretext task: solving “jigsaw puzzles”



(Image source: [Noroozi & Favaro, 2016](#))

Summary: pretext tasks from image transformations

- Pretext tasks focus on “visual common sense”, e.g., predict rotations, inpainting, rearrangement, and colorization.
- The models are forced learn good features about natural images, e.g., semantic representation of an object category, in order to solve the pretext tasks.
- We don’t care about the performance of these pretext tasks, but rather how useful the learned features are for downstream tasks (classification, detection, segmentation).

Pretext tasks from image transformations

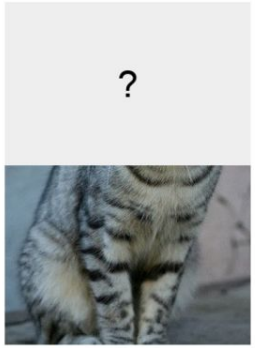
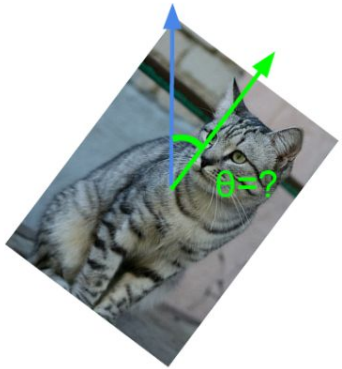


image completion



rotation prediction

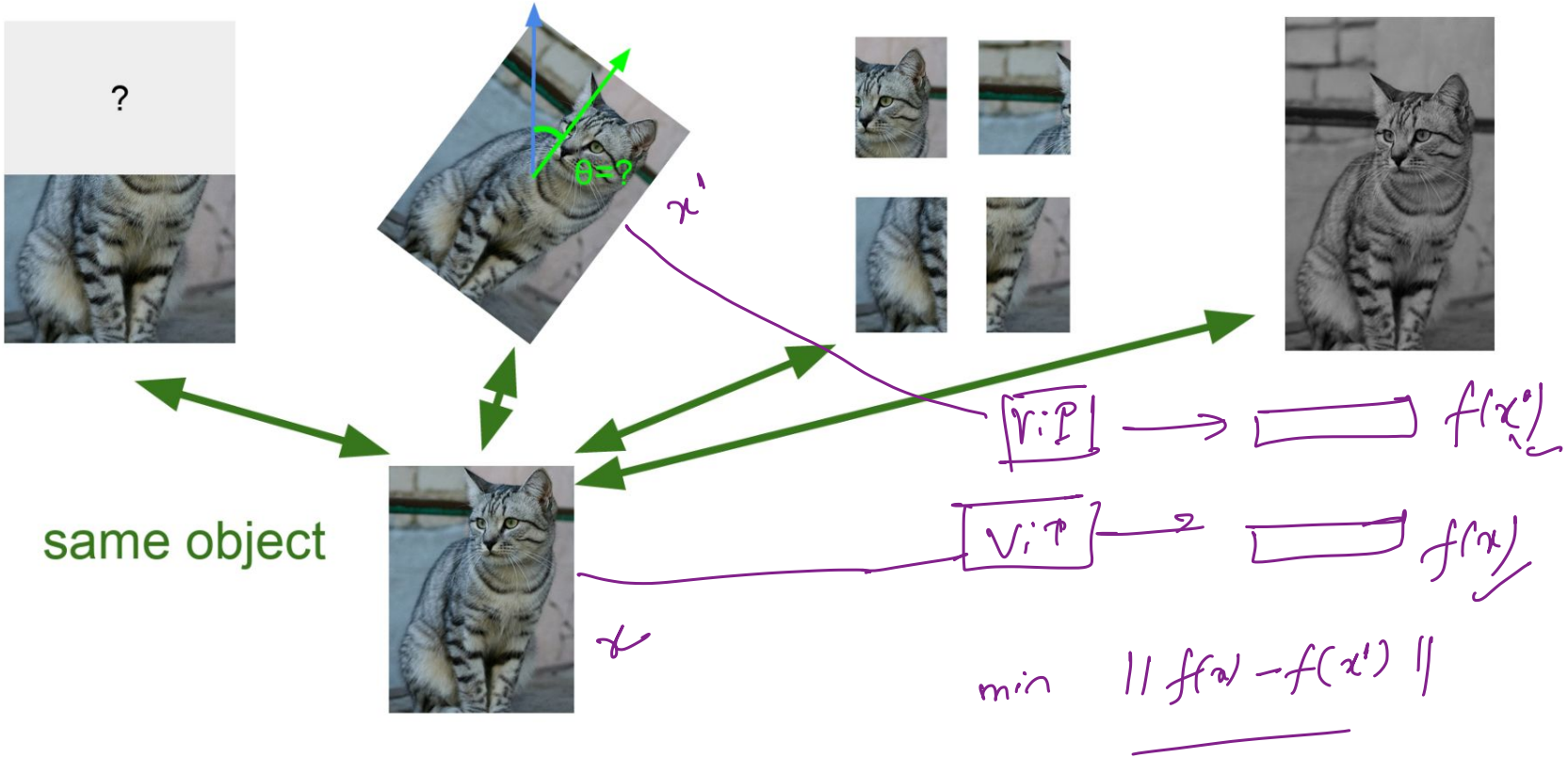
“jigsaw puzzle”

colorization

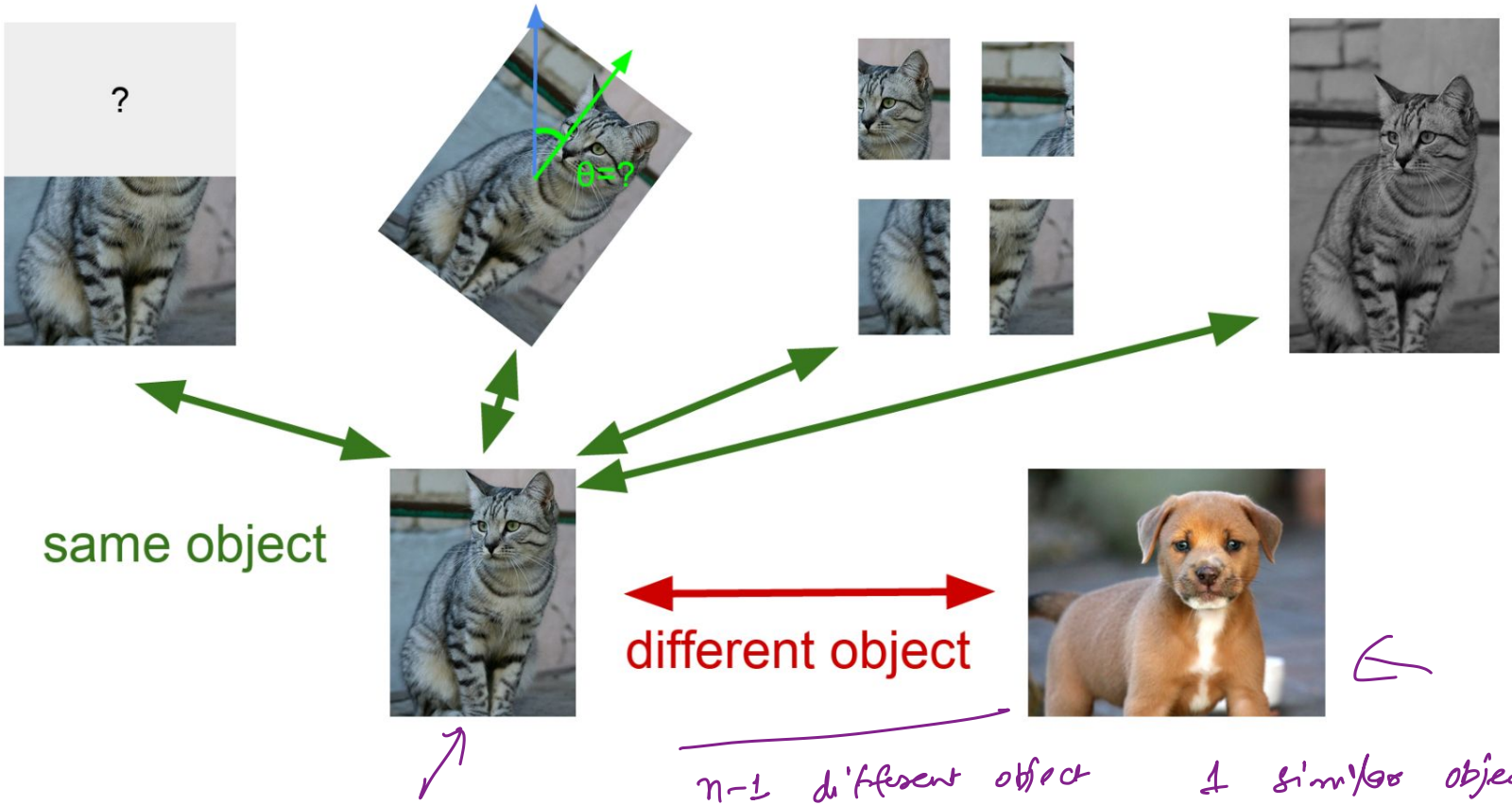
Learned representations may be tied to a specific pretext task!

Can we come up with a more general pretext task?

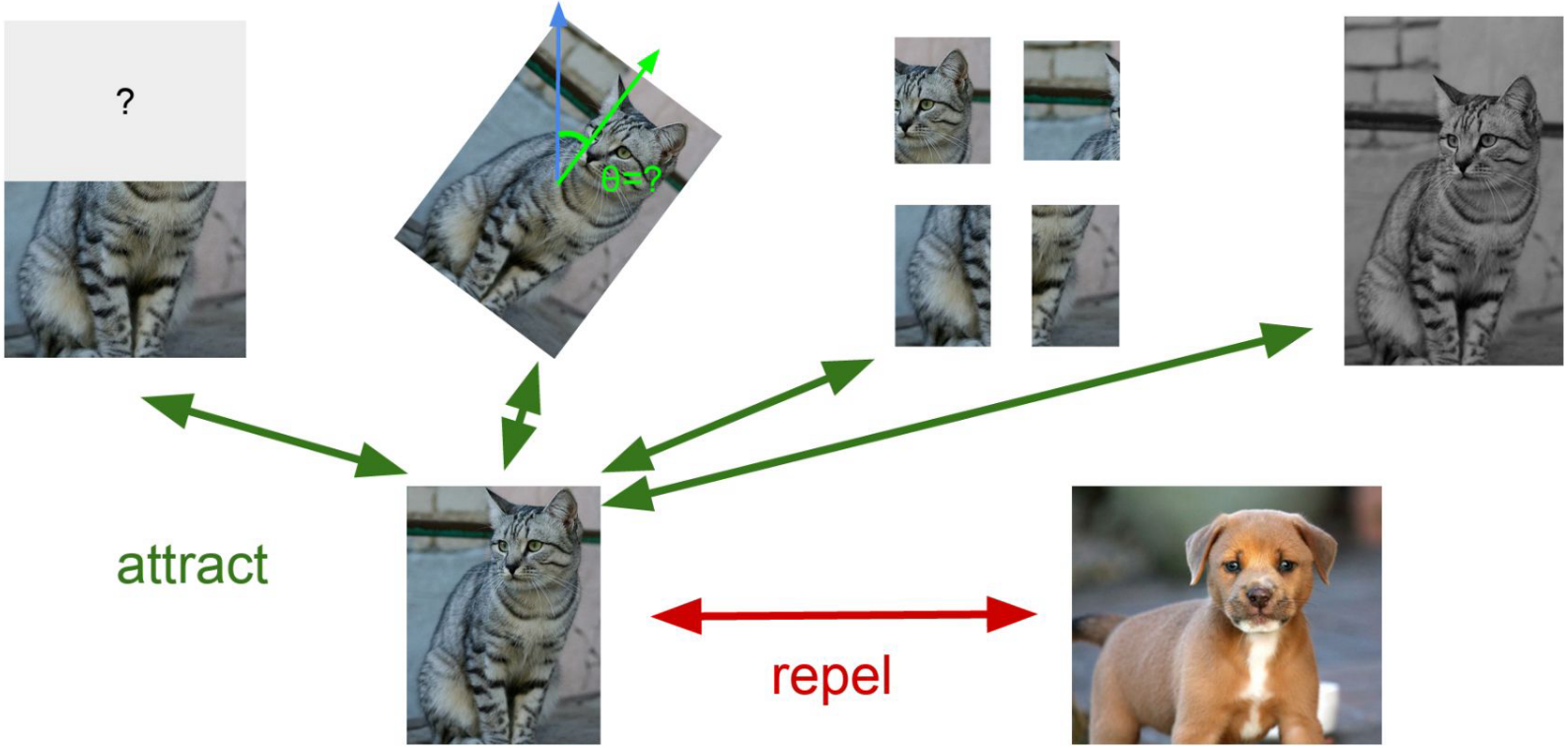
A more general pretext task?



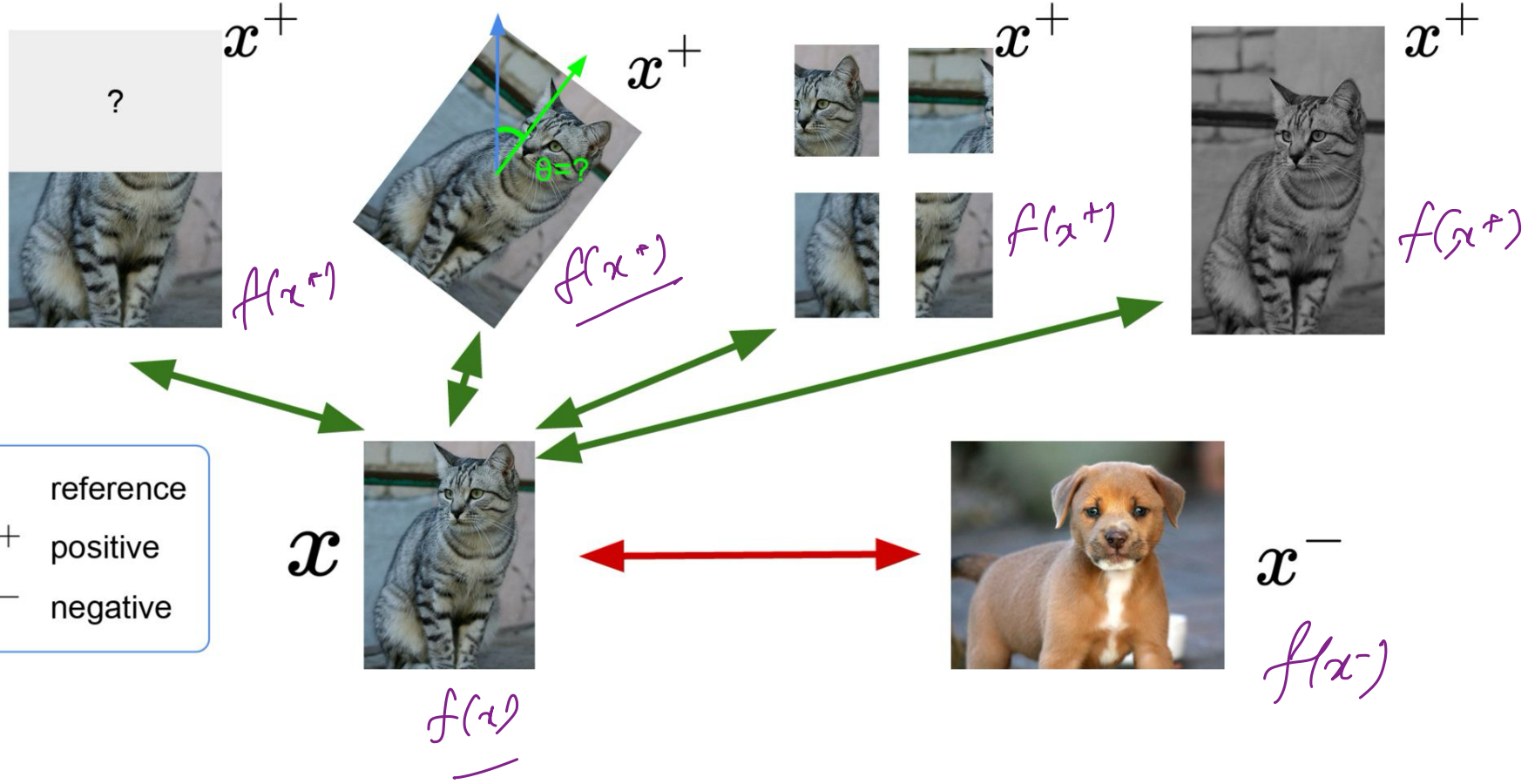
A more general pretext task?



Contrastive Representation Learning



Contrastive Representation Learning



A formulation of contrastive learning

What we want:

$$\text{score}(f(x), f(x^+)) >> \text{score}(f(x), f(x^-))$$

x: reference sample; x^+ positive sample; x^- negative sample

Given a chosen score function, we aim to learn an **encoder function** f that yields high score for positive pairs (x, x^+) and low scores for negative pairs (x, x^-).

$$Score(x, x^+) = Score(x, x^-) + \Delta/2$$

Triplet Loss

$$f(x, x^+, x^-) = Score(x, x^+) \quad Score(x, x^-)$$

max-margin loss

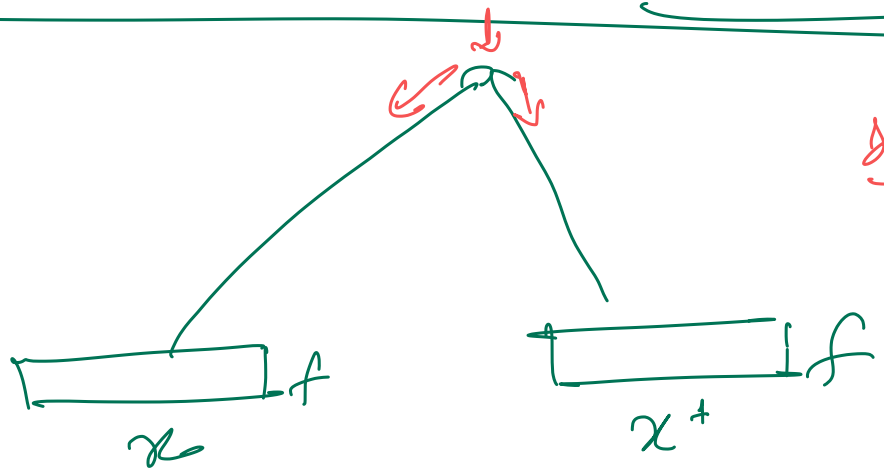
$$\max \frac{Score(x, x^+)}{Score(x, x^-)}$$

$$\mathcal{L} = \max\left(-Score(x, x^+) + Score(x, x^-) + \Delta, 0\right)$$

A formulation of contrastive learning

Loss function given 1 positive sample and N - 1 negative samples:

$$L = -\mathbb{E}_X \left[\log \frac{\exp(s(f(x), f(x^+))}{\exp(s(f(x), f(x^+)) + \sum_{j=1}^{N-1} \exp(s(f(x), f(x_j^-))} \right]$$

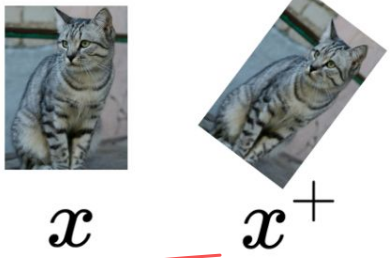


Siamese architecture

A formulation of contrastive learning

Loss function given 1 positive sample and N - 1 negative samples:

$$L = -\mathbb{E}_X \left[\log \frac{\exp(s(f(x), f(x^+))}{\exp(s(f(x), f(x^+)) + \sum_{j=1}^{N-1} \exp(s(f(x), f(x_j^-))} \right]$$



x x^+

x

x_1^-

x_2^-

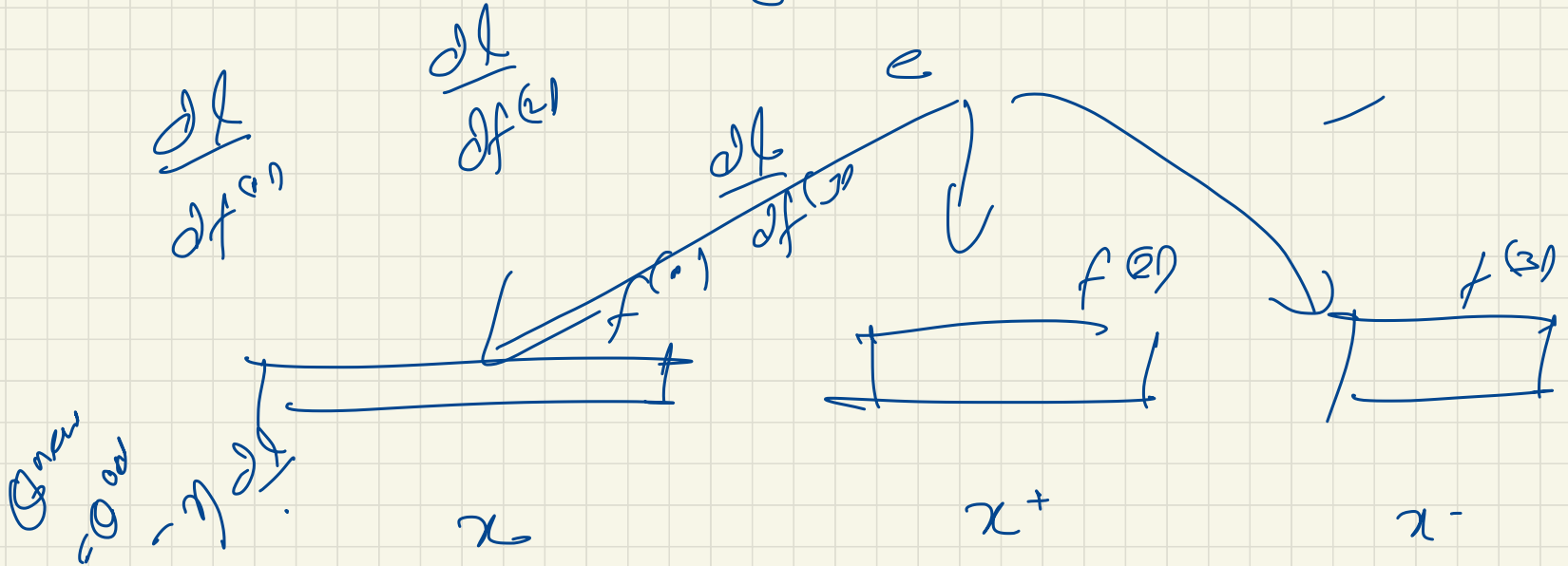
x_3^-

...

This seems familiar …

$s(f(x), f(x^+))$, $s(f(x), f(x_1^-))$ . . $s(f(x), f(x_{N-1}^-))$

$-\log \frac{e^{(s f(x), f(x^+))}}{}$



A formulation of contrastive learning

Loss function given 1 positive sample and N - 1 negative samples:

$$L = -\mathbb{E}_X \left[\log \frac{\exp(s(f(x), f(x^+))}{\exp(s(f(x), f(x^+)) + \sum_{j=1}^{N-1} \exp(s(f(x), f(x_j^-))} \right]$$

score for the positive pair

score for the N-1 negative pairs

This seems familiar …

Cross entropy loss for a N-way softmax classifier!

I.e., learn to find the positive sample from the N samples

SimCLR: A Simple Framework for Contrastive Learning

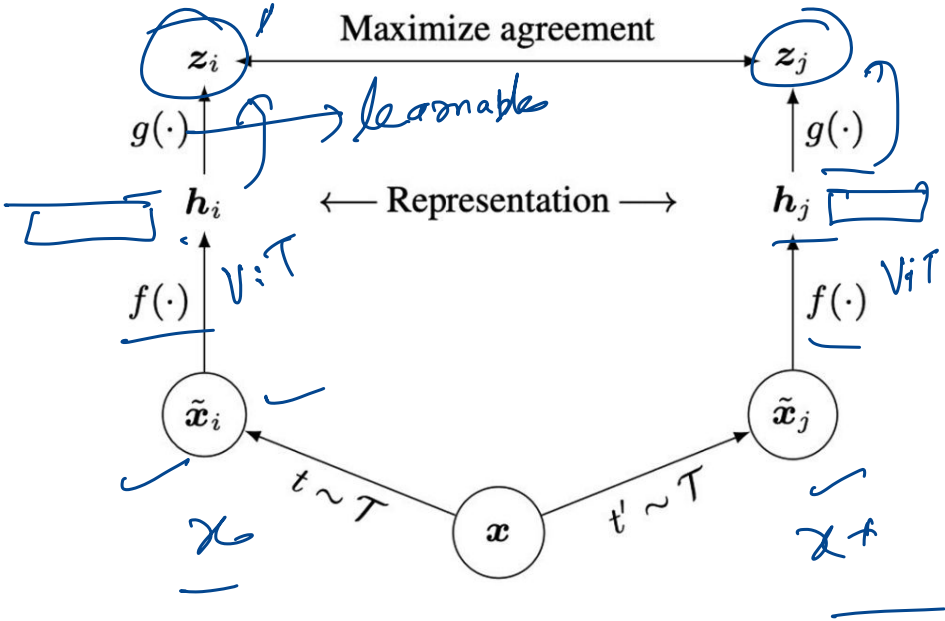
Cosine similarity as the score function:

$$s(u, v) = \frac{u^T v}{||u||||v||}$$

Use a projection network ***g(·)*** to project features to a space where contrastive learning is applied

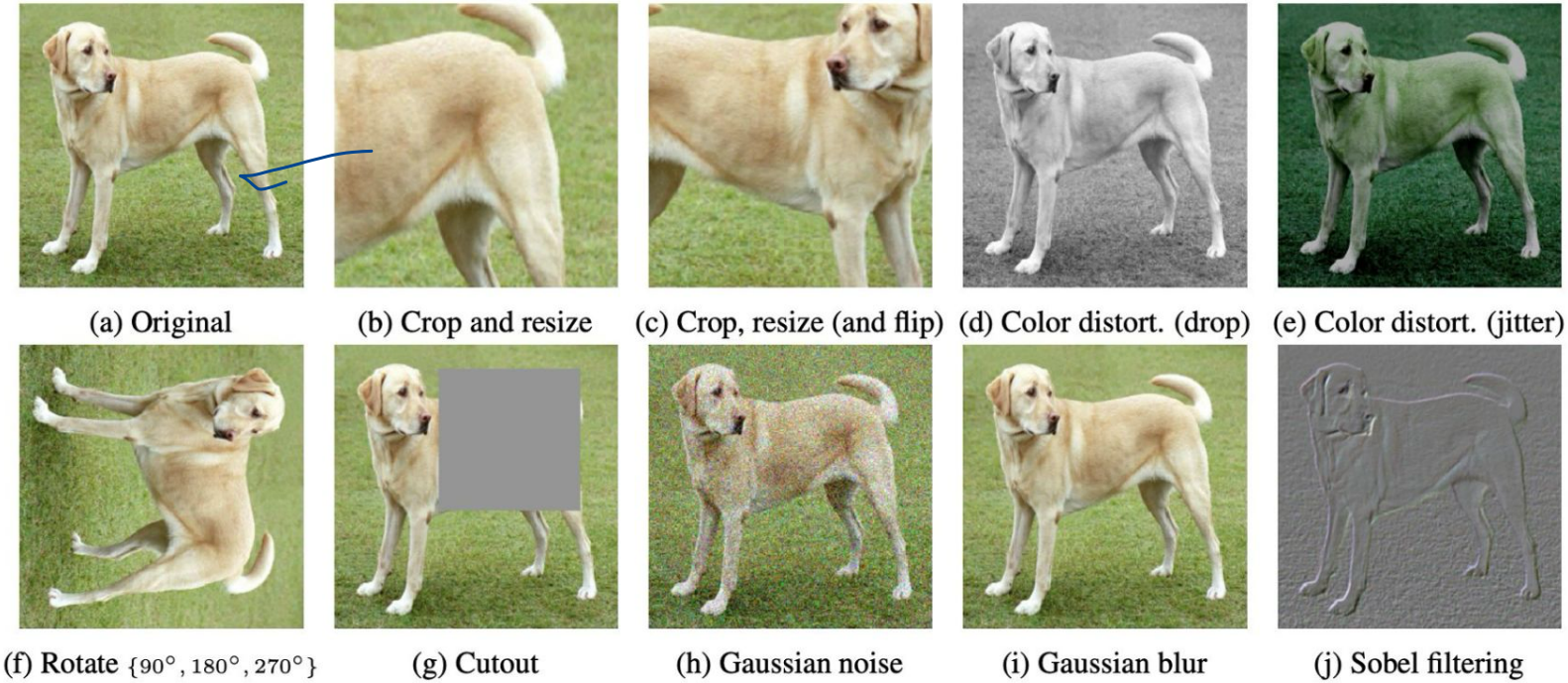
Generate positive samples through data augmentation:

- random cropping, random color distortion, and random blur.



Source: Chen et al., 2020

SimCLR: generating positive samples from data augmentation



(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

Source: Chen et al., 2020

SimCLR

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, constant τ, structure of $f, g, \mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}, t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i, j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

Generate a positive pair by sampling data augmentation functions

Image 1

2N images

each image → 1 +ve, 2N-2 -ve

2N-1

2N-1

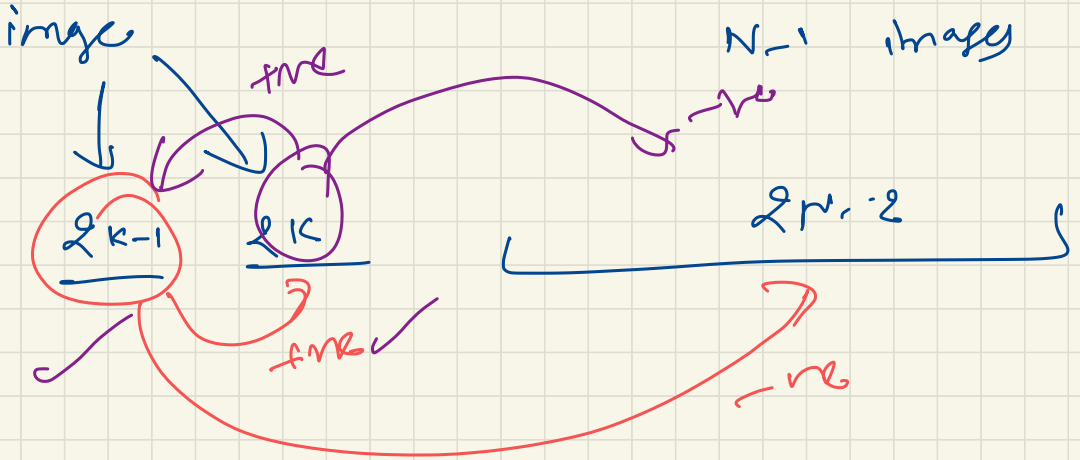


image
+ve
N-1 images
-ve
2k-1
2k
2N-2
+ve
-ve

SimCLR

Algorithm 1 SimCLR's main learning algorithm.

input: batch size N, constant τ, structure of $f, g, \mathcal{T}$.
for sampled minibatch $\{\boldsymbol{x}_k\}_{k=1}^{N}$ **do**
  for all $k \in \{1, \ldots, N\}$ **do**
    draw two augmentation functions $t \sim \mathcal{T}, t' \sim \mathcal{T}$
    # the first augmentation
    $\tilde{\boldsymbol{x}}_{2k-1} = t(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k-1} = f(\tilde{\boldsymbol{x}}_{2k-1})$ # representation
    $\boldsymbol{z}_{2k-1} = g(\boldsymbol{h}_{2k-1})$ # projection
    # the second augmentation
    $\tilde{\boldsymbol{x}}_{2k} = t'(\boldsymbol{x}_k)$
    $\boldsymbol{h}_{2k} = f(\tilde{\boldsymbol{x}}_{2k})$ # representation
    $\boldsymbol{z}_{2k} = g(\boldsymbol{h}_{2k})$ # projection
  end for
  for all $i \in \{1, \ldots, 2N\}$ and $j \in \{1, \ldots, 2N\}$ **do**
    $s_{i,j} = \boldsymbol{z}_i^\top \boldsymbol{z}_j / (\|\boldsymbol{z}_i\| \|\boldsymbol{z}_j\|)$ # pairwise similarity
  end for
  define $\ell(i, j)$ **as** $\ell(i,j) = -\log \frac{\exp(s_{i,j}/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(s_{i,k}/\tau)}$
  $\mathcal{L} = \frac{1}{2N} \sum_{k=1}^{N} [\ell(2k-1, 2k) + \ell(2k, 2k-1)]$
  update networks f and g to minimize $\mathcal{L}$
end for
return encoder network $f(\cdot)$, and throw away $g(\cdot)$

Generate a positive pair by sampling data augmentation functions

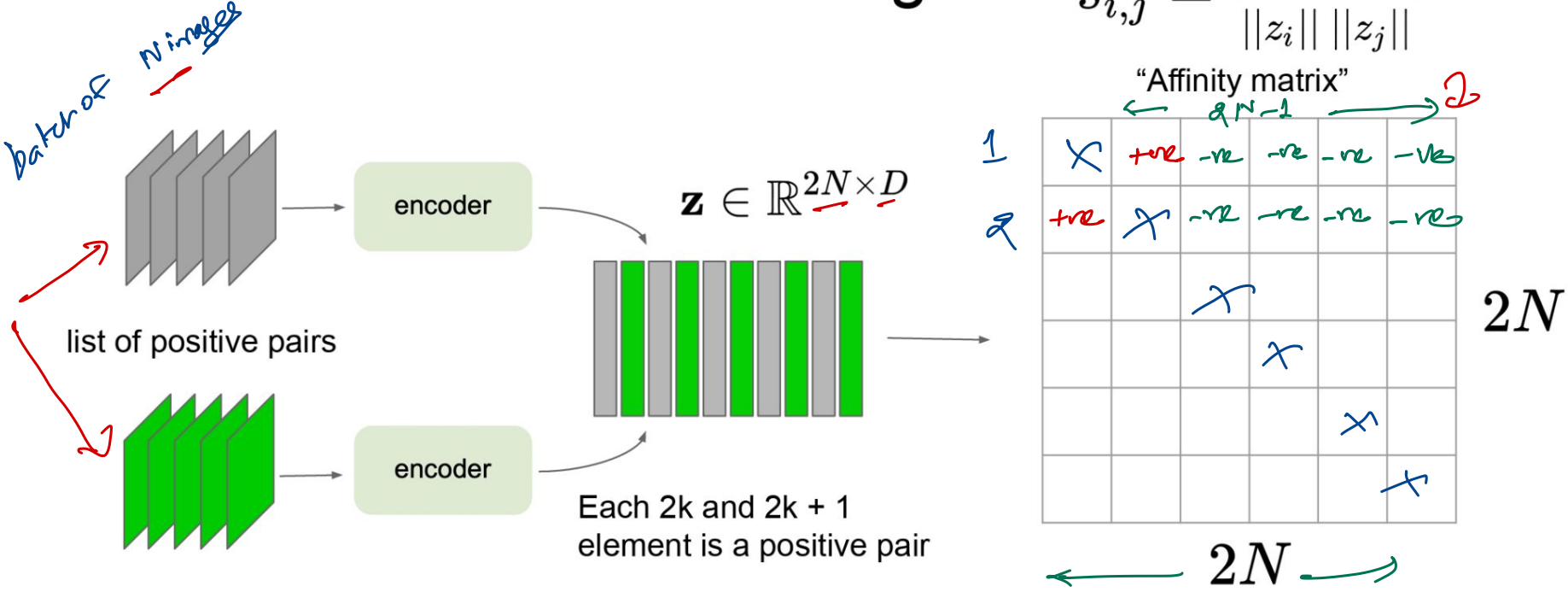
Iterate through and use each of the 2N sample as reference, compute average loss

InfoNCE loss: Use all non-positive samples in the batch as x^-

Source: Chen et al., 2020

SimCLR: mini-batch training

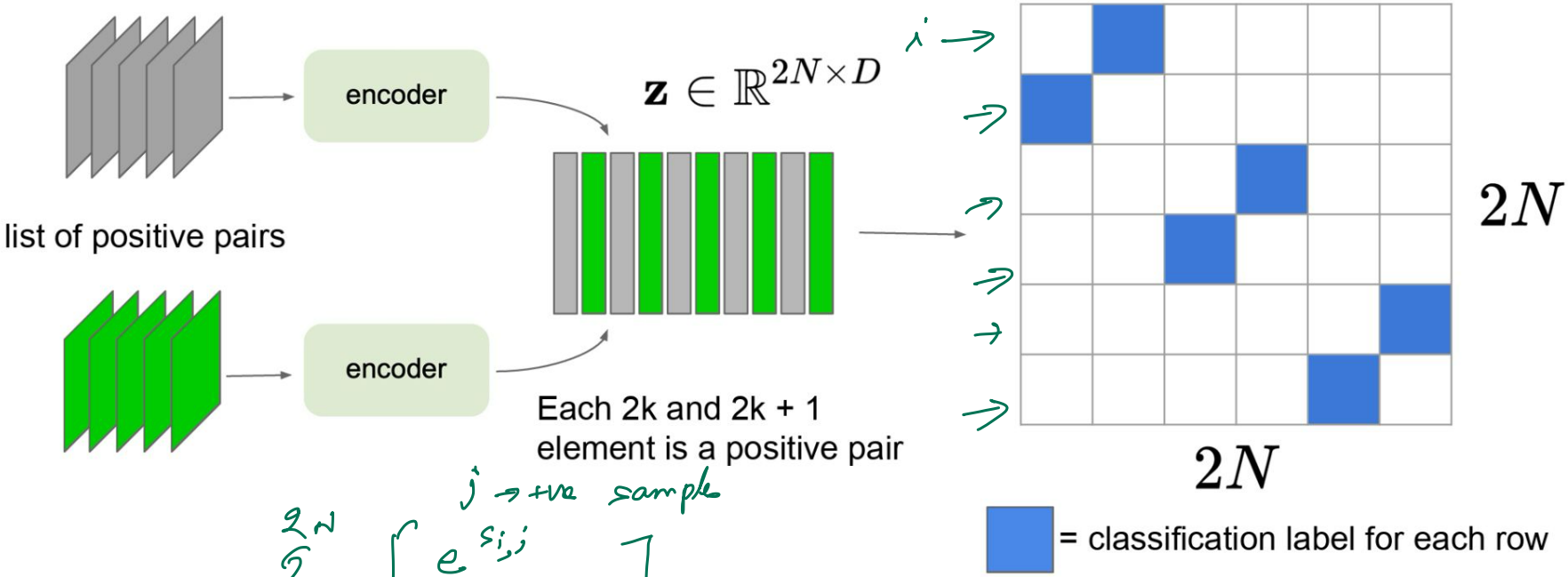
$$s_{i,j} = \frac{z_i^T z_j}{||z_i|| \, ||z_j||}$$



SimCLR: mini-batch training

$$s_{i,j} = \frac{z_i^T z_j}{||z_i||\,||z_j||}$$

"Affinity matrix"



$$\sum_{i=1}^{2N} \left[\frac{e^{s_{i,j}}}{\sum_{k \neq i} e^{s_{i,k}}} \right]$$

$j \rightarrow$ +ve sample

Semi-supervised learning on SimCLR features

Method	Architecture	Label fraction (Top 5) 1%	10%
Supervised baseline	ResNet-50	48.4	80.4
Methods using other label-propagation:			
Pseudo-label	ResNet-50	51.6	82.4
VAT+Entropy Min.	ResNet-50	47.0	83.4
UDA (w. RandAug)	ResNet-50	-	88.5
FixMatch (w. RandAug)	ResNet-50	-	89.1
S4L (Rot+VAT+En. M.)	ResNet-50 (4×)	-	91.2
Methods using representation learning only:			
InstDisc	ResNet-50	39.2	77.4
BigBiGAN	RevNet-50 (4×)	55.2	78.8
PIRL	ResNet-50	57.2	83.8
CPC v2	ResNet-161(*)	77.9	91.2
SimCLR (ours)	ResNet-50	75.5	87.8
SimCLR (ours)	ResNet-50 (2×)	83.0	91.2
SimCLR (ours)	ResNet-50 (4×)	**85.8**	**92.6**

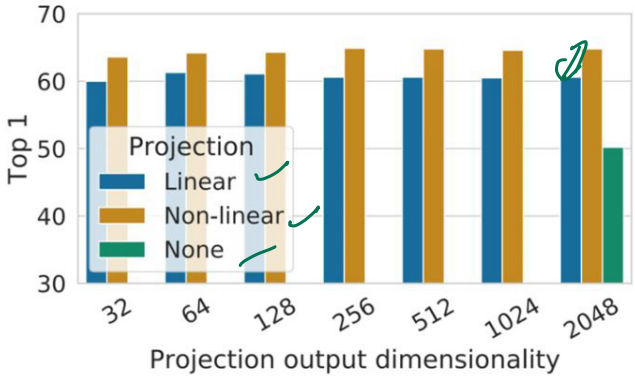
Table 7. ImageNet accuracy of models trained with few labels.

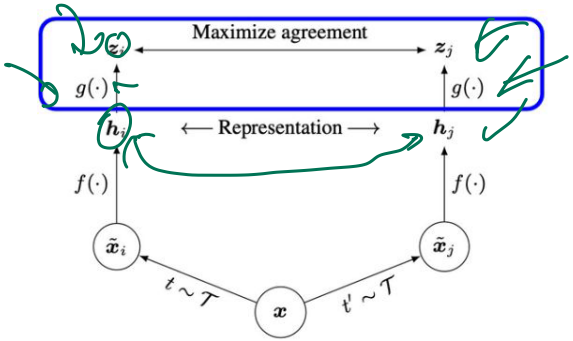
Train feature encoder on **ImageNet** (entire training set) using SimCLR.

Finetune the encoder with 1% / 10% of labeled data on ImageNet.

Source: Chen et al., 2020

SimCLR design choices: projection head





Linear / non-linear projection heads improve representation learning.

A possible explanation:

- contrastive learning objective may discard useful information for downstream tasks
- representation space **z** is trained to be invariant to data transformation.
- by leveraging the projection head **g(·)**, more information can be preserved in the **h** representation space

Source: Chen et al., 2020

Vision Language Pretraining

Okay, so we can encode text with Transformers, and we can encode images with Transformers….

Since the architectures are now basically the same, can we train a single model on both modalities?

Goal : Zero-shot image classification

Classes

names of the classes

- animal
- bird
- human
- car

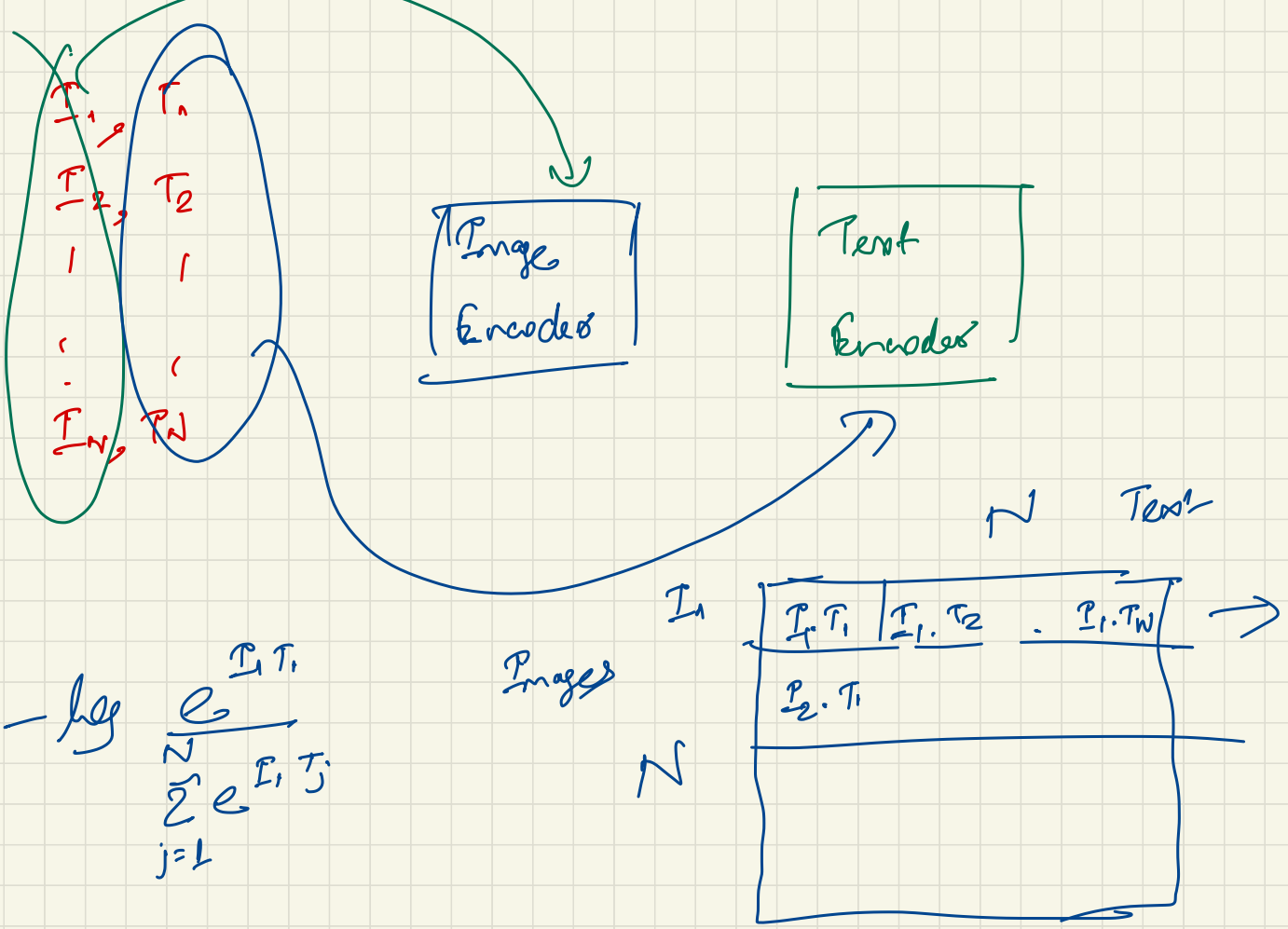
This is an image of an animal

a bird

Contrastive Language-Image Pretraining (CLIP)

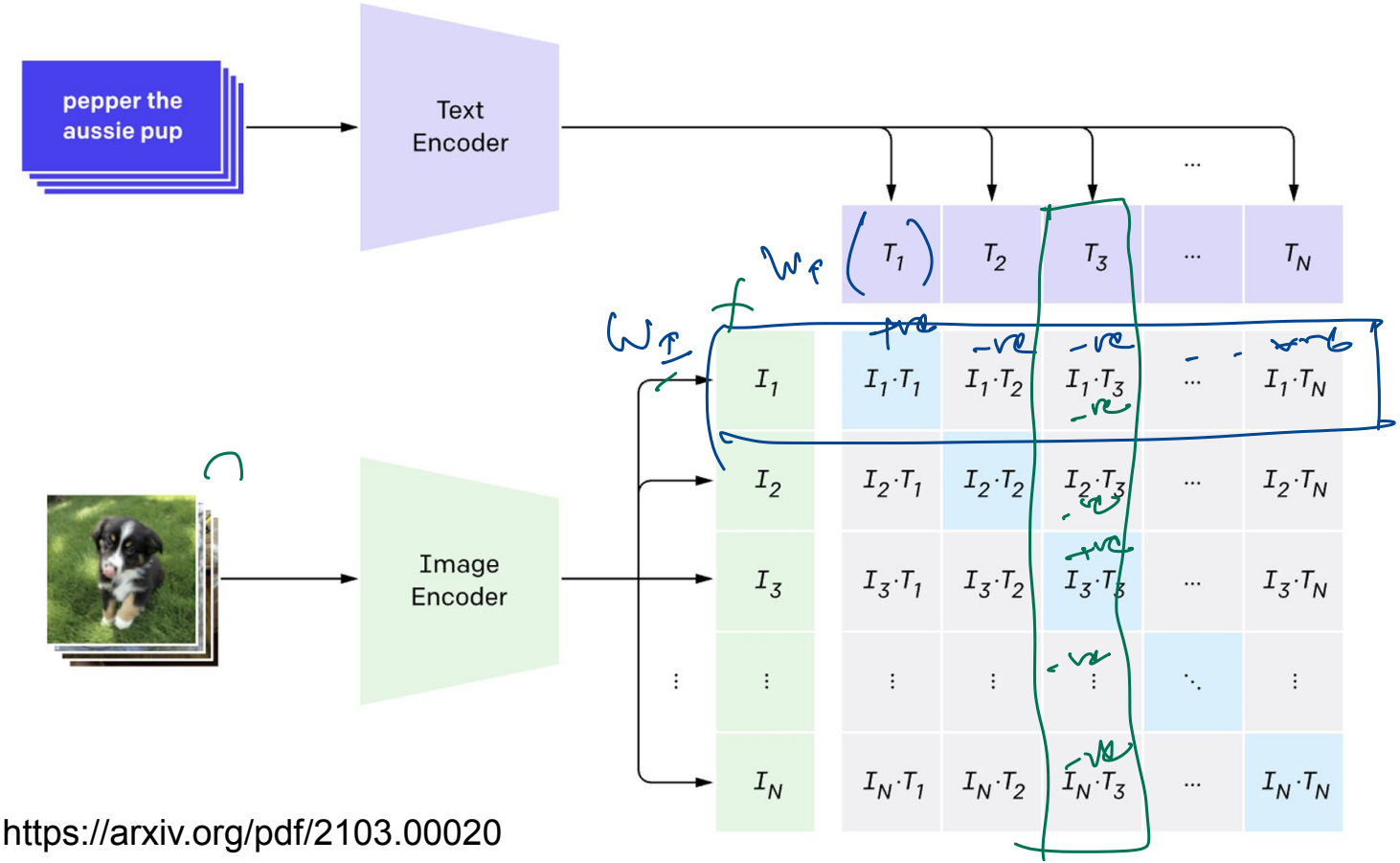
- OpenAI collect 400 million (image, text) pairs from the web

- Then, they train an image encoder and a text encoder with a simple contrastive loss: given a collection of images and text, predict which (image, text) pairs actually occurred in the dataset



$$-\log \frac{e^{I_1 \cdot T_1}}{\sum_{j=1}^{N} e^{I_1 \cdot T_j}}$$

Contrastive Language-Image Pretraining (CLIP)



https://arxiv.org/pdf/2103.00020

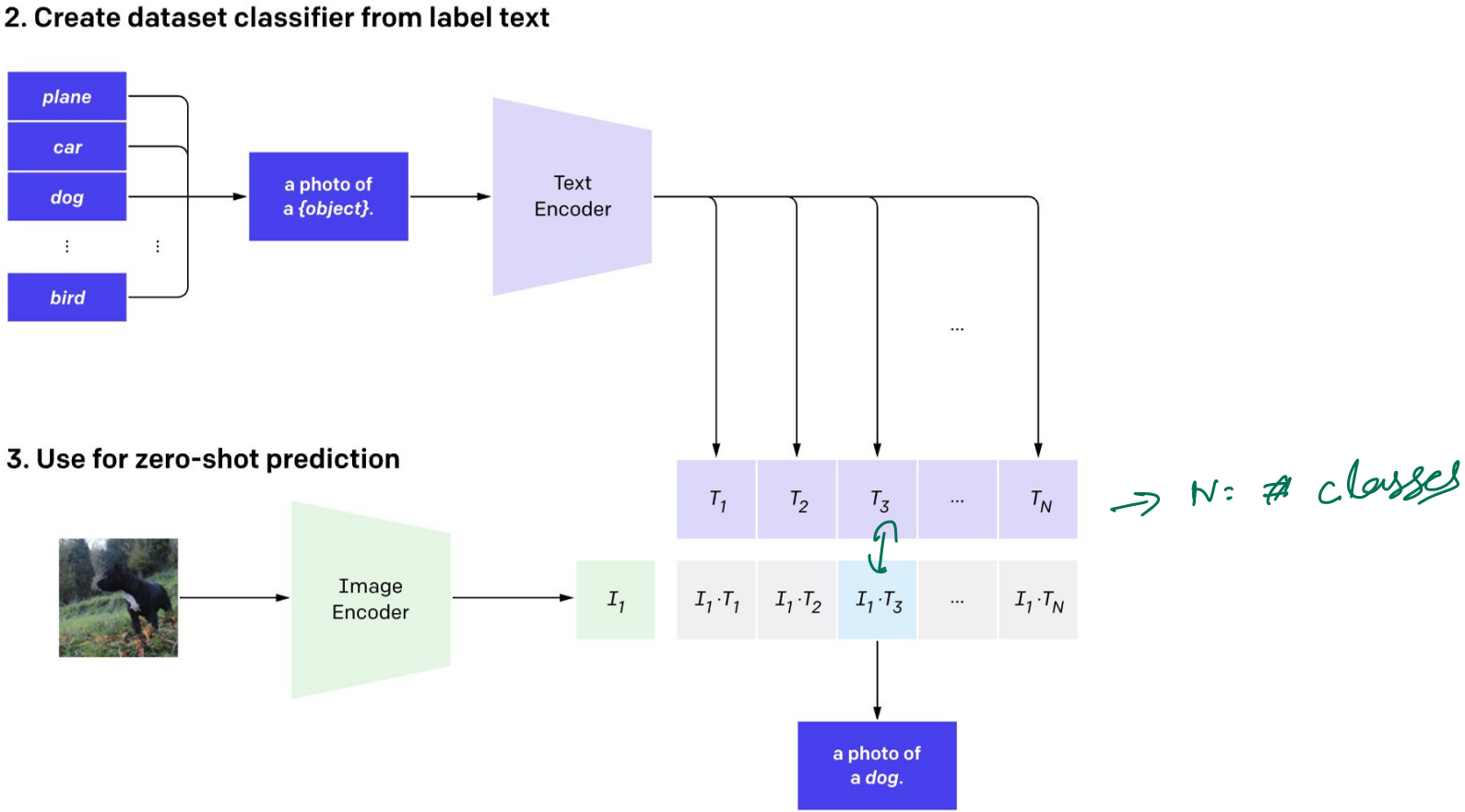
CLIP pseudocode

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# 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
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

https://arxiv.org/pdf/2103.00020

CLIP for zero-shot learning (Image Classification)



Example outputs

SUN397

television studio (90.2%) Ranked 1 out of 397 labels

✓ a photo of a **television studio**.

× a photo of a **podium indoor**.

× a photo of a **conference room**.

× a photo of a **lecture room**.

× a photo of a **control room**.

Other tricks

What if we know that the dataset is about pets?

Prompt: “A photo of a {label}, a type of pet”.

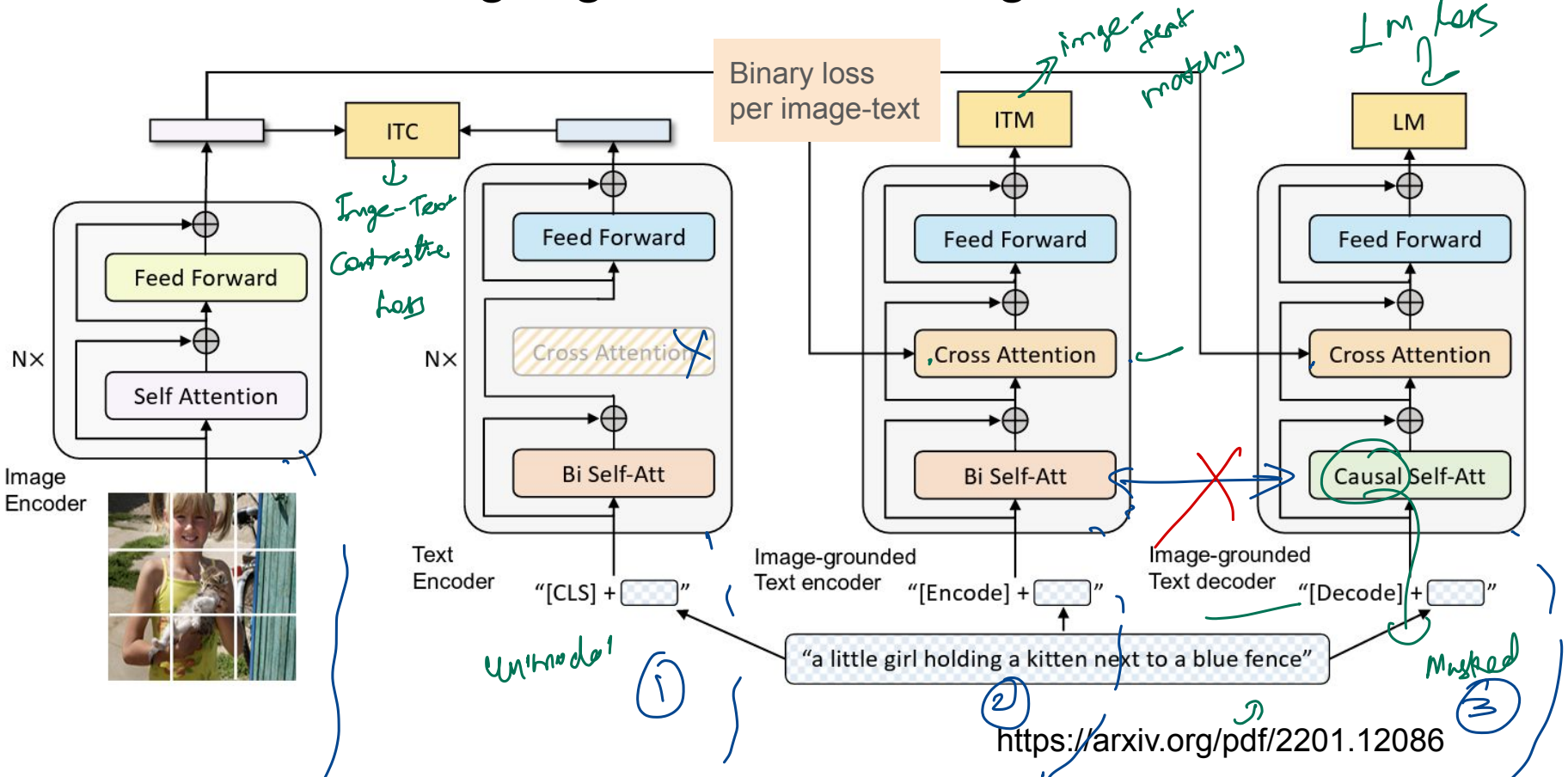
Satellite image classification dataset

Prompt: “a satellite photo of a {label}”.

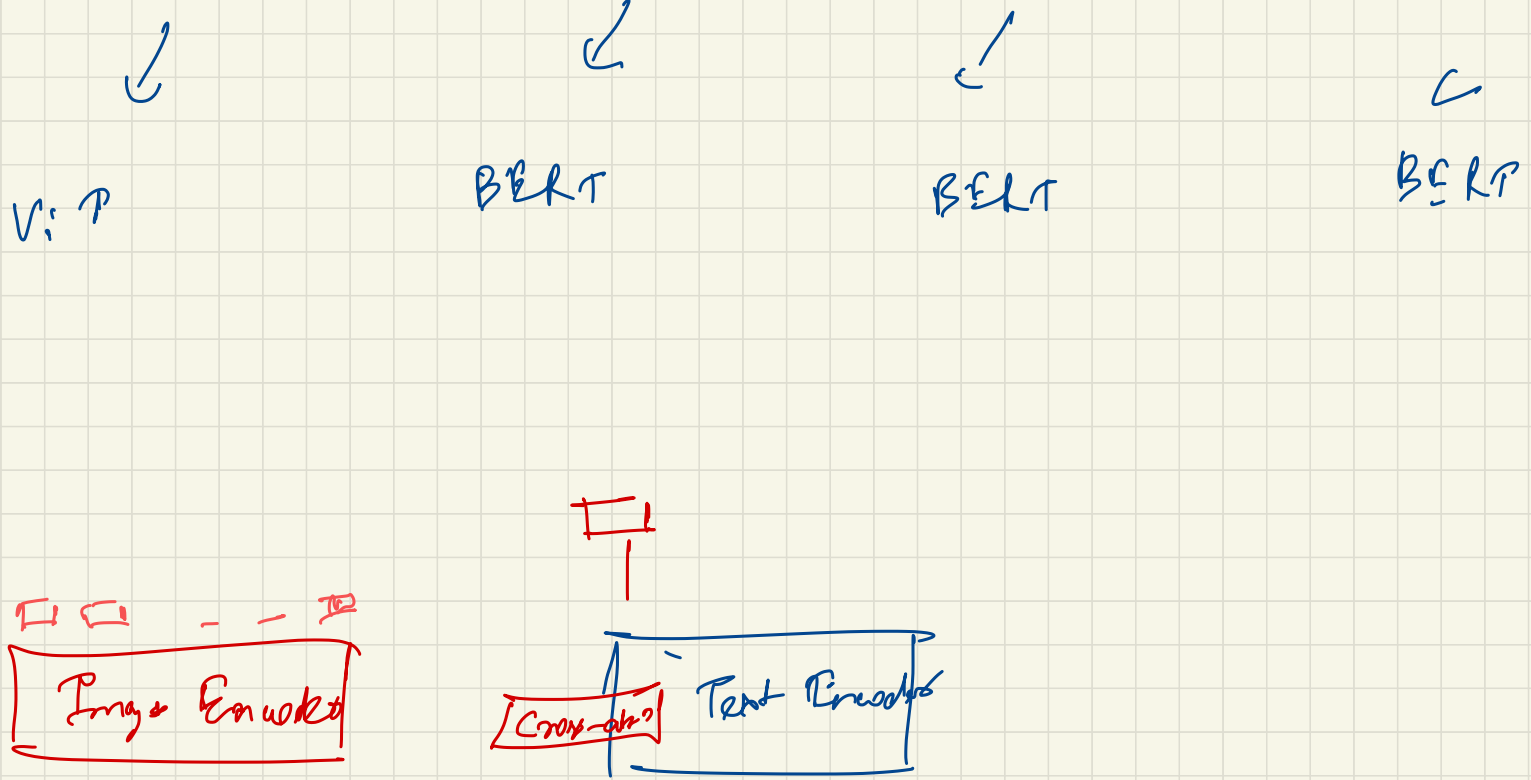
Ensembling over multiple zero-shot classifiers

Using different context prompts such as ‘A photo of a big {label}’ and “A photo of a small {label}”.
We construct the ensemble over the embedding space instead of probability space.

BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation



https://arxiv.org/pdf/2201.12086


ViT
BERT
BERT
BERT
Image Encoder
Cross-attn
Text Encoder
Image
Text

BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation

Multimodal mixture of encoder-decoder, a unified vision-language model which can operate in one of three functionalities:

1. Unimodal encoder is trained with an ***image-text contrastive (ITC)*** loss to align the vision and language representations.
2. Image-grounded text encoder uses additional cross-attention layers to model vision-language interactions and is trained with an ***image-text matching (ITM)*** loss to distinguish between positive and negative image-text pairs.
3. Image-grounded text decoder replaces the bi-directional self-attention layers with causal self-attention layers and shares the same cross-attention layers and feed-forward networks as the encoder. The decoder is trained with a language modeling (LM) loss to generate captions given images.

Effect of Parameter sharing

During pre-training, the text encoder and decoder share all parameters except for the self-attention layers.

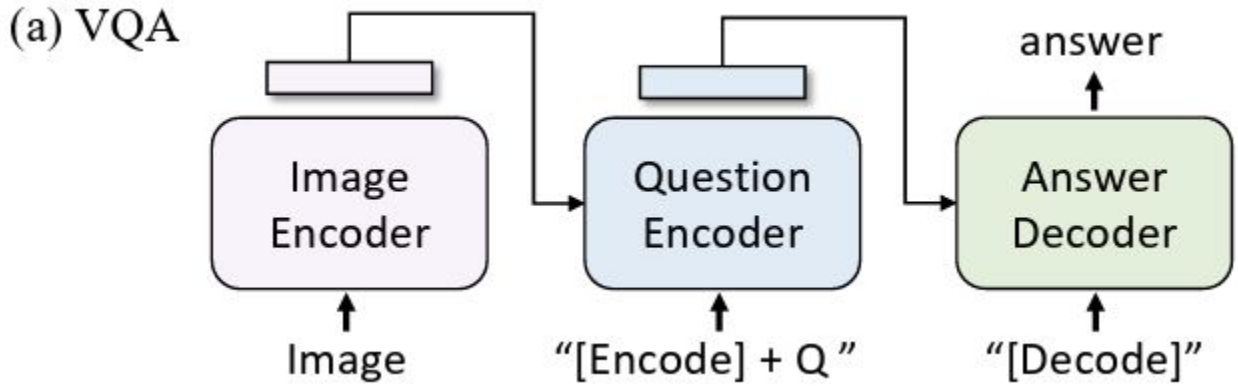
Layers shared	#parameters	Retrieval-FT (COCO)		Retrieval-ZS (Flickr)		Caption-FT (COCO)		Caption-ZS (NoCaps)	
		TR@1	IR@1	TR@1	IR@1	B@4	CIDEr	CIDEr	SPICE
All	224M	77.3	59.5	93.1	81.0	37.2	125.9	100.9	13.1
All except CA	252M	77.5	59.9	93.1	81.3	37.4	126.1	101.2	13.1
All except SA	252M	78.4	60.7	93.9	82.1	38.0	127.8	102.2	13.9
None	361M	78.3	60.5	93.6	81.9	37.8	127.4	101.8	13.9

Table 3. Comparison between different parameter sharing strategies for the text encoder and decoder during pre-training.

If the SA layers are shared, the model's performance would degrade due to the conflict between the encoding task and the decoding task.

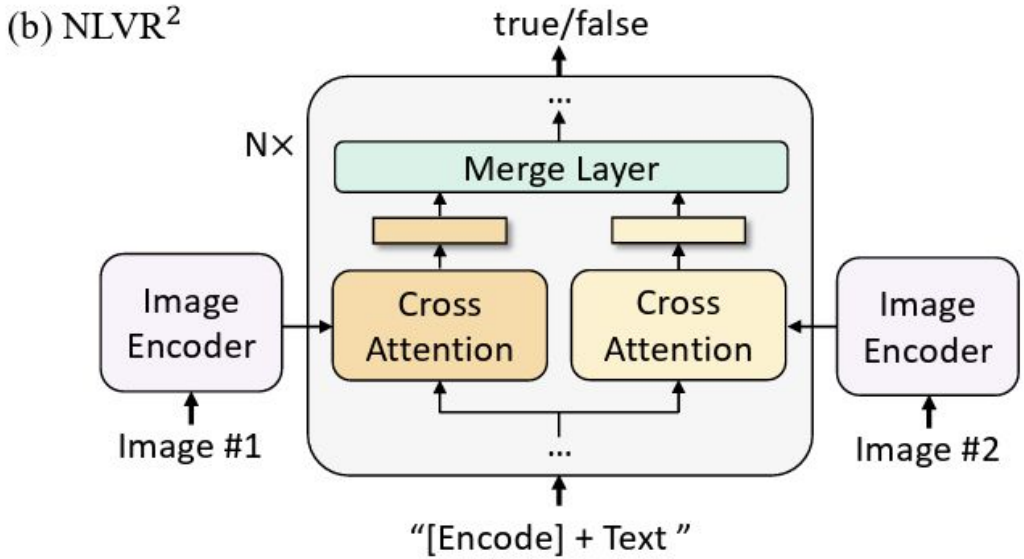
Performing tasks: Visual Question Answering

VQA requires the model to predict an answer given an image and a question.
Treat it as an answer generation task given the image and question.



Performing tasks: Natural Language Visual Reasoning

NLVR asks the model to predict whether a sentence describes a pair of images.



For each transformer block in the image-grounded text encoder, there exist two cross-attention layers to process the two input images, and their outputs are merged and fed to the FFN.

The two CA layers are initialized from the same pre-trained weights.

The merge layer performs

- simple average pooling in the first six layers of the encoder and
- performs concatenation followed by a linear projection in layers 6-12.

An MLP classifier is applied on the output embedding of the [Encode] token.