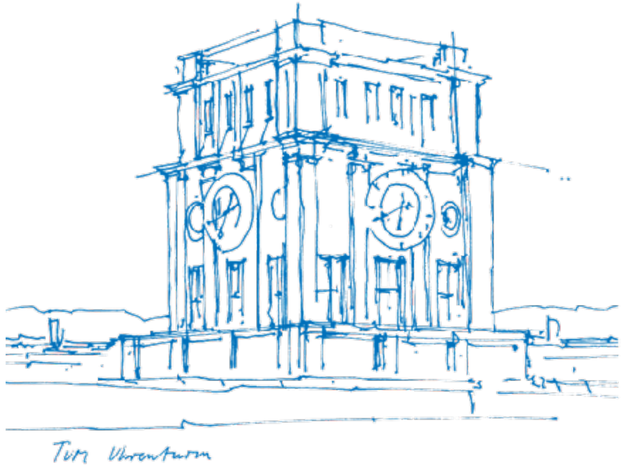
Computer Vision III:

Semi- and Unsupervised learning

Nikita Araslanov

24.01.2023



Course progress

1. Introduction
2. Object detection 1
3. Object detection 2
4. Multiple object tracking 1
5. Multiple object tracking 2
6. Semantic segmentation
7. Instance segmentation
8. Panoptic segmentation
9. Video object segmentation
10. Transformers
11. Unsupervised and semi-supervised learning (24.01) ← we are here
12. Q&A (February 21 in Zoom – further details on Moodle)
13. Exam: 03.03

Attention is all you need

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

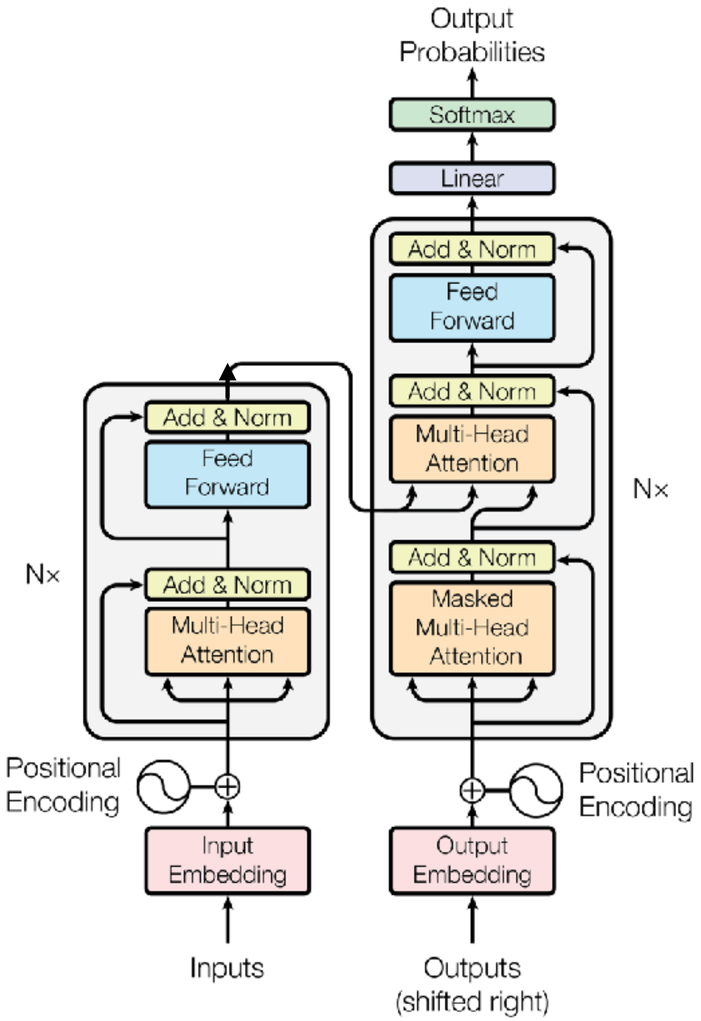
Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

Illia Polosukhin* ‡ illia.polosukhin@gmail.com



So, what is so special about it?

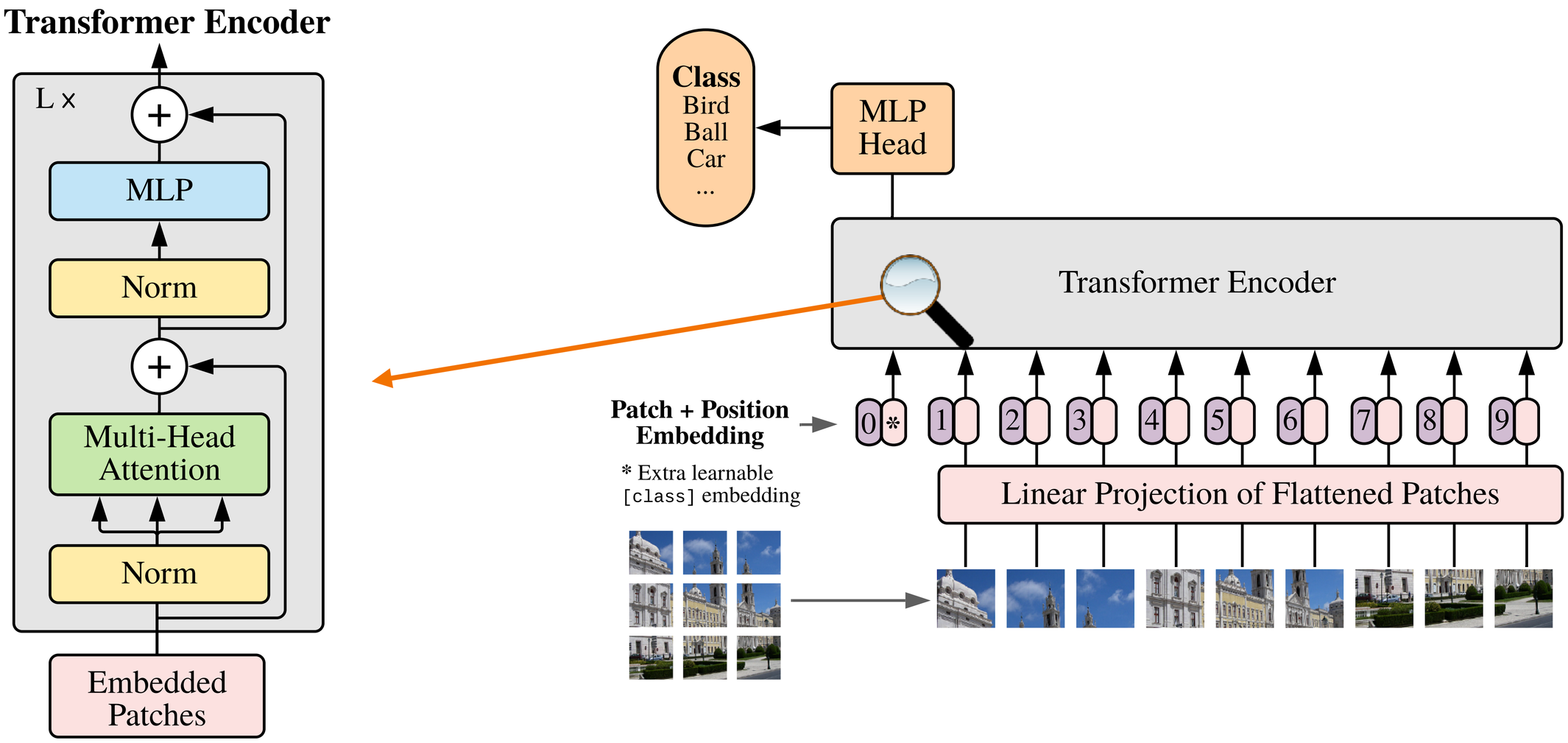
Universality

Universality: developing components that work across all possible settings.

- Consider other modalities (language, speech, etc.);
- Modern deep learning are only partially universal, even in the contexts we have seen so far (detection and segmentation);
- Transformers can be applied to language and images with excellent results in both domains!

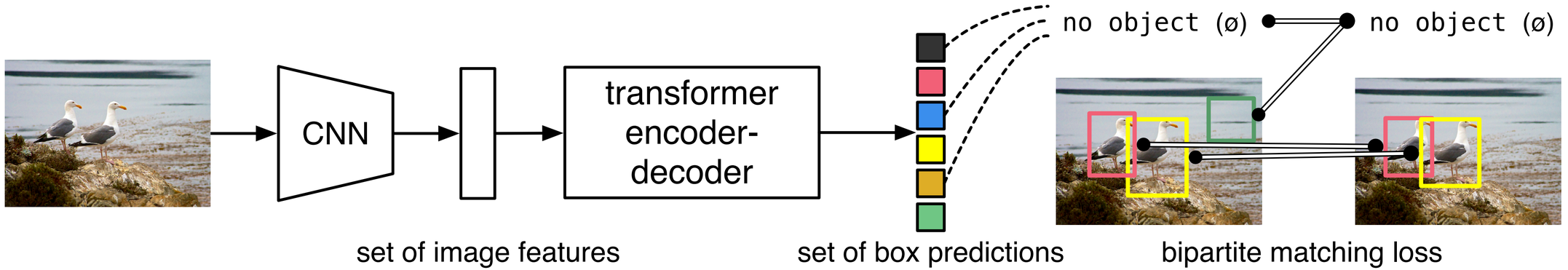
[Adapted from: Vaswani et al., ICML '21]

ViT Encoder



(Dosovitskiy et al., 2020)

Detection: DETR



- The CNN predicts local feature embeddings.
- The Transformer predicts the bounding boxes in parallel.
- During training, we uniquely assigns predictions to ground truth boxes with Hungarian matching.
- No need for non-maximum suppression.

DETR: Summary

- Accurate detection with a (relatively) simple architecture.
- No need for non-maximum suppression
  - We can simply disregard bounding boxes with “empty” class, or with low confidence.
- Accurate panoptic segmentation with a minor extension.

Issues:

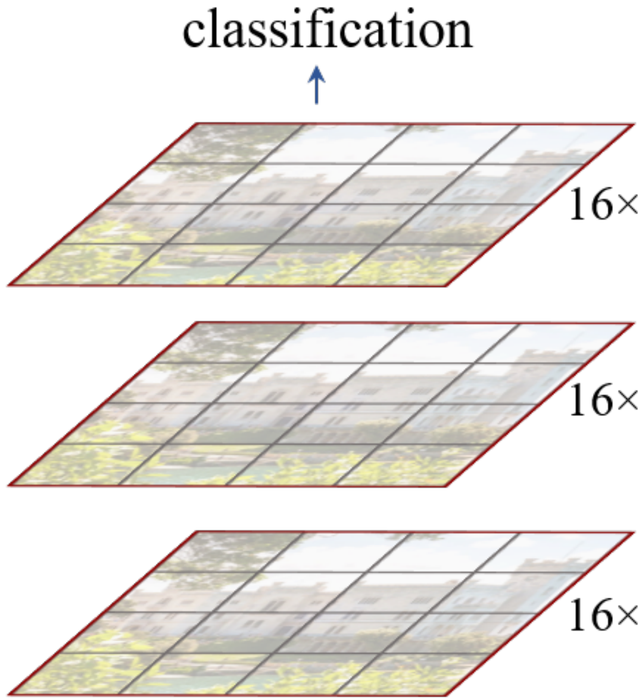
- High computational and memory requirements (especially the memory).
- Slow convergence / long training.

See: Zhu et al., “Deformable DETR: Deformable Transformers for End-to-End Object Detection” (ICLR 2021).

Swin Transformer

Recall ViT:

- We process patch embeddings with a series of self-attention layers;
  - Quadratic computational complexity.
- The number of tokens remains the same in all layers (QUIZ: How many / what does it depend on?)
- In CNNs, we gradually decrease the resolution, which has computation benefits (and increases the receptive field size).
- Do ViTs benefit from the same strategy?

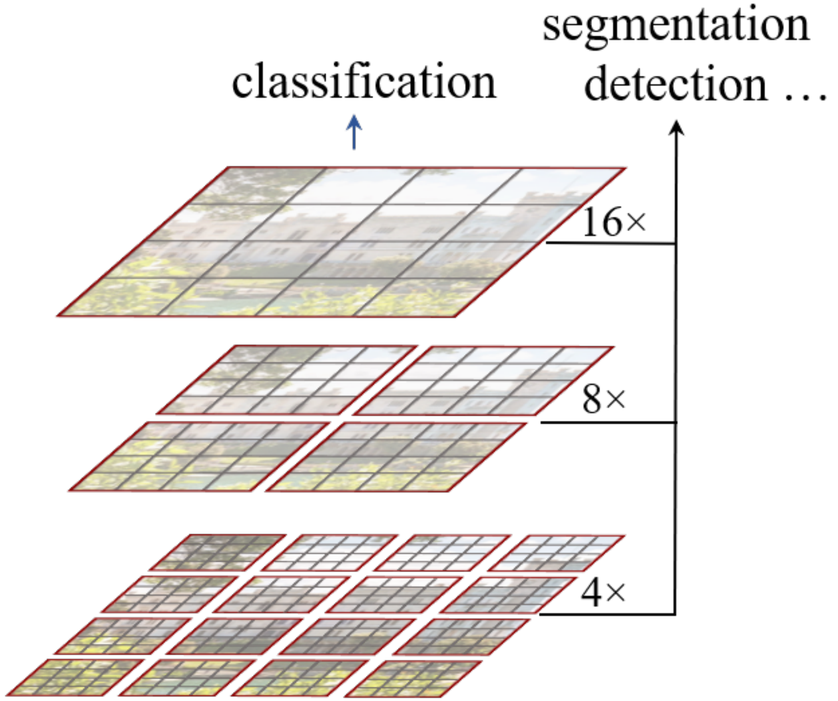


Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” (2021)

Swin Transformer

Swin Transformer:

- Local attention windows with a fixed size.
- Construct a hierarchy of image patches.
- Linear computational complexity.
- Output representation is compatible with standard backbones (e.g. ResNet)
  - We can test many downstream tasks (e.g. object detection)

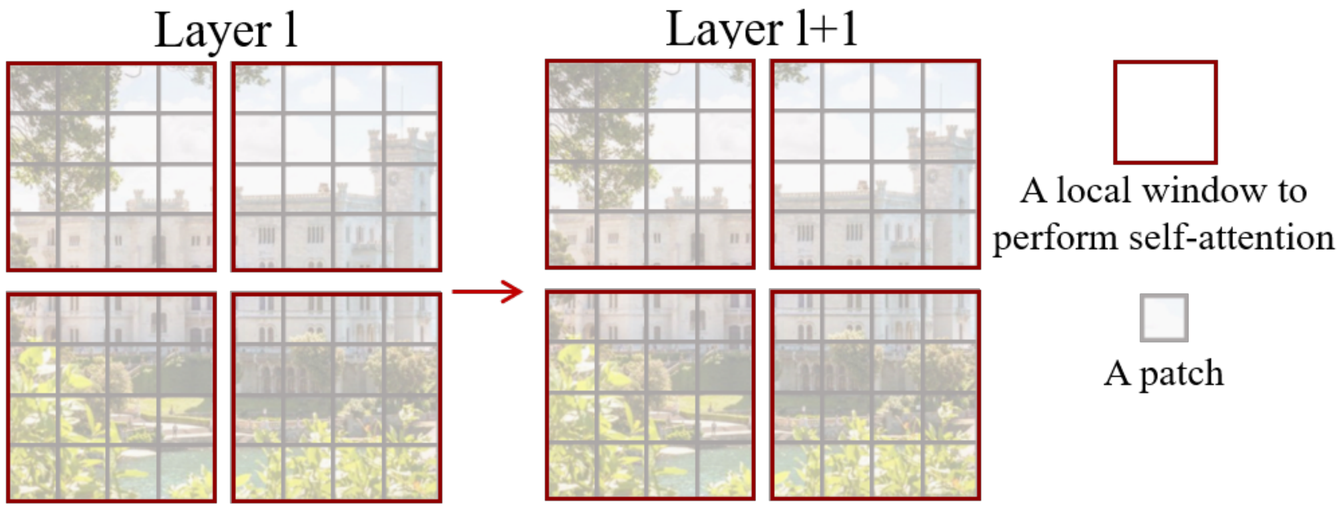


Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” (2021)

Swin Transformer

A naive implementation will hurt context reasoning:

- At any given level (except the last one), our context is not global anymore:

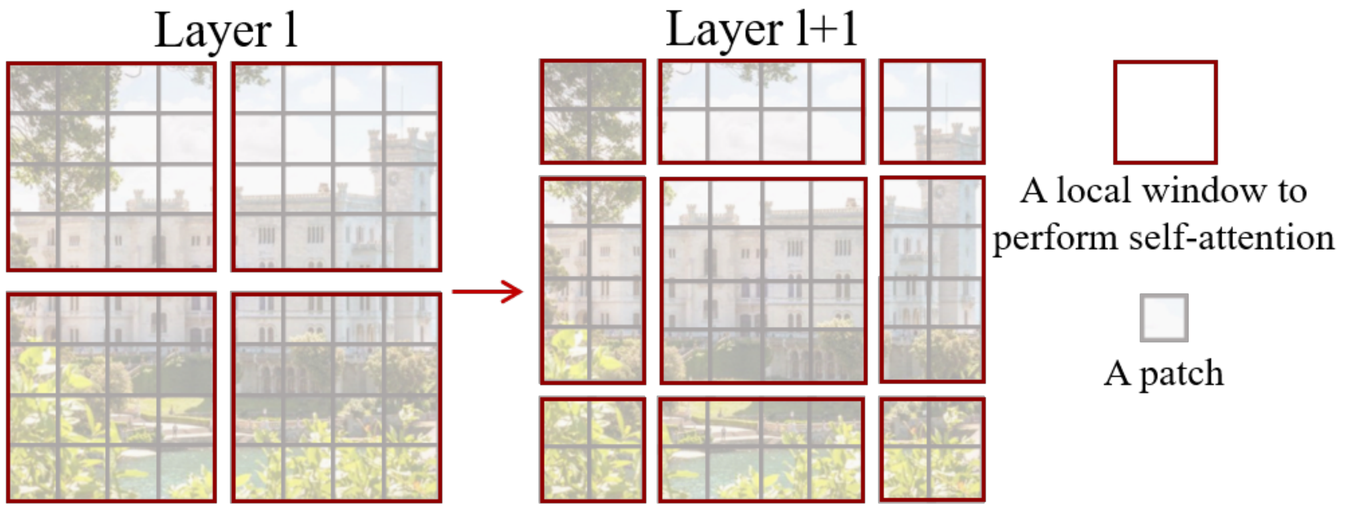


Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” (2021)

Swin Transformer

A naive implementation will hurt context reasoning:

- Solution: alternate the layout of local windows:

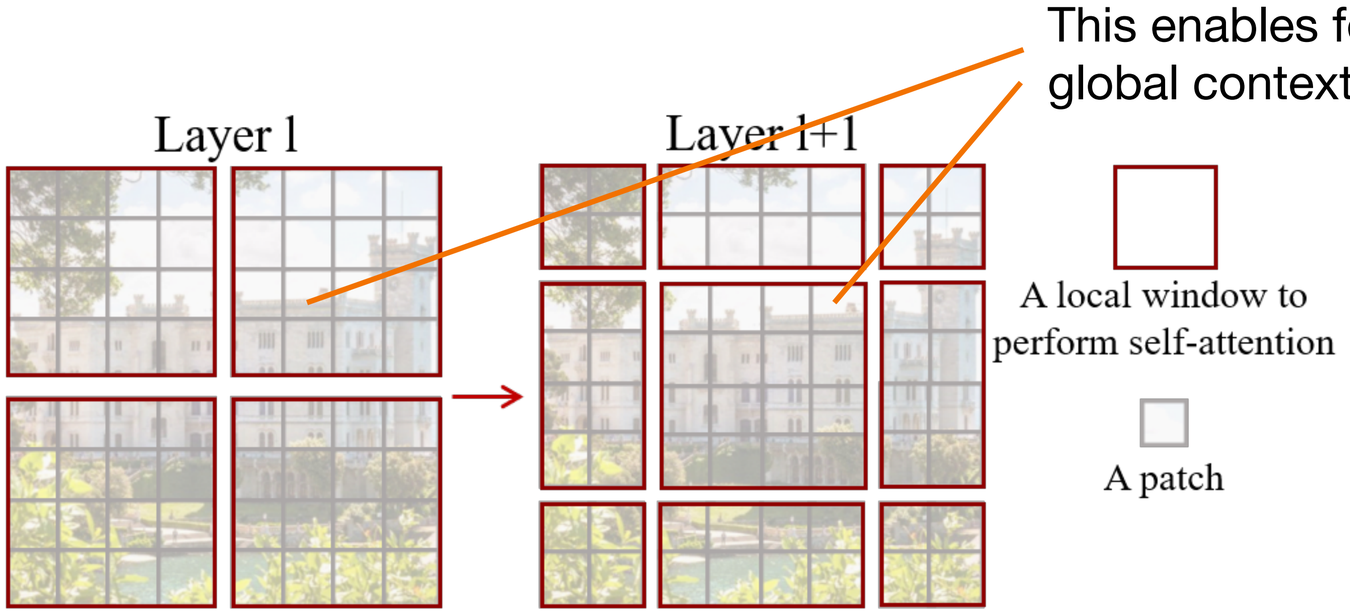


Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” (2021)

Swin Transformer

A naive implementation will hurt context reasoning:

- Solution: alternate the layout of local windows:

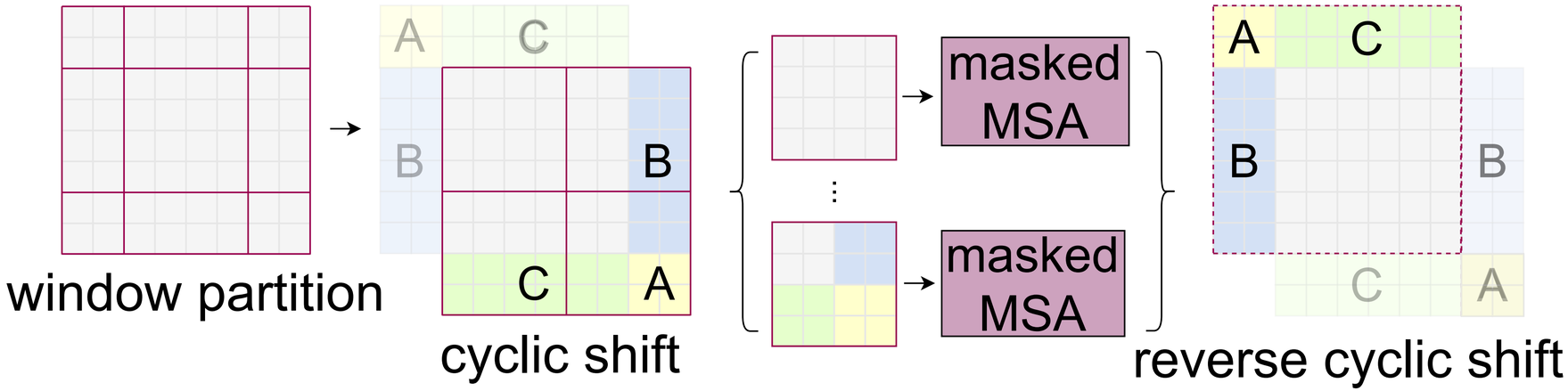


Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” (2021)

Swin Transformer

Efficient implementation of overlapping attention windows

- equivalent to using “circular” padding (see PyTorch documentation).

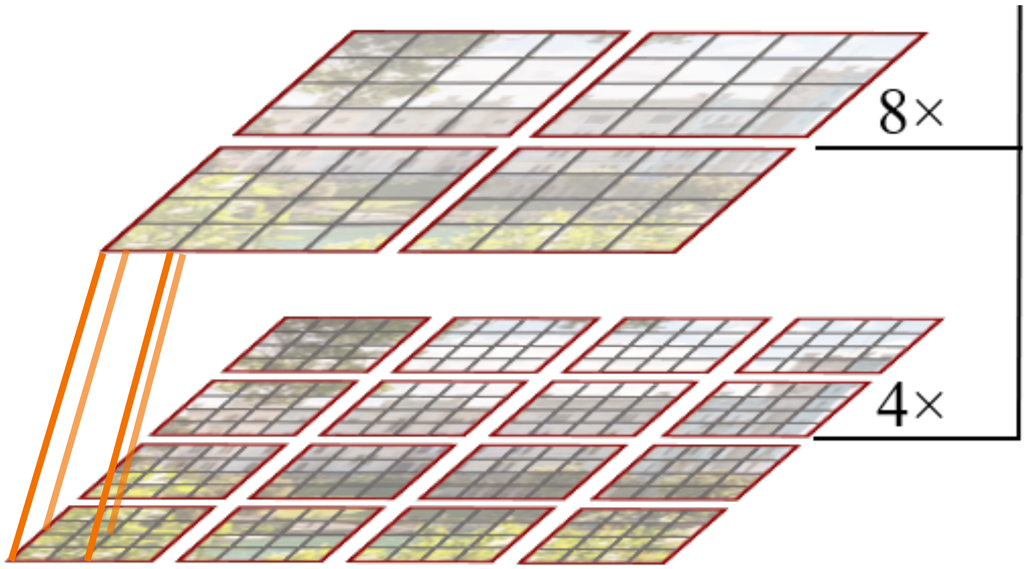


Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” (2021)

Swin Transformer

Successively decrease the number of tokens using **patch merging**:

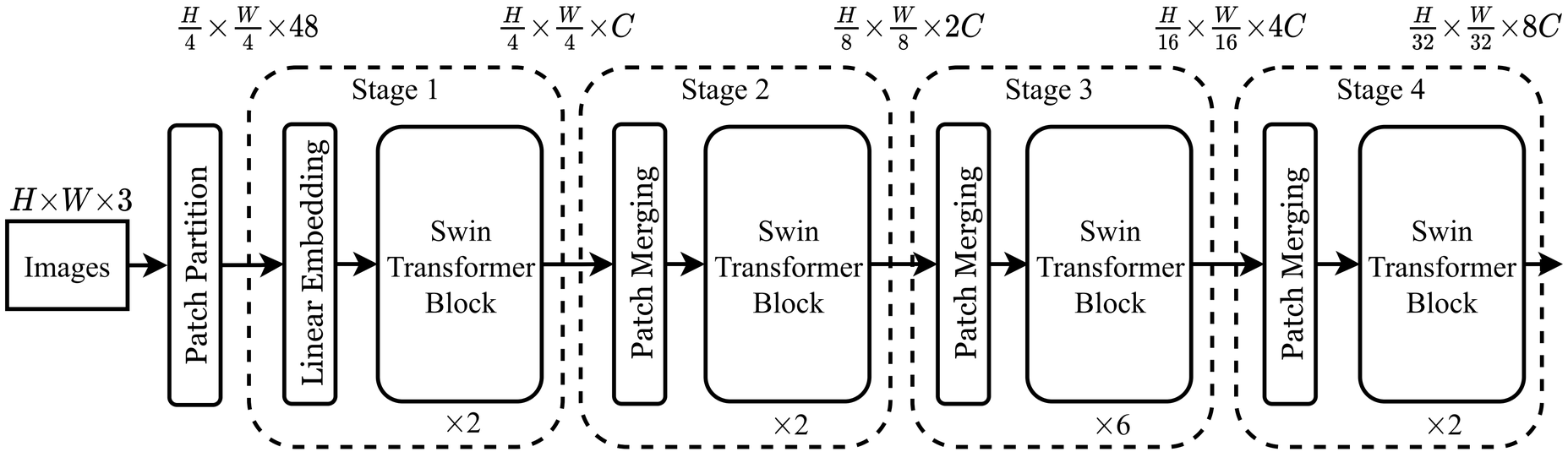
- concatenate 2x2 C-dim patches into a feature vector (4xC-dim);
- linearly transform to a 2xC dimensional vector.



Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” (2021)

Swin Transformer

Alternate Swin Transformer blocks with patch merging:



Note how the pattern of dimensionality reduction is similar to CNNs.

Liu et al., “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows” (2021)

Swin Transformer: Results

- More efficient and accurate than ViT and some of CNNs (despite using lower resolution input).
- Note: No pre-training on on large datasets!
- Improved scalability on large datasets (compare ImageNet-1K and ImageNet-22K)

ImageNet-22K

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
R-101x3 [38]	384^2	388M	204.6G	-	84.4
R-152x4 [38]	480^2	937M	840.5G	-	85.4
ViT-B/16 [20]	384^2	86M	55.4G	85.9	84.0
ViT-L/16 [20]	384^2	307M	190.7G	27.3	85.2
Swin-B	224^2	88M	15.4G	278.1	85.2
Swin-B	384^2	88M	47.0G	84.7	86.4
Swin-L	384^2	197M	103.9G	42.1	87.3

ImageNet-1K

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
RegNetY-4G [48]	224^2	21M	4.0G	1156.7	80.0
RegNetY-8G [48]	224^2	39M	8.0G	591.6	81.7
RegNetY-16G [48]	224^2	84M	16.0G	334.7	82.9
EffNet-B3 [58]	300^2	12M	1.8G	732.1	81.6
EffNet-B4 [58]	380^2	19M	4.2G	349.4	82.9
EffNet-B5 [58]	456^2	30M	9.9G	169.1	83.6
EffNet-B6 [58]	528^2	43M	19.0G	96.9	84.0
EffNet-B7 [58]	600^2	66M	37.0G	55.1	84.3
ViT-B/16 [20]	384^2	86M	55.4G	85.9	77.9
ViT-L/16 [20]	384^2	307M	190.7G	27.3	76.5
DeiT-S [63]	224^2	22M	4.6G	940.4	79.8
DeiT-B [63]	224^2	86M	17.5G	292.3	81.8
DeiT-B [63]	384^2	86M	55.4G	85.9	83.1
Swin-T	224^2	29M	4.5G	755.2	81.3
Swin-S	224^2	50M	8.7G	436.9	83.0
Swin-B	224^2	88M	15.4G	278.1	83.5
Swin-B	384^2	88M	47.0G	84.7	84.5

Swin Transformer: Summary

- Reconciles the inductive biases from CNNs with Transformers;
- State-of-the-art on image classification, object detection and semantic segmentation;
- Linear computational complexity
  - local attention windows of fixed size (independent from the image resolution).

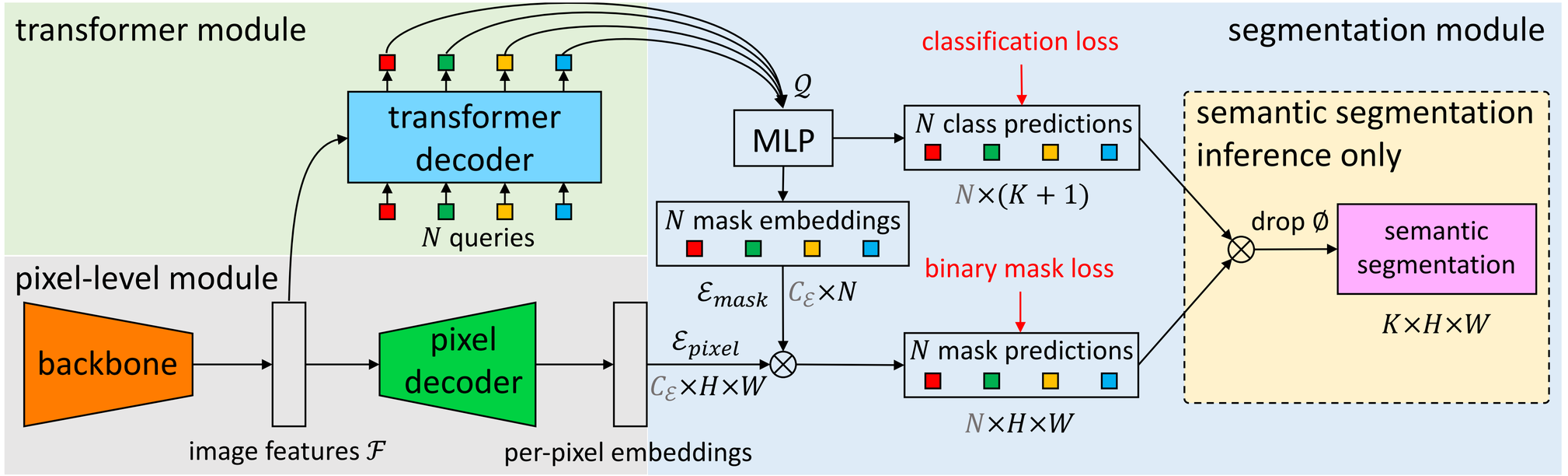
Swin Transformer: Summary

- Reconciles the inductive biases from CNNs with Transformers;
- State-of-the-art on image classification, object detection and semantic segmentation;
- Linear computational complexity
  - local attention windows of fixed size (independent from the image resolution).

- Demonstrates that Transformers are strong vision models across a range of classic downstream applications.
- Best paper award at ICCV ’21.

MaskFormer

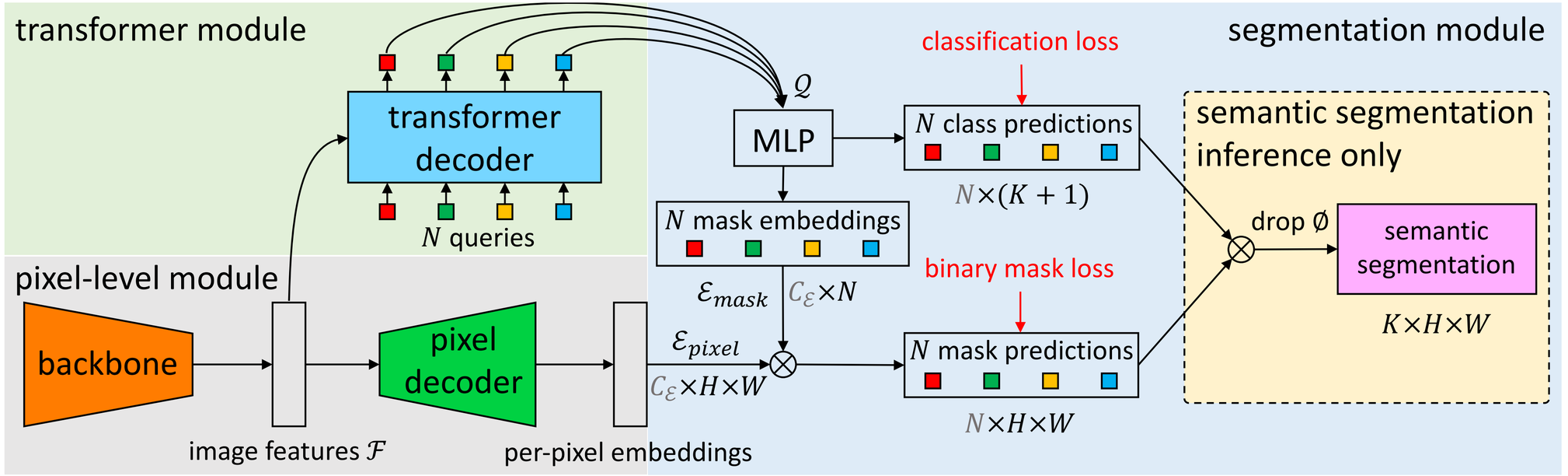
MaskFormer: a unified model for semantic and panoptic segmentation



[Cheng et al., 2021]

MaskFormer

MaskFormer: a unified model for semantic and panoptic segmentation

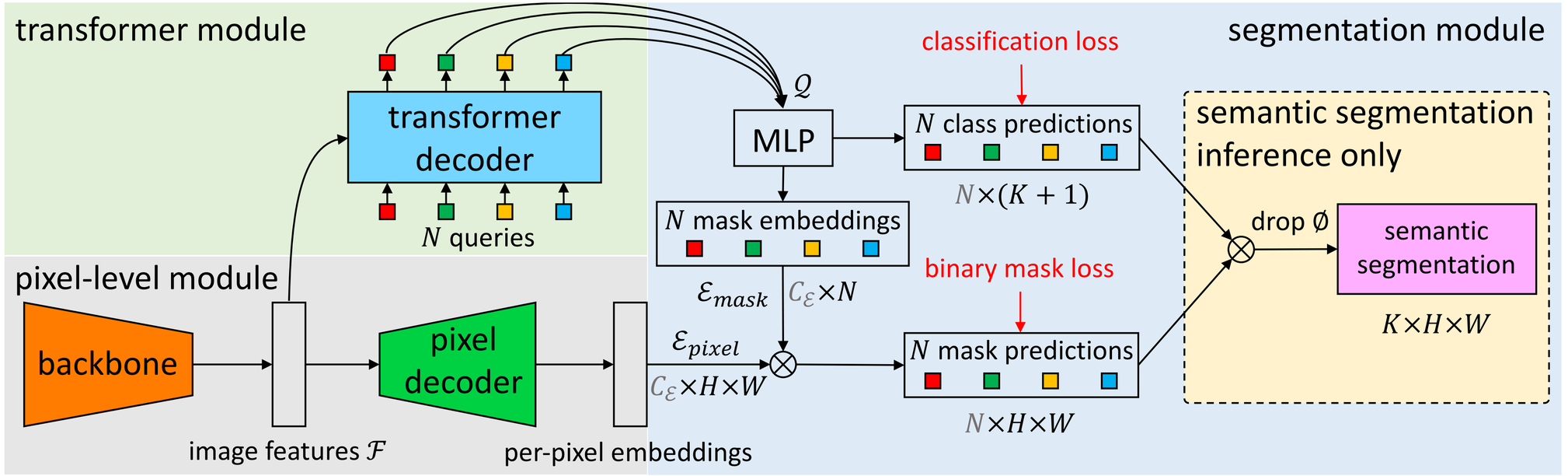


Observe similarity to SOLO

[Cheng et al., 2021]

MaskFormer

MaskFormer: a unified model for semantic and panoptic segmentation

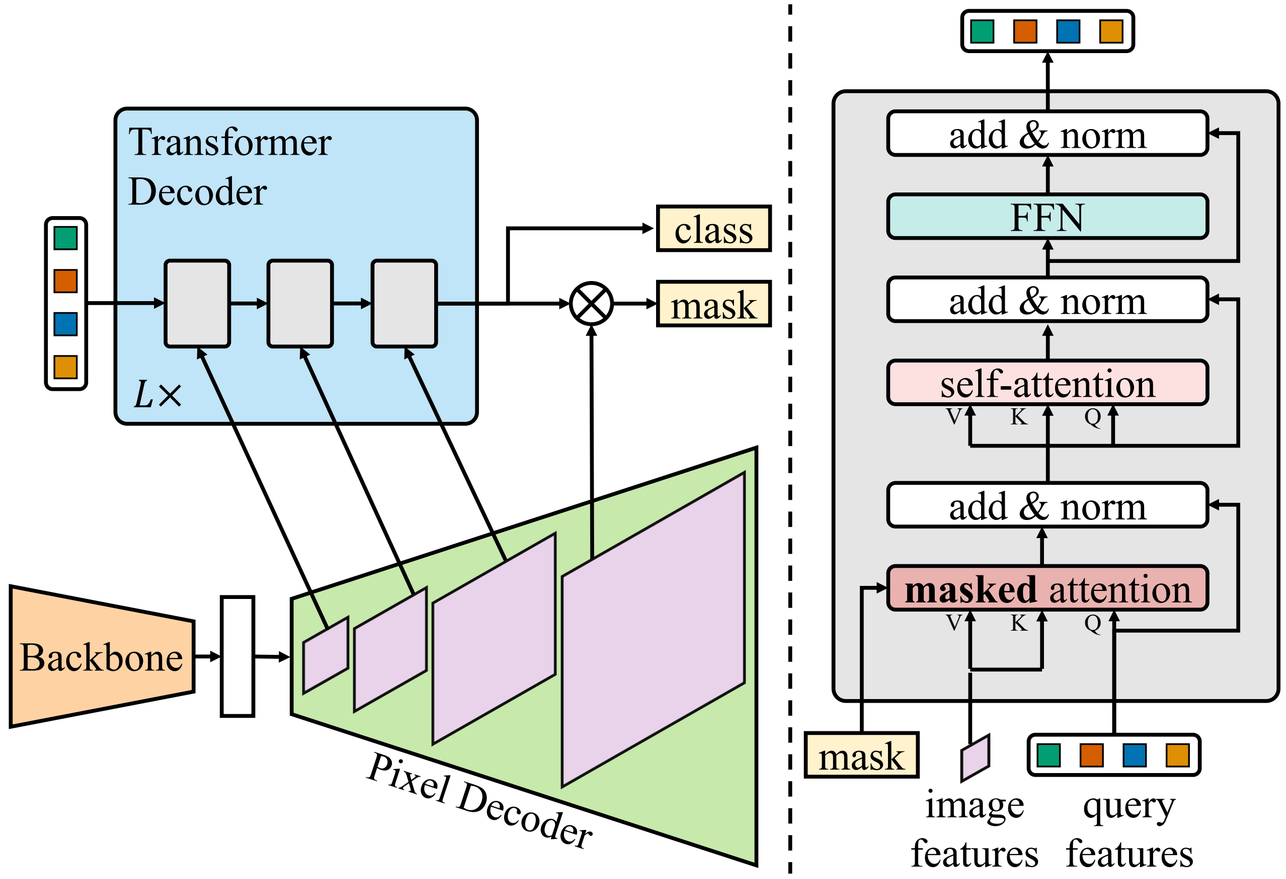


Works well, but has troubles with small objects/segments.

[Cheng et al., 2021]

Mask2Former

Idea: Attend with self-attention at multiple scales of the feature hierarchy:



[Cheng et al., 2021]

Masked attention

Idea: constrain attention only to the foreground area corresponding to the query.

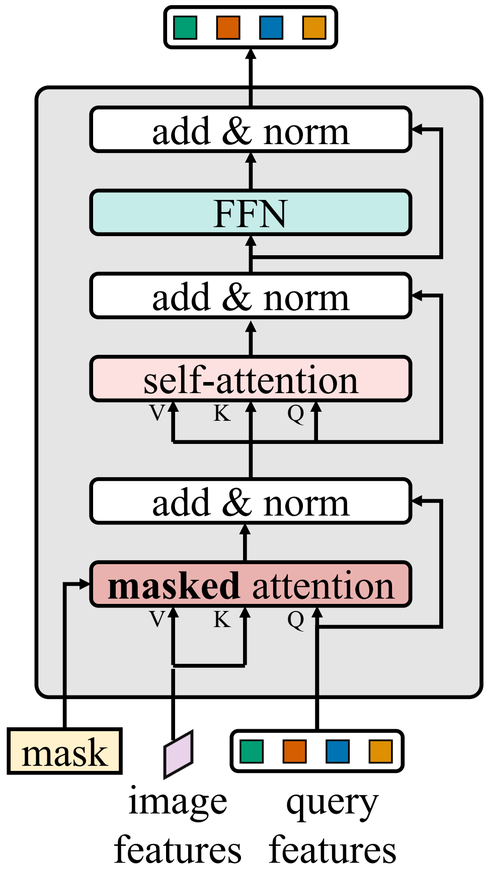
Standard self-attention: $\mathbf{X}_l = \text{softmax}(\mathbf{Q}_l \mathbf{K}_l^{\mathrm{T}})\mathbf{V}_l + \mathbf{X}_{l-1}$

Masked attention: $\mathbf{X}_l = \text{softmax}(\boldsymbol{\mathcal{M}}_{l-1} + \mathbf{Q}_l \mathbf{K}_l^{\mathrm{T}})\mathbf{V}_l + \mathbf{X}_{l-1}$

$$\boldsymbol{\mathcal{M}}_{l-1}(x,y) = \begin{cases} 0 & \text{if } \mathbf{M}_{l-1}(x,y) = 1 \\ -\infty & \text{otherwise} \end{cases}$$

mask binarisation

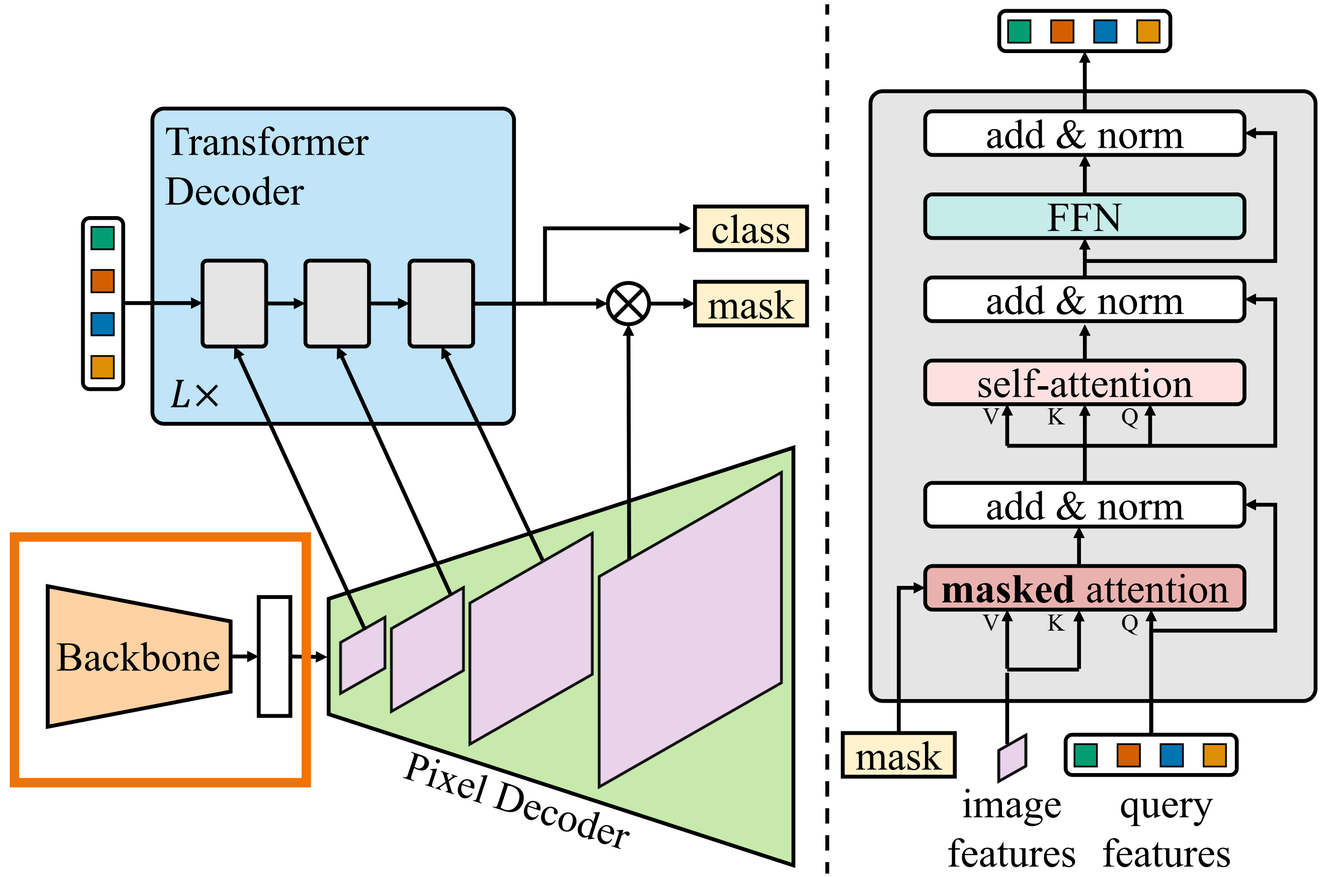
Where does this come from?



[Cheng et al., 2021]

Mask2Former

Idea: Attend with self-attention at multiple scales of the feature hierarchy:



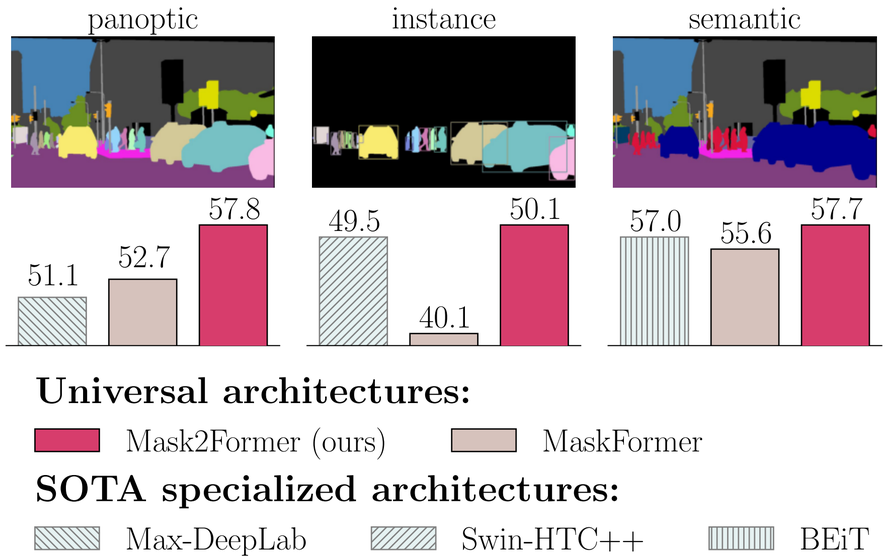
The same as MaskFormer, so we have the initial masks corresponding to each query

[Cheng et al., 2021]

Mask2Former: Summary

A unified model:

- note that we do not talk about “stuff” and “things” anymore
- queries abstract those notions away.
- Achieves state-of-the-art accuracy across segmentation tasks:



Conclusions

- Transformers have revolutionised the field of NLP, achieving incredible results.
- We observe massive impact on computer vision (DETR, ViT).
- Complementing CNNs, Transformers have reached state-of-the-art in object classification, detection, tracking and image generation.

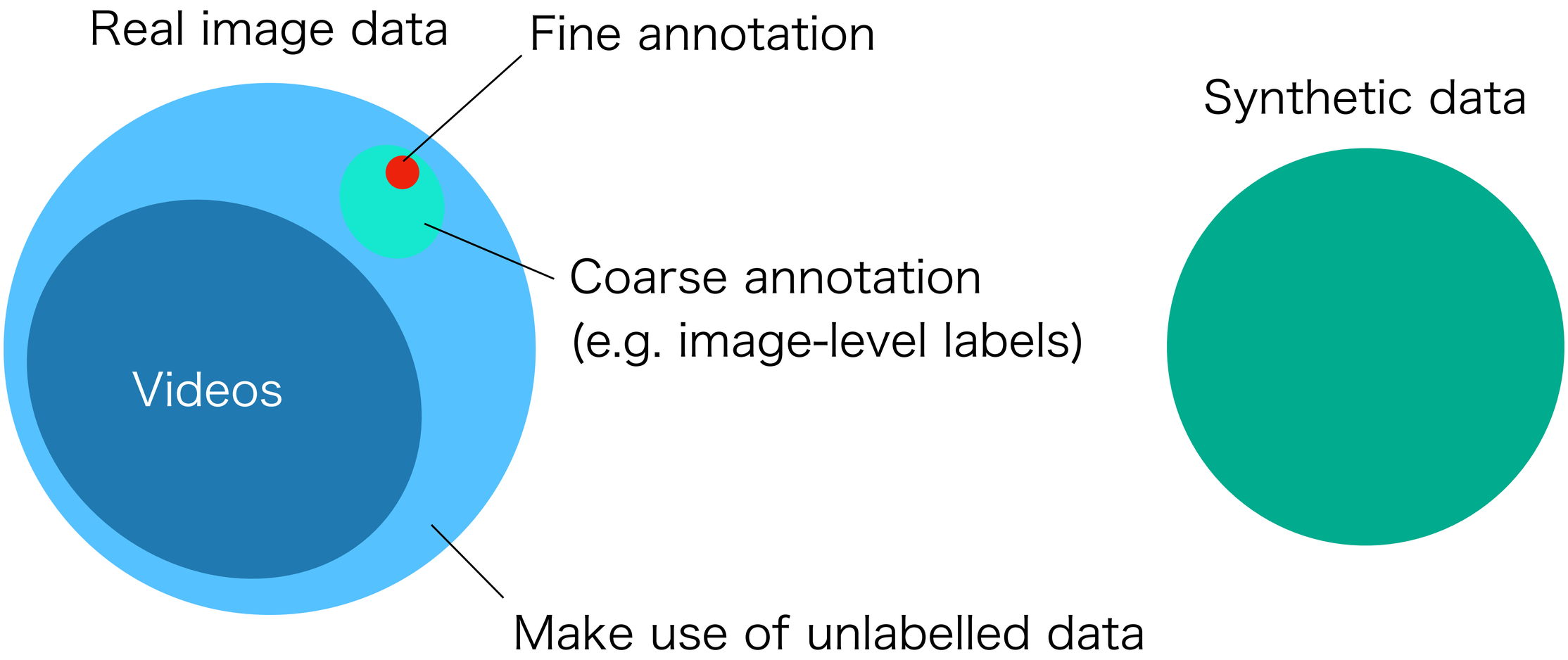
Conclusions

- Transformers have revolutionised the field of NLP, achieving incredible results.
- We observe massive impact on computer vision (DETR, ViT).
- Complementing CNNs, Transformers have reached state-of-the-art in object classification, detection, tracking and image generation.

- A grain of salt: This often comes at an increased computational budget (larger GPUs, longer training).

Learning without labels

Limited supervision



Semi-supervised paradigms

- Entropy minimisation:
  - also known as “self-training” or “pseudo-labelling”.
- Consistency regularisation:
  - enforce desired invariance or equivariance constraints.
- Domain alignment:
  - shared feature space between the labelled and unlabelled examples.
  - can be explicit (GAN) or implicit (proxy tasks).

Semi-supervised loss

- It is all about the loss on the unsupervised samples:

$$\mathcal{L}(\{(x_i, y_i)\}_i, \{\hat{x}_i\}_i) = \sum_i \mathcal{L}_{\text{supervised}}(x_i, y_i) + \lambda \sum_i \mathcal{L}_{\text{unsupervised}}(\hat{x}_i)$$

... and the training procedure:

- joint training (domain alignment);
- pre-train on the labelled set, then finetune on the unlabelled set (or jointly).

Semi-supervised loss

- Example:
  - Entropy minimisation for semantic segmentation (“self-training”).

$$\mathcal{L}(\{(x_i, y_i)\}_i, \{\hat{x}_i\}_i) = \sum_i \mathcal{L}_{\text{supervised}}(x_i, y_i) + \lambda \sum_i \mathcal{L}_{\text{unsupervised}}(\hat{x}_i)$$

- Objective: minimise the entropy of class distribution for each pixel:

$$\mathcal{L}_{\text{unsupervised}}(\hat{x}_i) := \mathbb{E}[-\log p(x_i)] \approx -\sum_i p(x_i) \log p(x_i)$$

Grandvalet and Bengio, “Semi-supervised Learning by Entropy Minimization” (2004).

Semi-supervised loss

- Example:
  - Entropy minimisation for semantic segmentation (“self-training”).

$$\mathcal{L}(\{(x_i, y_i)\}_i, \{\hat{x}_i\}_i) = \sum_i \mathcal{L}_{\text{supervised}}(x_i, y_i) + \lambda \sum_i \mathcal{L}_{\text{unsupervised}}(\hat{x}_i)$$

- Objective: minimise the entropy of class distribution for each pixel:

$$\mathcal{L}_{\text{unsupervised}}(\hat{x}_i) := \mathbb{E}[-\log p(x_i)] \approx -\sum_i p(x_i) \log p(x_i)$$

QUIZ: Intuition?

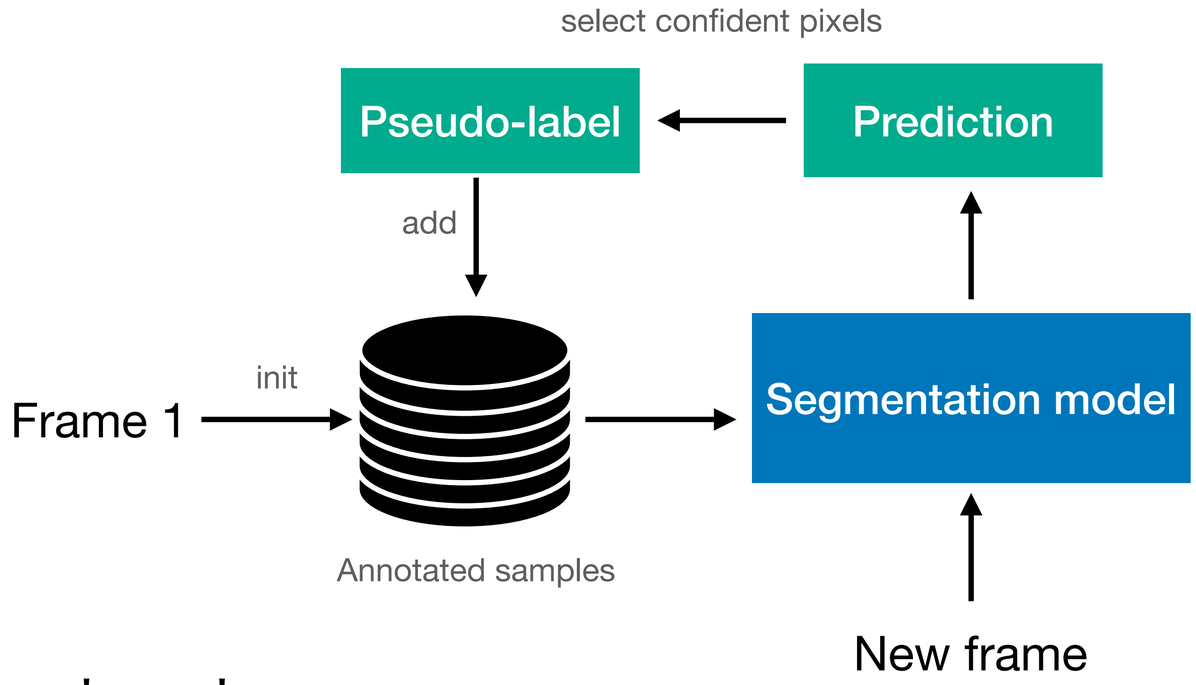
Grandvalet and Bengio, “Semi-supervised Learning by Entropy Minimization” (2004).

Self-training with pseudo labels

1. Train a strong baseline on the labelled set:
   - e.g. with heavy data augmentation (crops, photometric noise).
2. Predict “pseudo-labels” for the unlabelled set.
3. Select a subset of the labelled set.
4. Continue training the network on the joint set (labelled and pseudo-labelled samples).
5. Repeat steps 2-4.

OnAVOS: Online Adaptation

- Online adaptation: adapt model to appearance changes in every frame, not just the first frame.



- Drawback: can be slow.

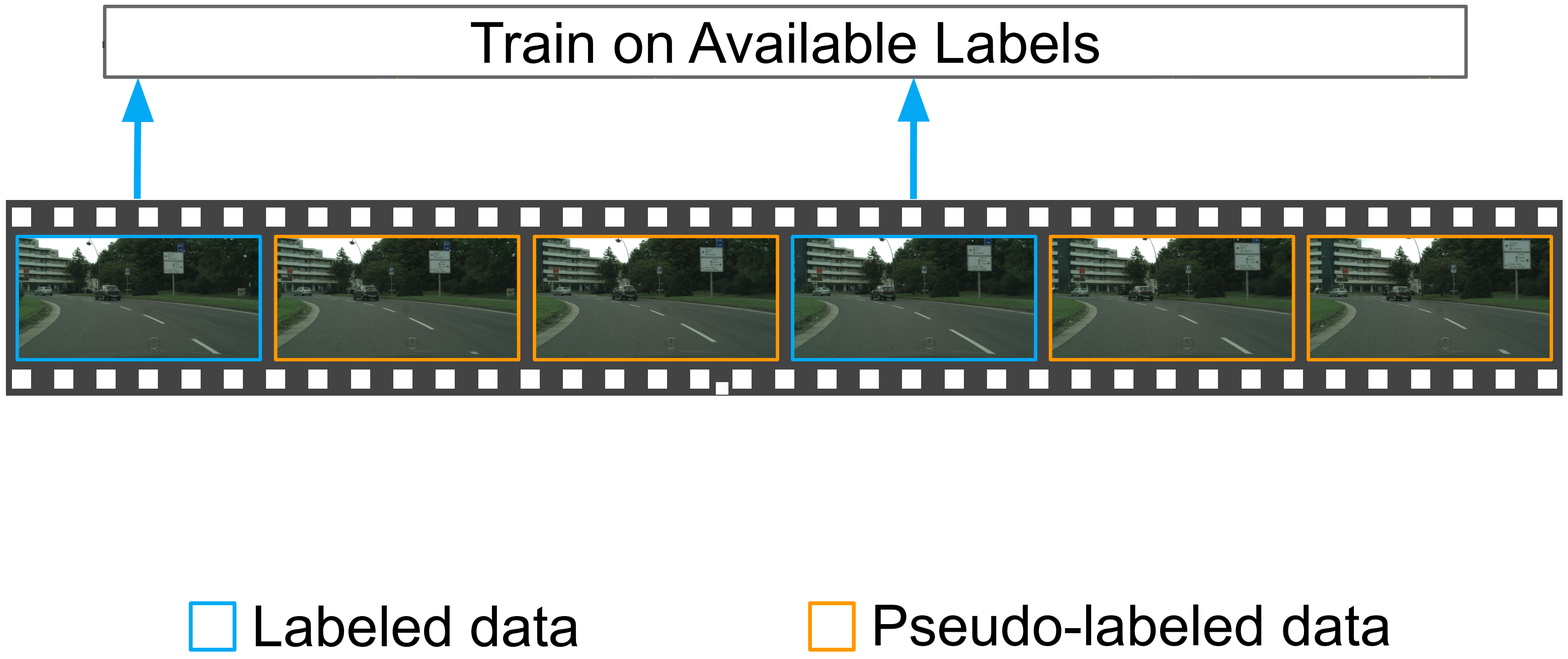
Self-training with pseudo labels

Open questions:

- How to select the labels (the confidence threshold)?
  - high vs low threshold trade-off (QUIZ)
  - high: no learning signal (the gradient will be close to zero);
  - low: noisy labels → low accuracy.
- Tedious to train (multiple training rounds).

Self-training: Naive student

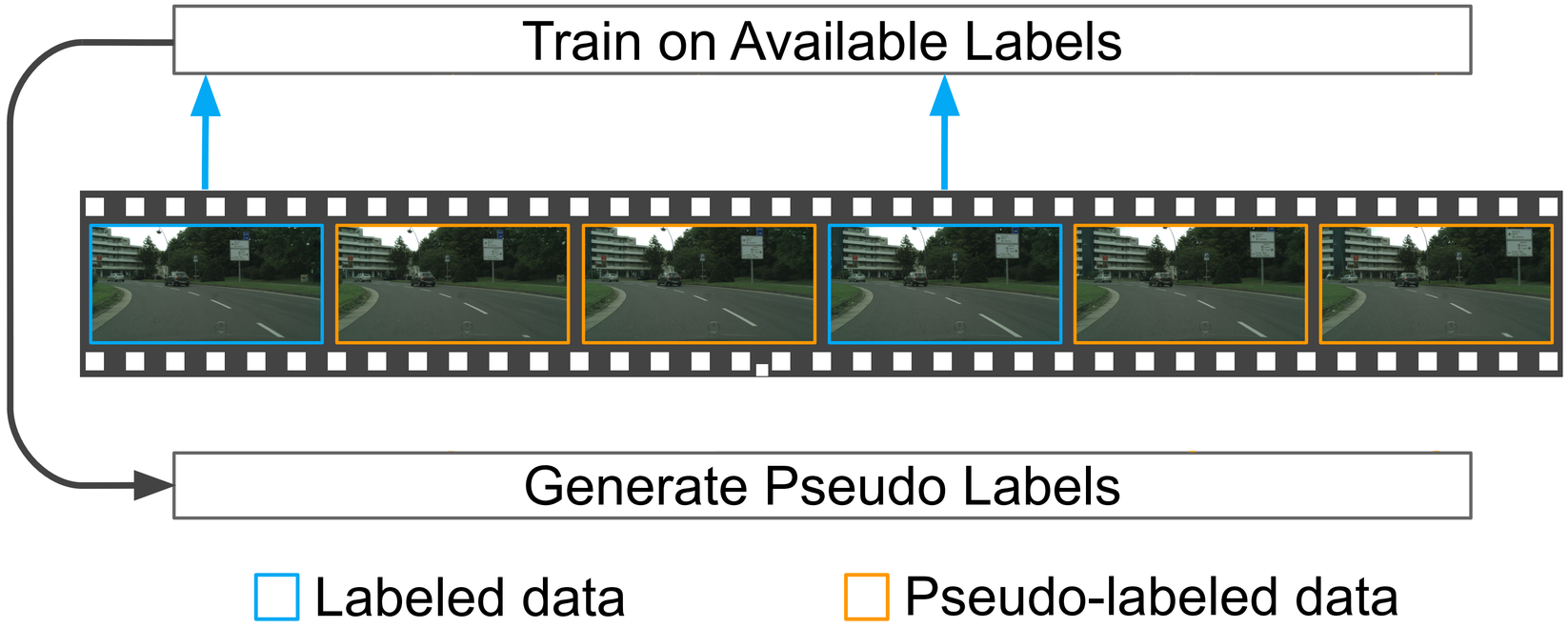
1. Train a teacher network on the labelled images:



Chen et al., Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation (2020)

Self-training: Naive student

2. Generate pseudo-labels

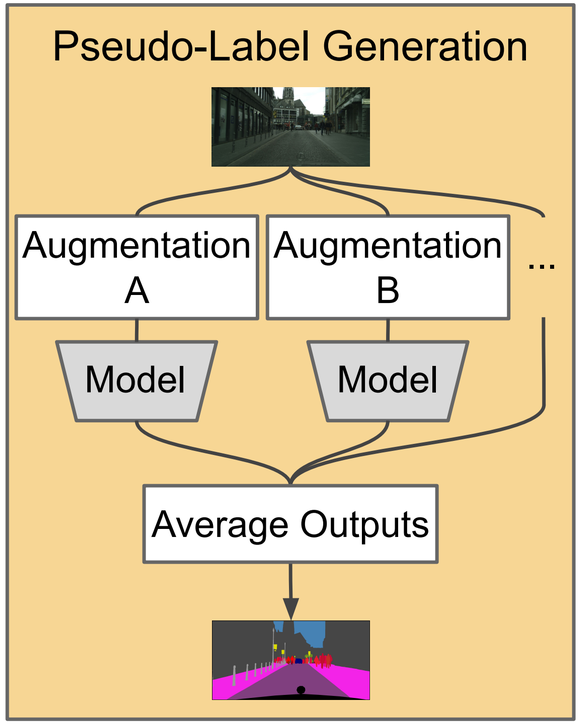


Chen et al., Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation (2020)

Generating pseudo labels

2. Generate pseudo-labels:

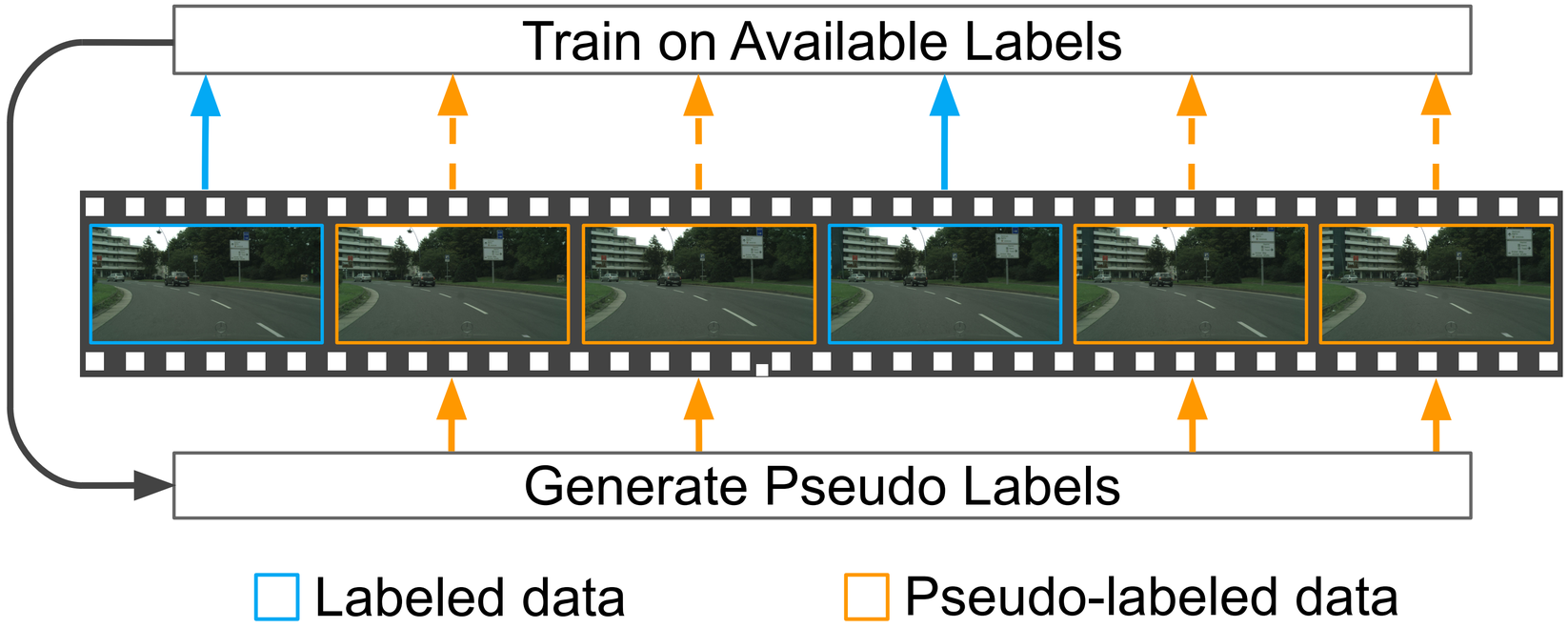
- Test-time augmentation: average the prediction across image augmentations (etc. multi-scale, flips).
- This improves prediction accuracy.



Chen et al., Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation (2020)

Self-training: Naive student

3. Train a student network on the pseudo-labels:

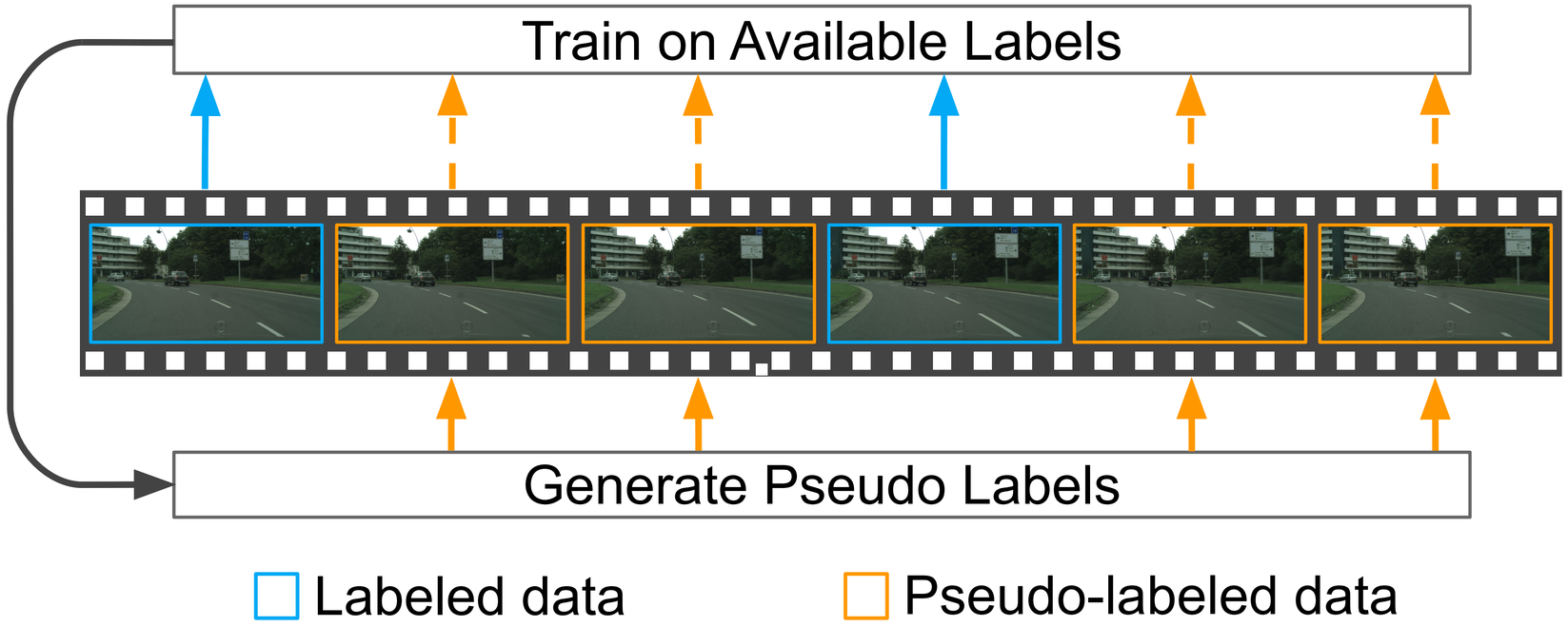


Chen et al., Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation (2020)

Self-training: Naive student

4. Fine-tune the student on the labelled images;

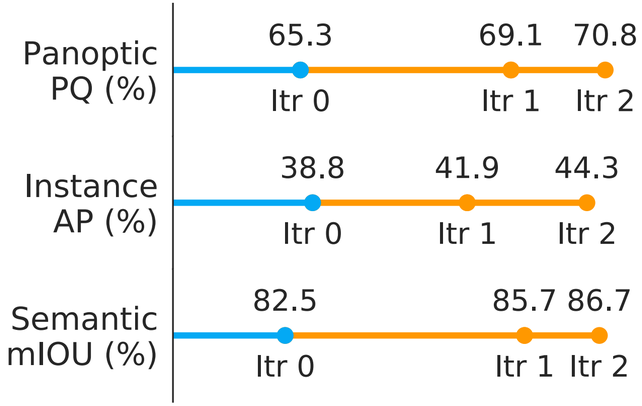
5. The student becomes a teacher → repeat the cycle.



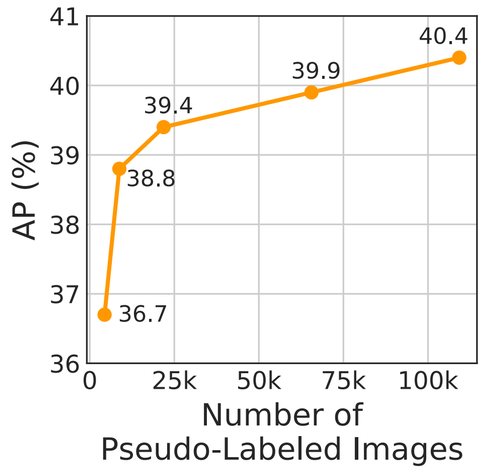
Chen et al., Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation (2020)

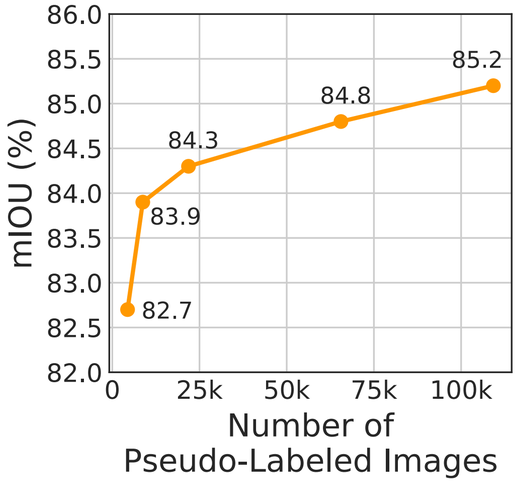
Self-training: Naive student

Improved accuracy over iterations:



A positive trend w.r.t. number of pseudo-labelled samples:





Eventually seems to saturate (open problem)

Chen et al., Naive-Student: Leveraging Semi-Supervised Learning in Video Sequences for Urban Scene Segmentation (2020)

Self-training with pseudo labels

Issues:

- How to select the pseudo-labels (the confidence threshold)?
  - high vs low threshold trade-off (QUIZ)
  - high: no learning signal (the gradient will be close to zero);
  - low: noisy labels → low accuracy.
- Tedious to train (multiple training rounds).
- Sensitive to the initial model:
  - fails if the initial predictions are largely inaccurate.

Self-training with pseudo labels

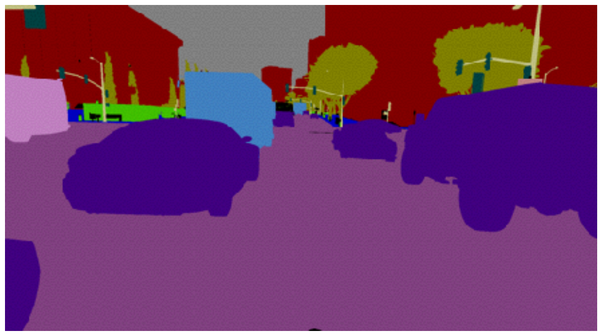
Issues:

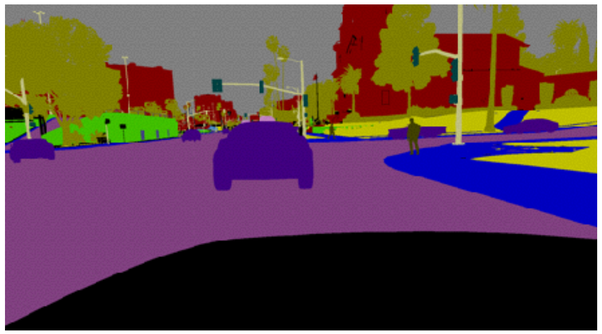
- How to select the pseudo-labels (the confidence threshold)?
  - high vs low threshold trade-off (QUIZ)
  - high: no learning signal (the gradient will be close to zero);
  - low: noisy labels → low accuracy.
- Tedious to train (multiple training rounds).
- Sensitive to the initial model:
  - fails if the initial predictions are largely inaccurate. → How can we build stronger initialisation?

Learning from synthetic data

How about using synthetic data?

- Labels are easier (hence cheaper) to obtain at a large scale.





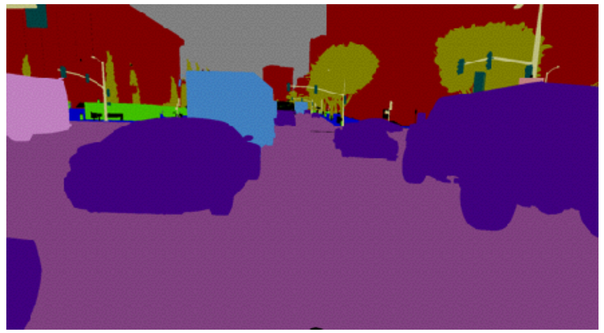
Richter et al., “Playing for Data: Ground Truth from Computer Games” (2016).

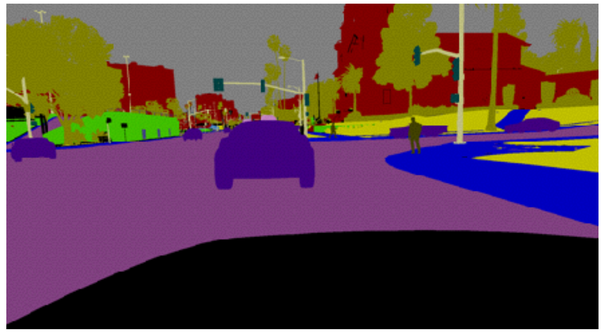

CRUCIAL FIX

Learning from synthetic data

How about using synthetic data?

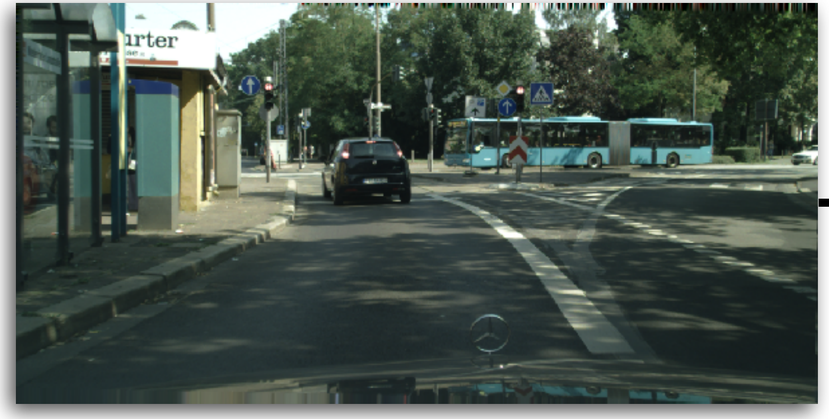
- Labels are easier (hence cheaper) to obtain at a large scale.
- Issue: poor generalisation.





Richter et al., “Playing for Data: Ground Truth from Computer Games” (2016).

Unsupervised Domain Adaptation



Unlabelled real-world data

Model

train-time only

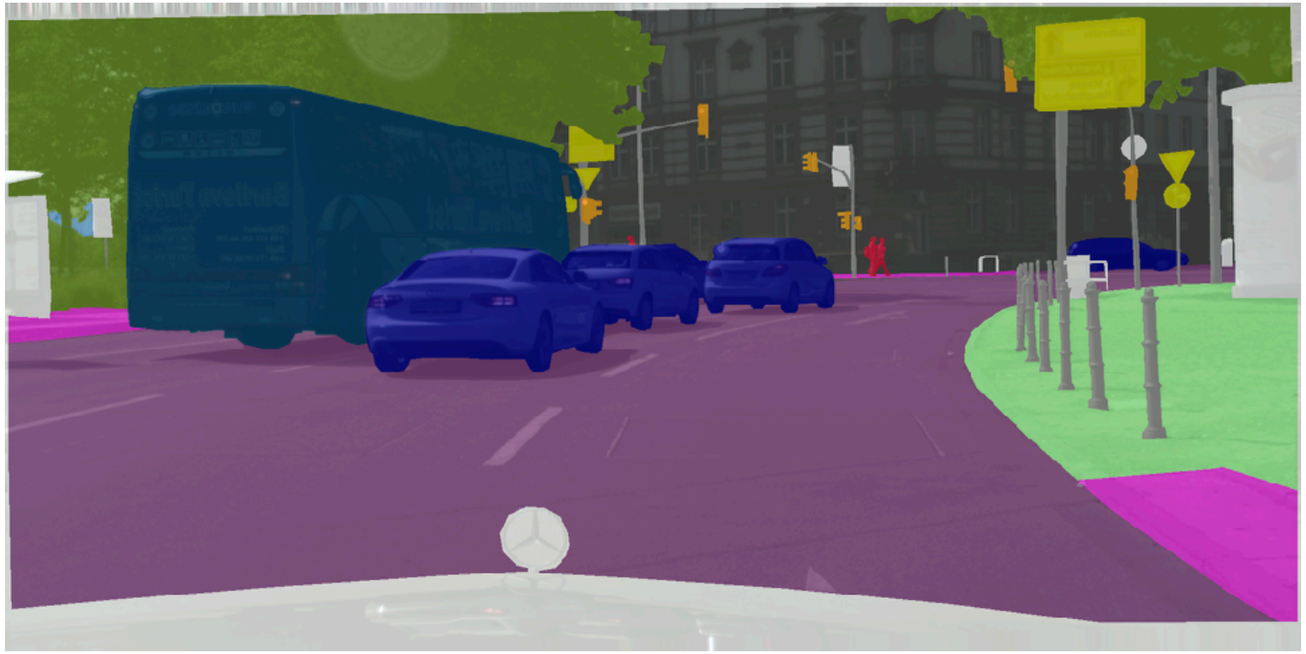
Semantic maps

Labelled synthetic data

Araslanov & Roth, 2021

Consistency regularisation

Consistent prediction across image transformations:



Araslanov & Roth, 2021

Consistency regularisation

Consistent prediction across image transformations:

Flipping

Araslanov & Roth, 2021

Consistency regularisation

Consistent prediction across image transformations:

Scaling

Araslanov & Roth, 2021

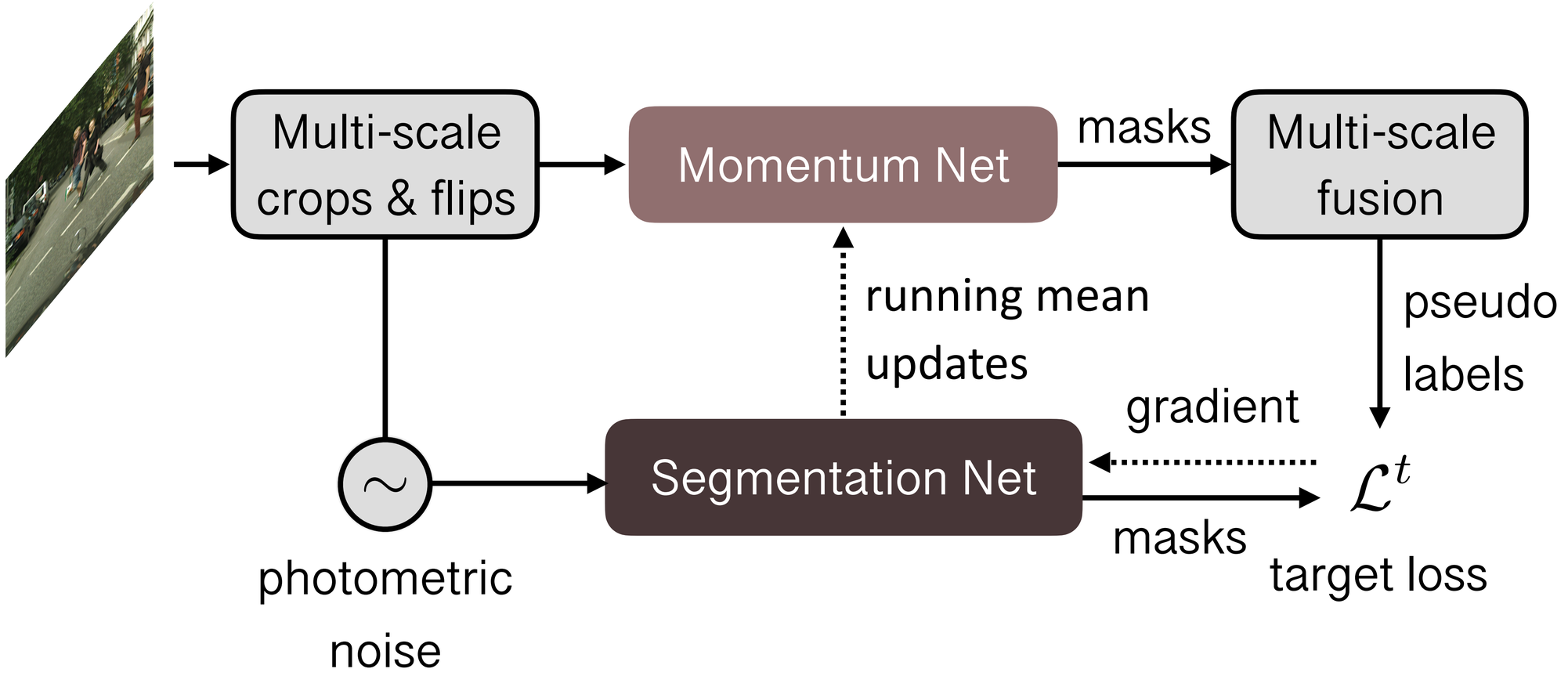
Consistency regularisation

Consistent prediction across image transformations:

Semantic meaning does not change.

Araslanov & Roth, 2021

Framework

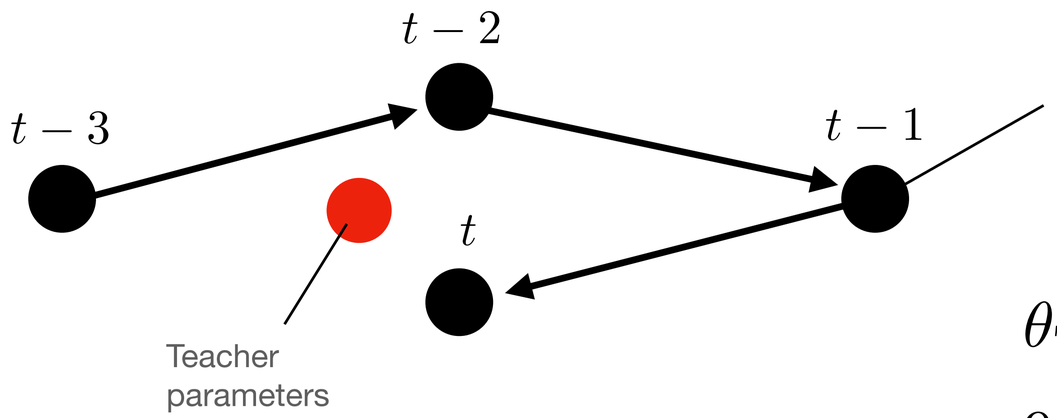


Araslanov & Roth, 2021

Momentum net

The momentum net “tracks” the parameters of the segmentation net:

- Exponential moving average:



$$\theta_{\mathrm{T}} = m\theta_{\mathrm{T}} + (1-m)\theta_{\mathrm{S}}$$

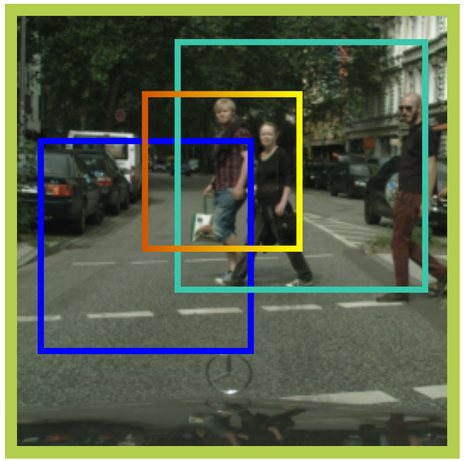
$$\text{e.g. } m = 0.9$$

- Also known as the “mean” teacher;
- Empirically more accurate than the model trained with SGD.

Momentum net

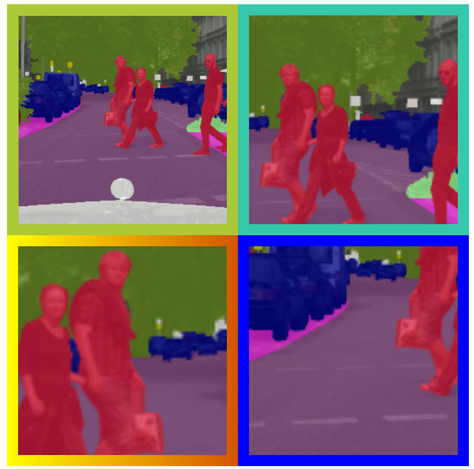
Test-time augmentation is applied online at training time:

Original

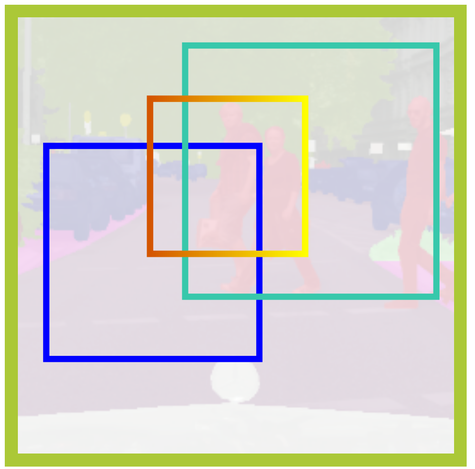


Input Batch

Predictions



Averaging



1. Random crops & horizontal flipping

2. Re-project and average predictions

3. Apply adaptive threshold

Araslanov & Roth, 2021


Before adaptation (Baseline)
After adaptation (Ours)
Araslanov & Roth, 2021

Momentum net

Consistency regularisation:

- a single pseudo-label for multiple image perturbations.

Advantages:

- simple to implement and very powerful;
  - DNNs are not scale-invariant, so this is a useful training signal.
- smoothly changing predictions even in a video;

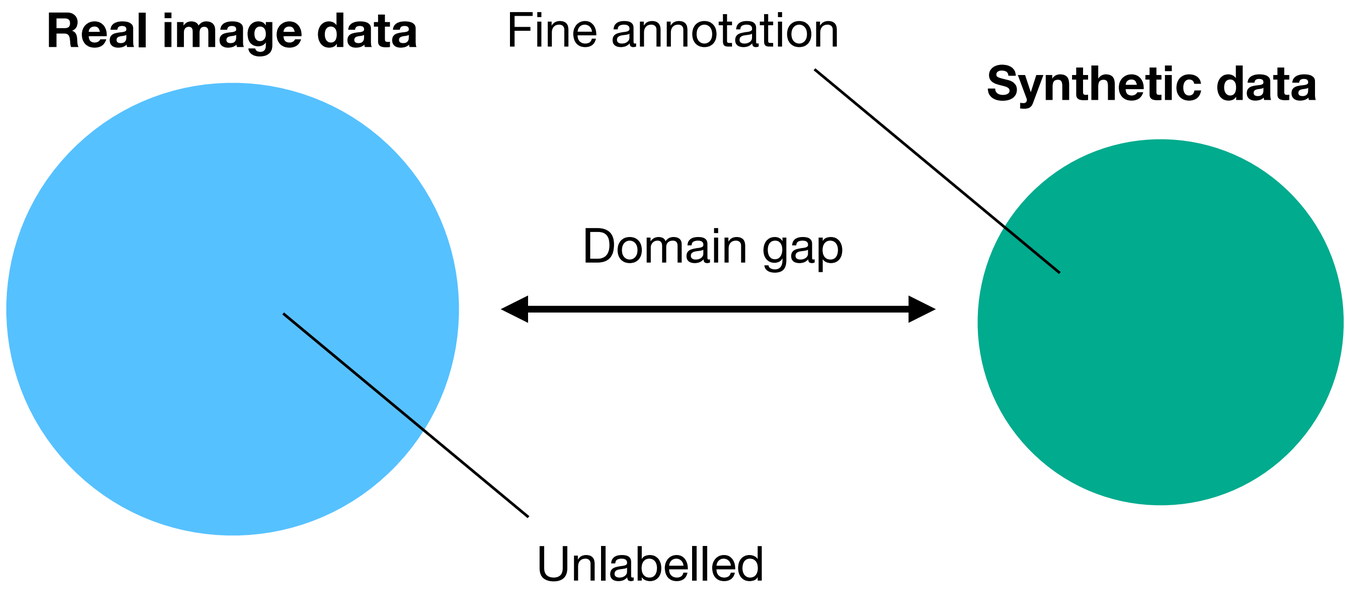
Disadvantages:

- needs careful hyperparameter tuning (e.g. the adaptive threshold).

Domain alignment

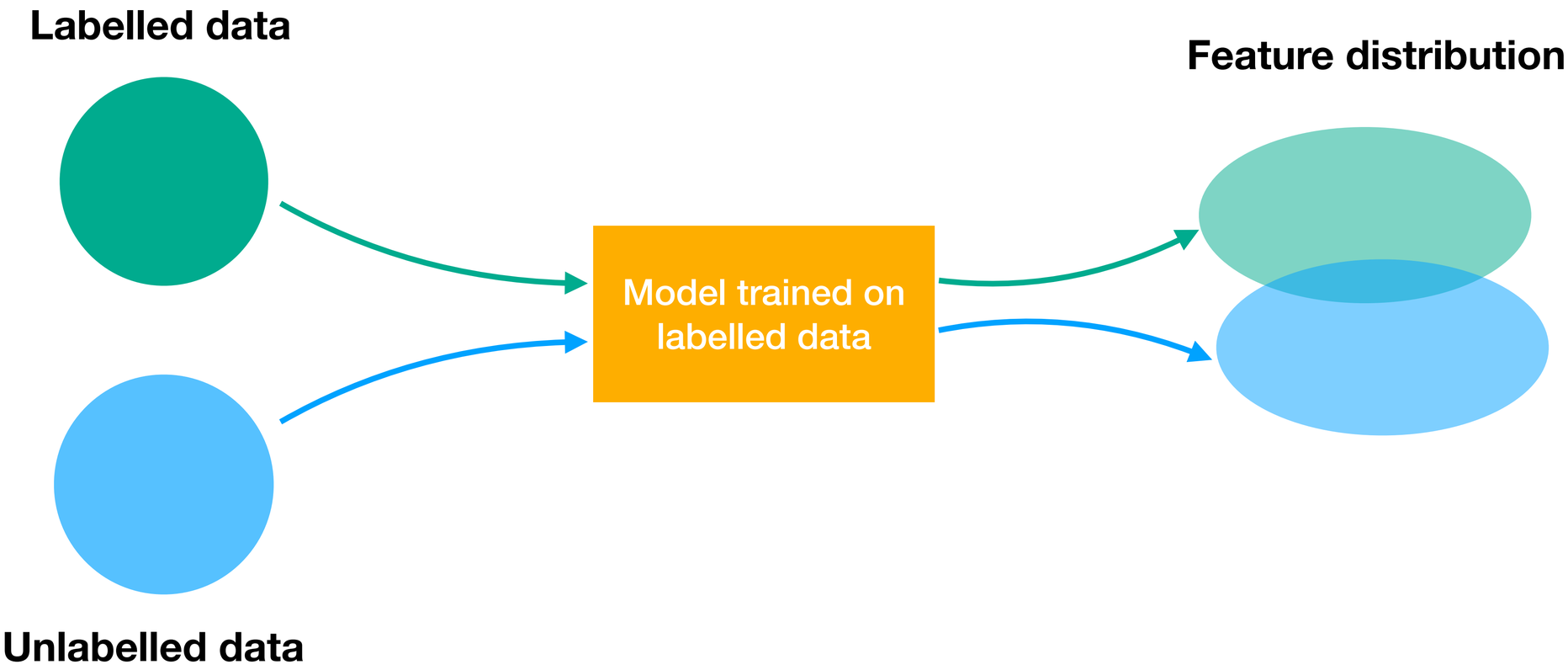
Labelled and unlabelled data may come from different distribution

- e.g. due to differences in the synthetic and real appearance.



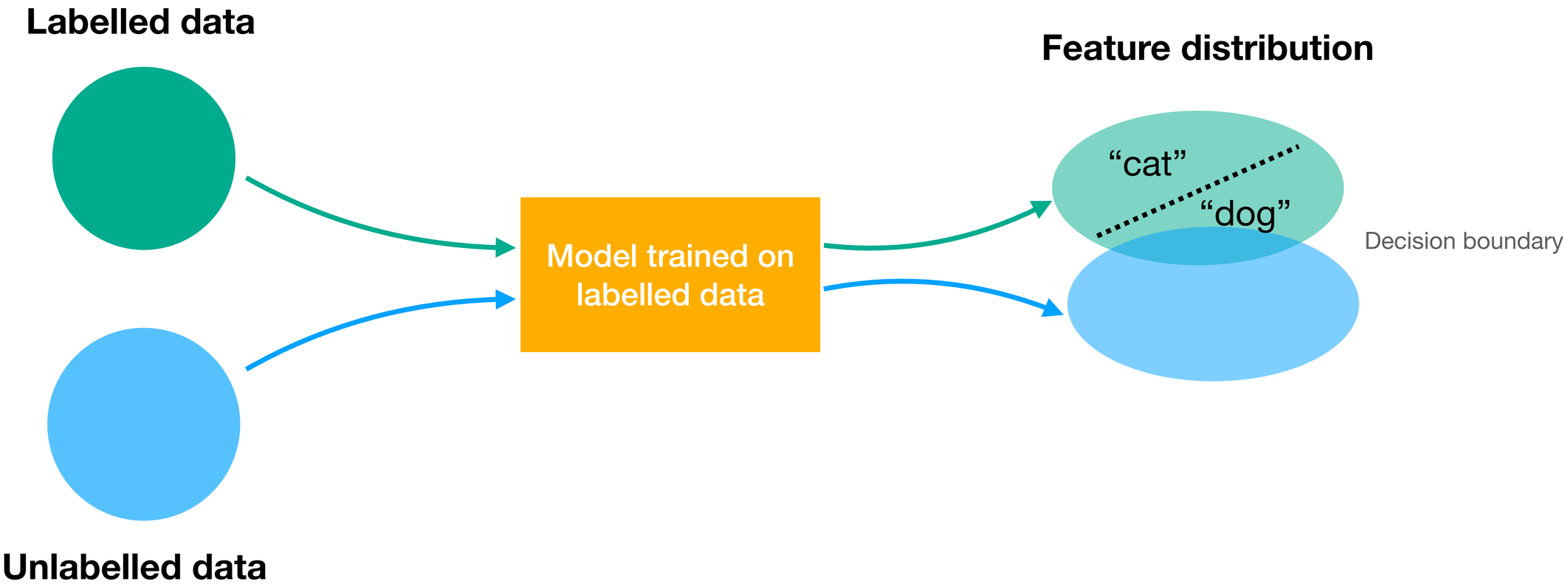
Domain alignment

This translates into disjoint feature distribution of a model trained only on the labelled data:



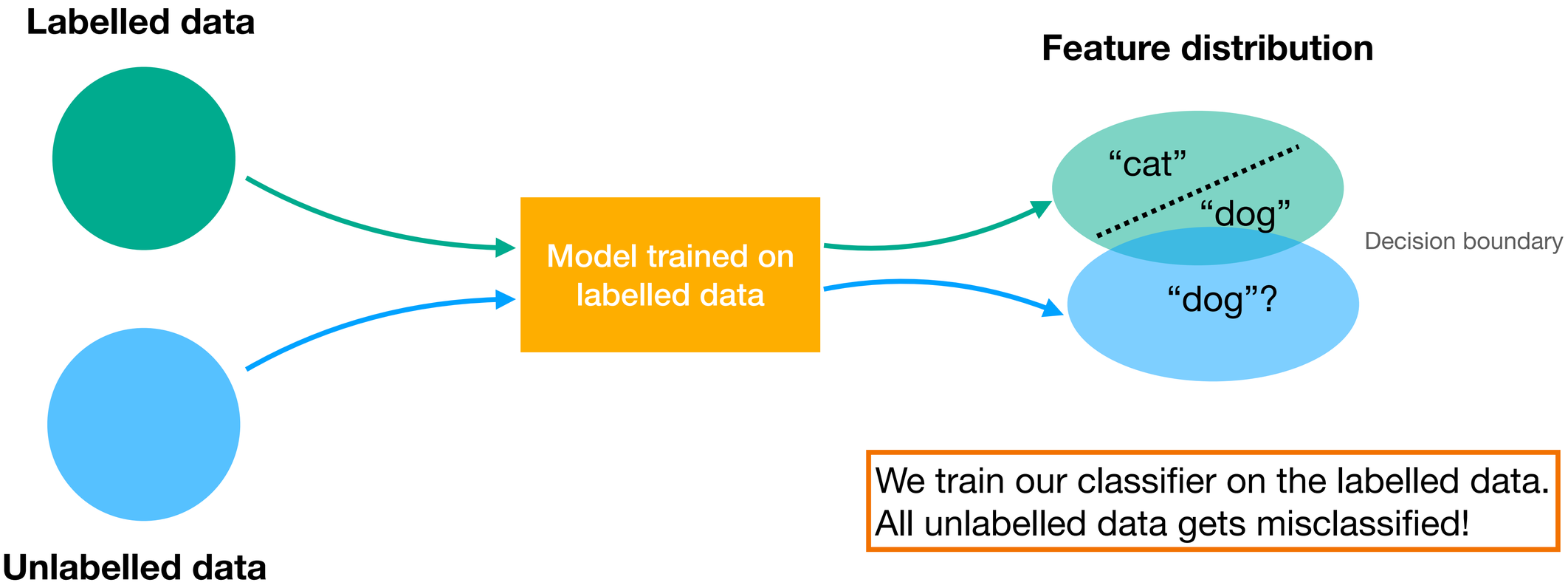
Domain alignment

This translates into disjoint feature distribution of a model trained only on the labelled data:



Domain alignment

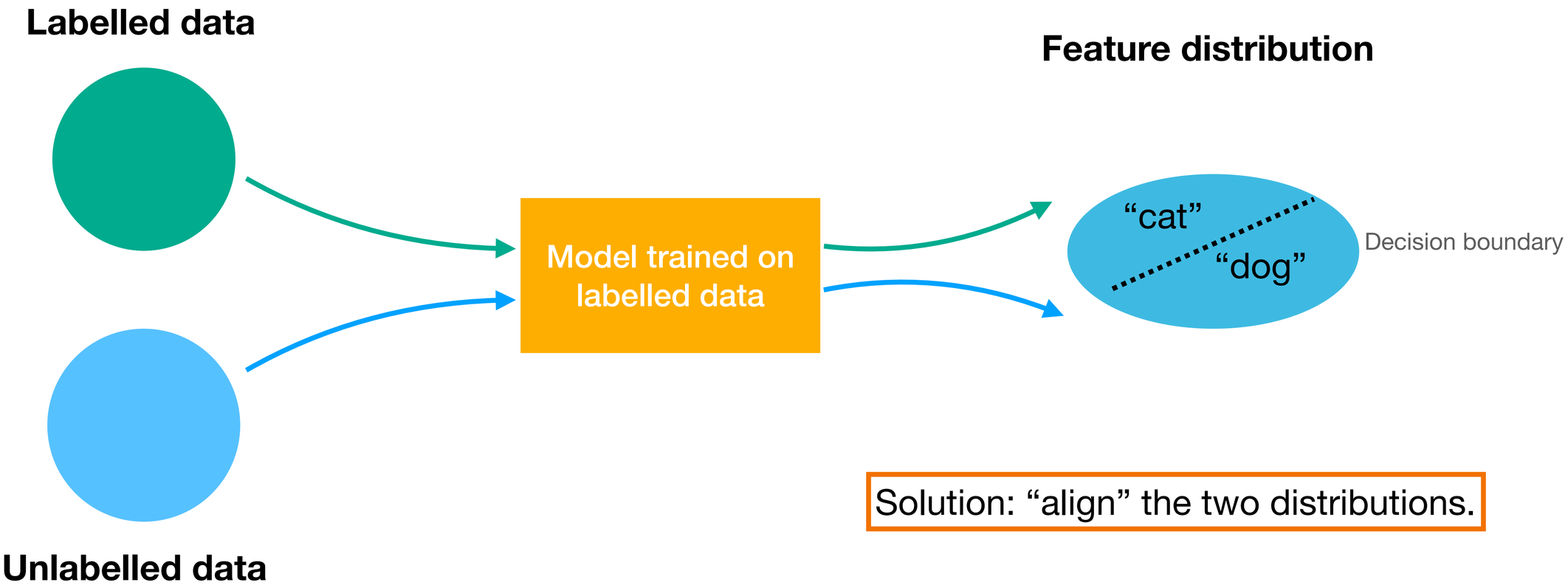
This translates into disjoint feature distribution of a model trained only on the labelled data:



We train our classifier on the labelled data.
All unlabelled data gets misclassified!

Domain alignment

This translates into disjoint feature distribution of a model trained only on the labelled data:

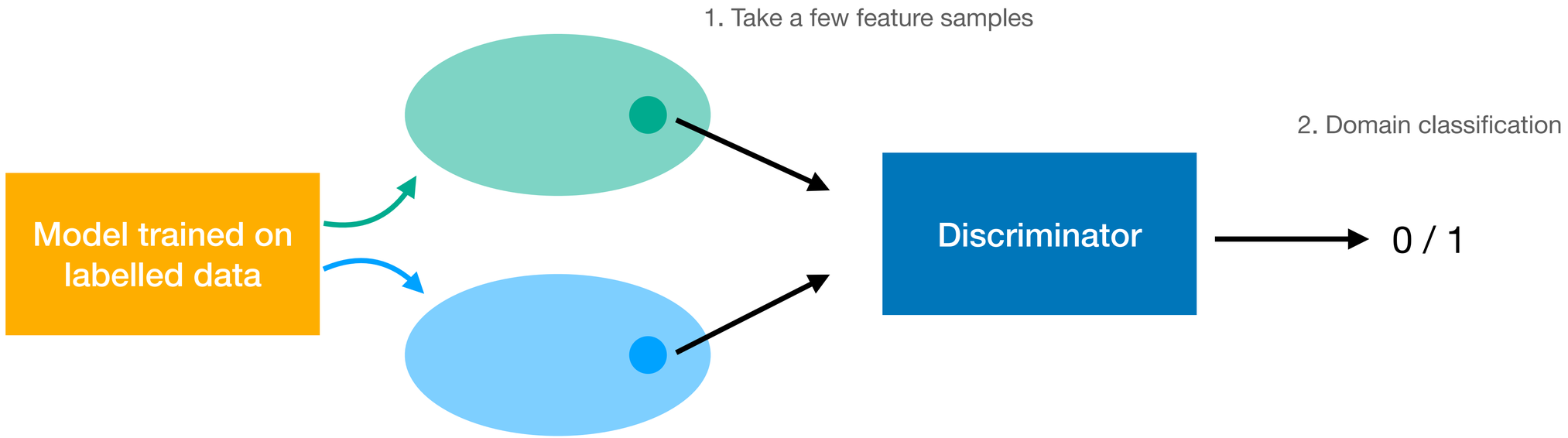


Solution: “align” the two distributions.

Domain alignment

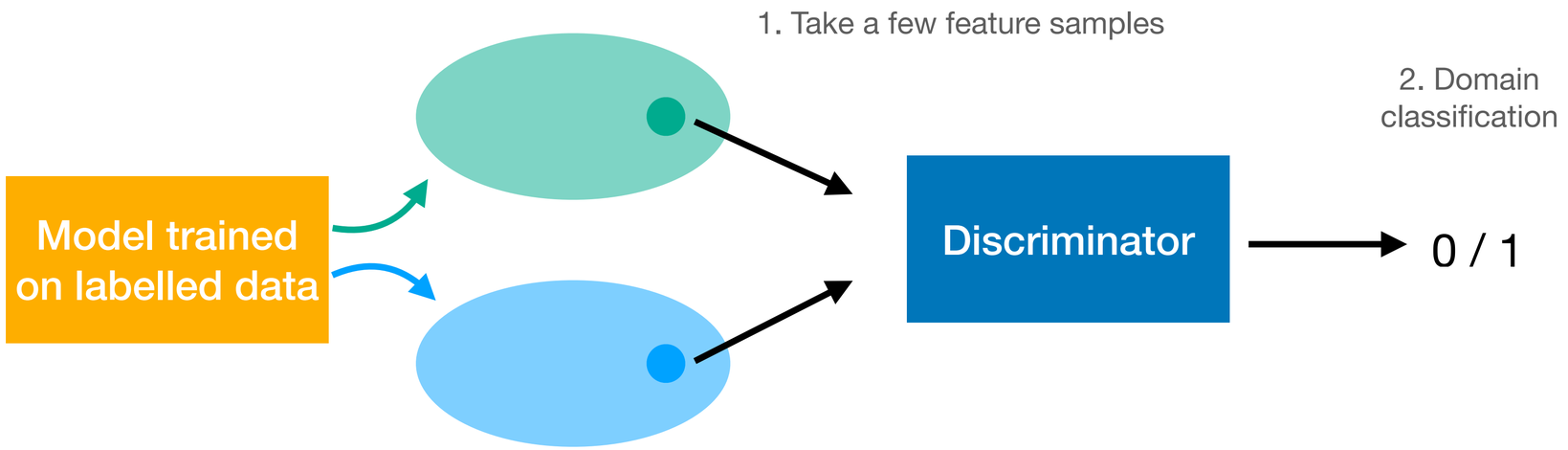
Domain alignment: make two feature distributions indistinguishable

- We could use a GAN:



Domain alignment

- Discriminator learns to classify the origin of the provided feature.
- The model learns
  - to classify the labelled images;
  - a feature representation that reduced discriminator accuracy.



Domain alignment

Additional reading:

- Ganin et al., “Domain-adversarial training of neural networks”. In JMLR, 2016.
- We can apply the same idea in the image space (i.e. make synthetic images look more real).
  - Hoffman et al., “CyCADA: Cycle-Consistent Adversarial Domain Adaptation”. In ICML, 2018.
  - Richter et al., “Enhancing photorealism enhancement”. In TPAMI 2022.

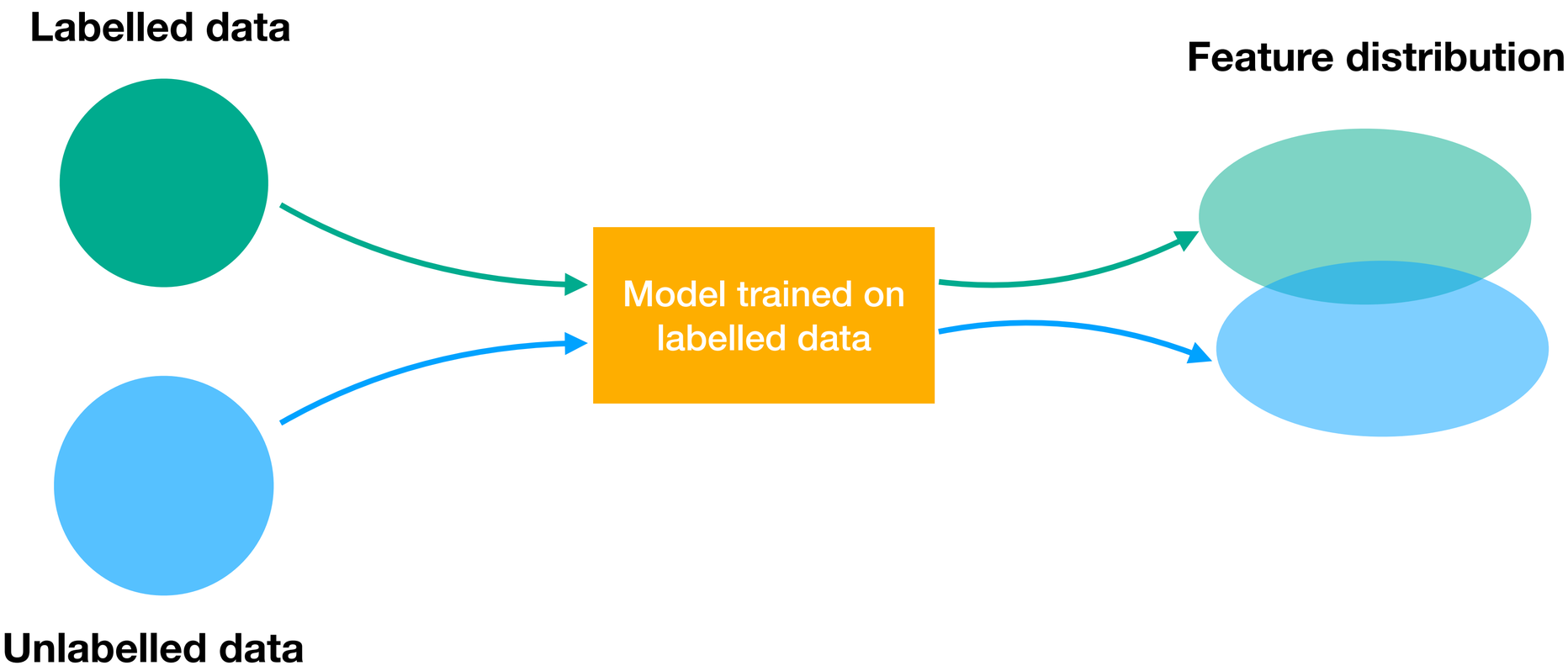
Summary

- Entropy minimisation: minimise the entropy of class distribution.
  - improves accuracy, but lead to miscalibration.
- Consistency regularisation: leverage invariance or equivariance constraints.
  - very effective, but it limited by available augmentation techniques.
- Domain alignment: learn a joint feature space.
  - typically less fine-tuning required;
  - but can be still challenging to train (GAN).

Unsupervised learning

Learning from pretext tasks

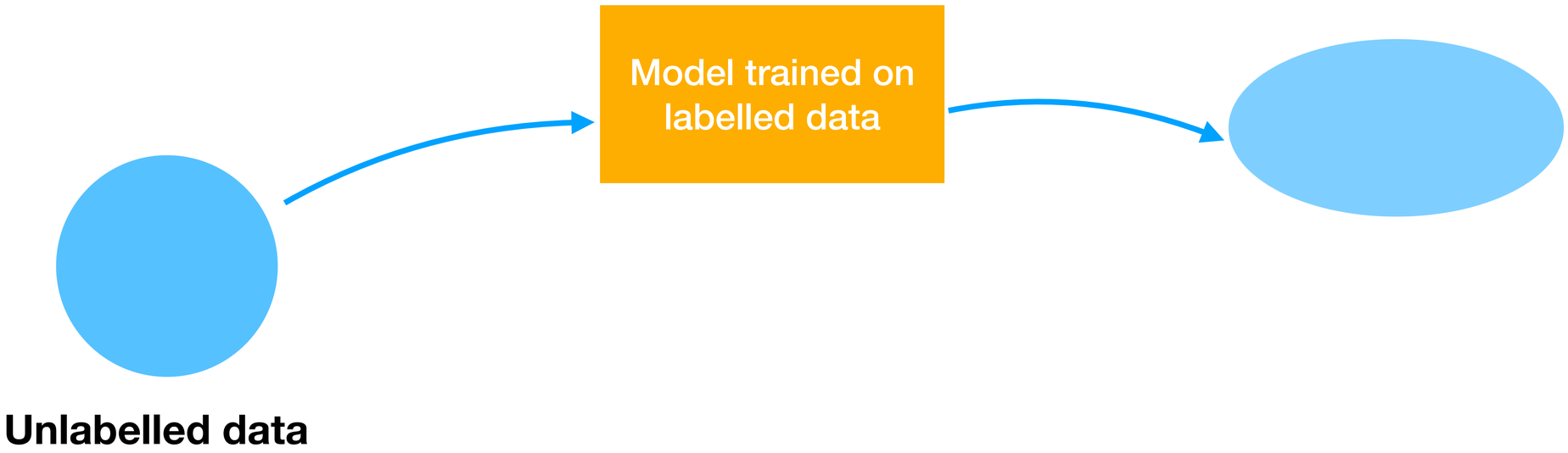
Suppose we do not have any labelled data:



Learning from pretext tasks

Suppose we do not have any labelled data:

- Can we hope to learn anything useful from the unlabelled data?



Learning from pretext tasks

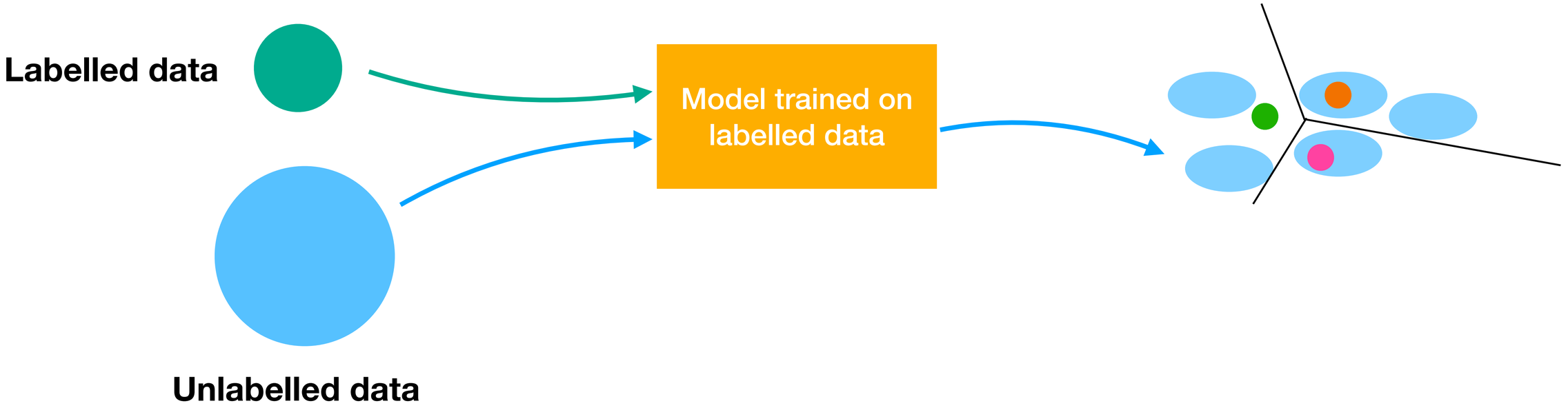
Suppose we do not have any labelled data:

- Can we hope to learn anything useful from the unlabelled data?
- Idea: Learn useful feature representation (e.g. linearly separable clusters);
  - assign meaningful labels with a few labeled examples and a simple classifier.

Learning from pretext tasks

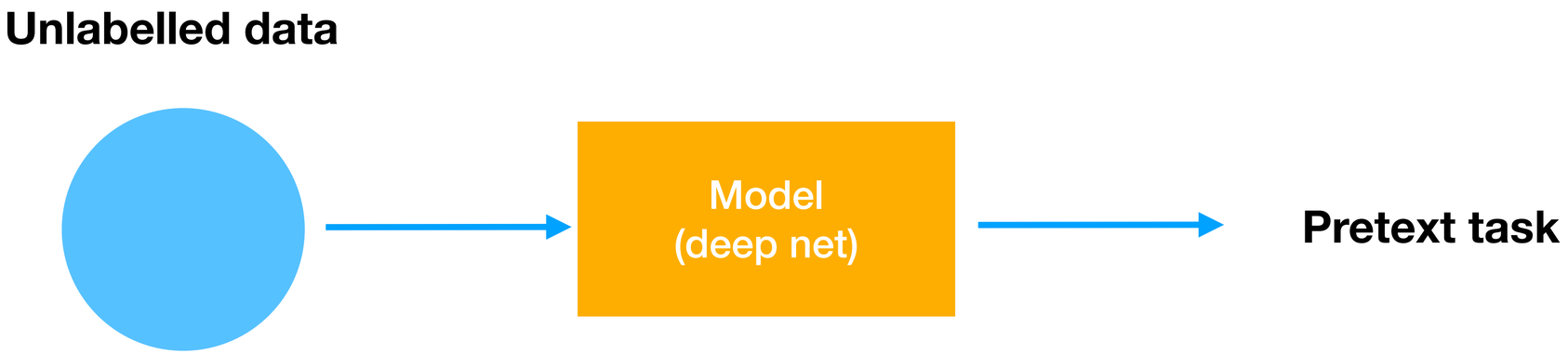
Suppose we do not have any labelled data:

- Can we hope to learn anything useful from the unlabelled data?
- Idea: Learn useful feature representation (e.g. linearly separable clusters);
  - assign meaningful labels with a few labeled examples and a simple classifier.



Learning from proxy tasks

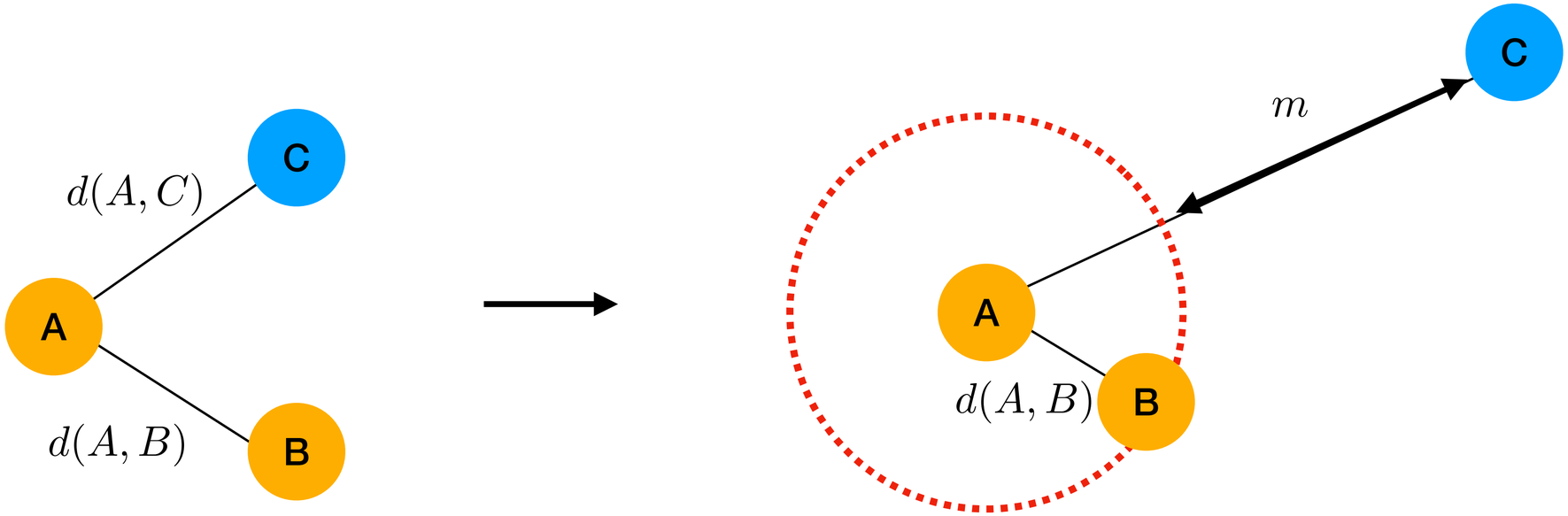
What are useful pretext tasks?



Recall metric learning

$$\mathcal{L}(A, B, C) = \max(0, \|f(A) - f(B)\|^2 - \|f(A) - f(C)\|^2 + m)$$

Intuitive idea:

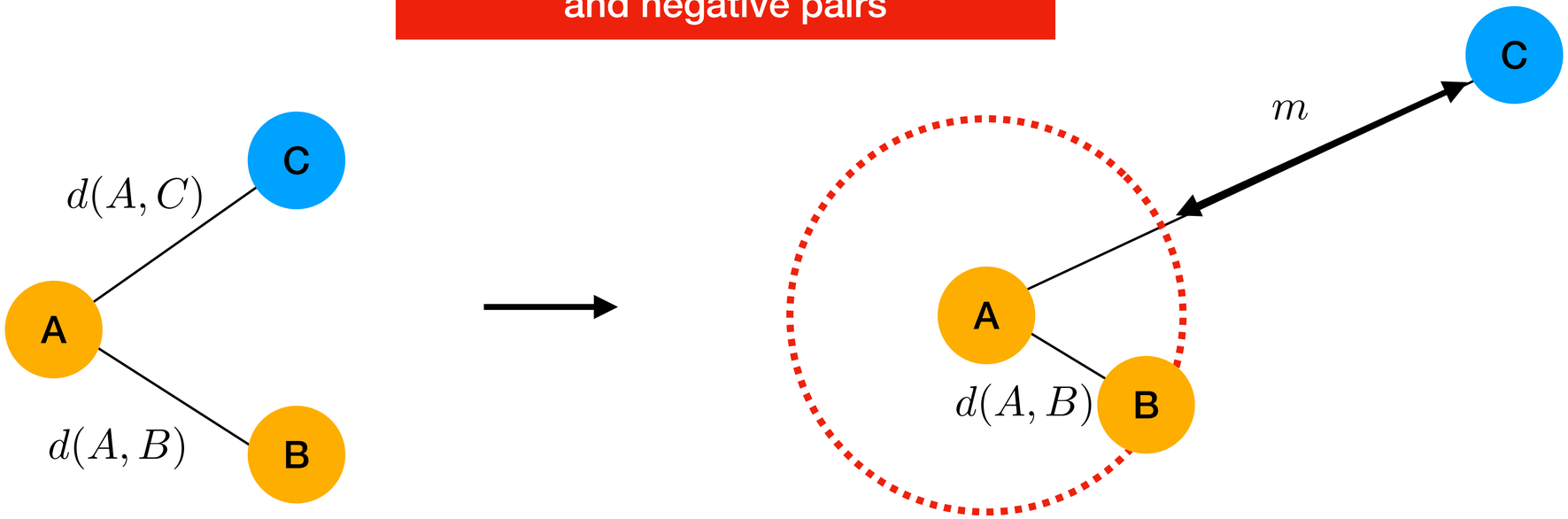


Recall metric learning

$$\mathcal{L}(A, B, C) = \max(0, \|f(A) - f(B)\|^2 - \|f(A) - f(C)\|^2 + m)$$

Intuitive idea:

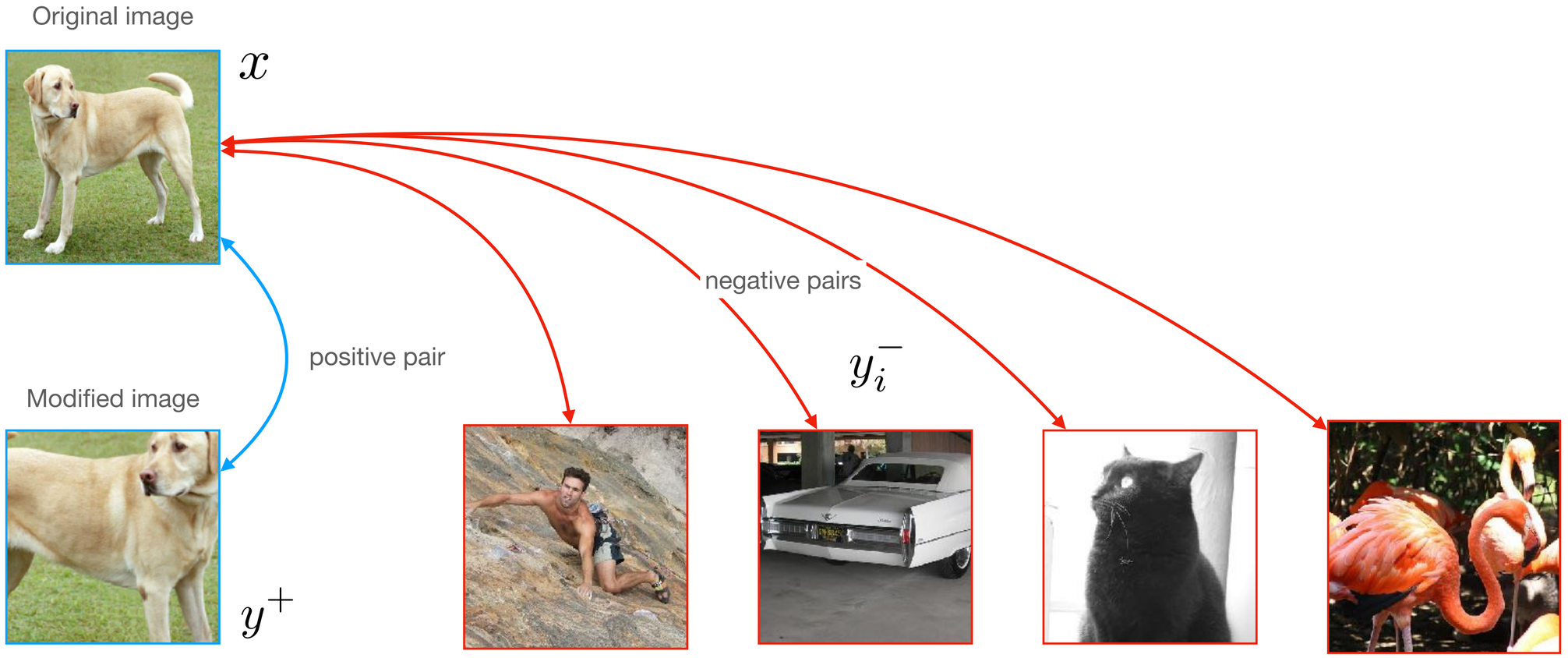
We need labels to define positive and negative pairs



Contrastive learning

- Contrastive learning is an extension of metric learning to unsupervised scenarios.
- Idea:
  - Use data augmentation (e.g. cropping) to create a positive pair of the same image;
  - Two image samples for a negative pair.

Contrastive learning: Example



Contrastive learning: Example

- Represent each image a single feature vector (e.g. last layer in a CNN).
- Consider normalised dot product of two such vectors:

$$d(x, y) = \frac{x \cdot y}{\|x\| \|y\|} \qquad \text{Note:} \quad d(x, y) \in [-1, 1]$$

- For a given set $\{x, y^+, \{y_i^-\}_{i=1,\ldots,n}\}$ consider contrastive score:

$$s(x) = e^{-d(x,y^+)/\tau} / \left(e^{-d(x,y^+)/\tau} + \sum_{i=1}^{n} e^{-d(x,x_i^-)/\tau}\right)$$

Contrastive learning: Example

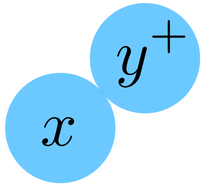
$$s(x) = e^{-d(x,y^+)/\tau} / \left(e^{-d(x,y^+)/\tau} + \sum_{i=1}^{n} e^{-d(x,x_i^-)/\tau} \right)$$

Observations:

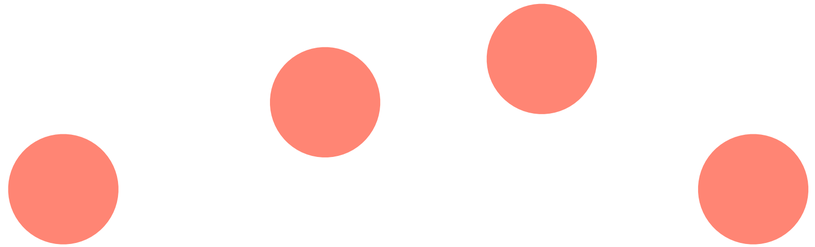
- Temperature τ – a hyperparameter (usually between 0.01 and 1.0).
- What is the range?
- What does it mean when it reaches maximum/minimum?
- We clearly want to maximise this value! (many implementations)
- Example loss: $-\log s(x)$

Contrastive learning: Example

Original image



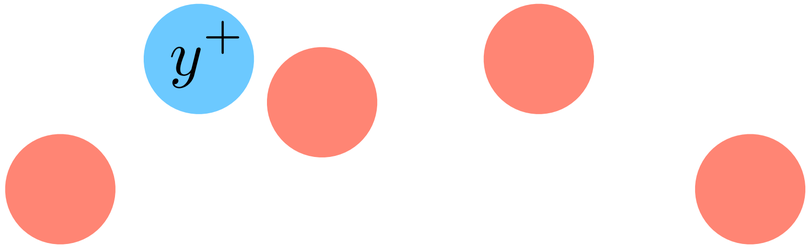
$$s(x) \approx 1$$



Contrastive learning: Example

Original image

$s(x) \approx 0$



Contrastive learning: Example

Original image

x

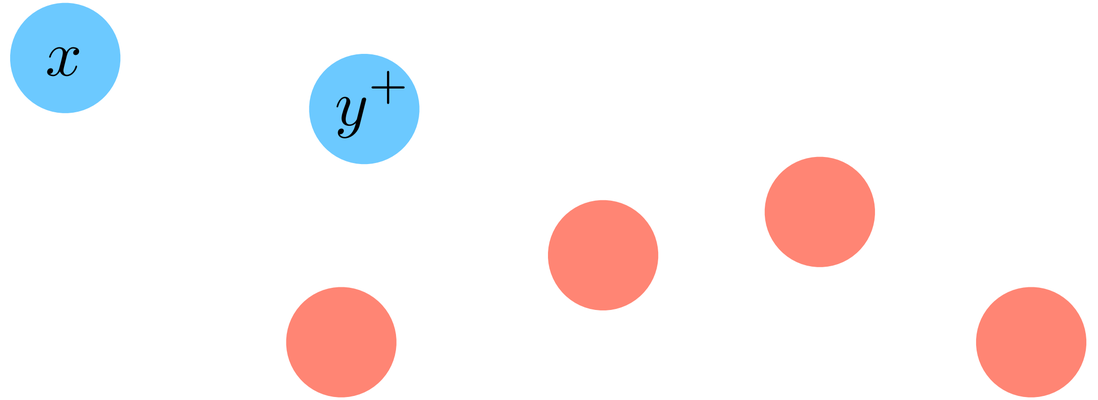
y^+

$s(x) \approx 0.5$

Contrastive learning: Example

Original image

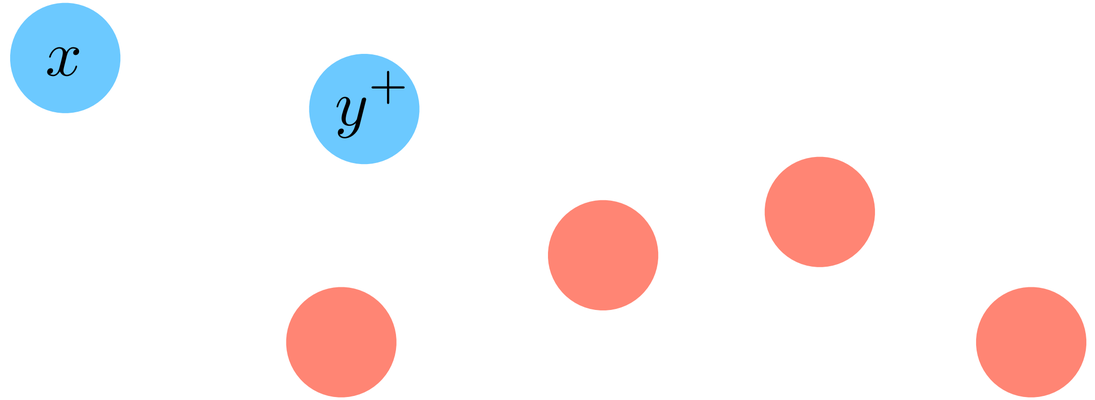
$s(x) \approx 1$ very low temperature



Contrastive learning: Example

Original image

$s(x) \approx 0$ very high temperature

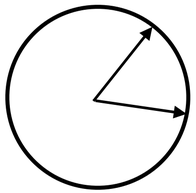


Contrastive learning: Intuition

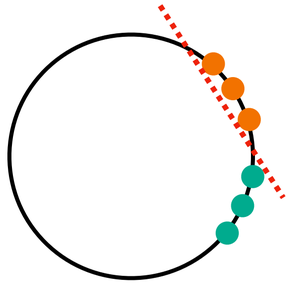
- Note that we normalise the feature embeddings:

$$d(x, y) = \frac{x \cdot y}{\|x\|\|y\|}$$

- Every unit vector corresponds to a point on a unit sphere:



- The goal of contrastive learning is to cluster the representation on the sphere:

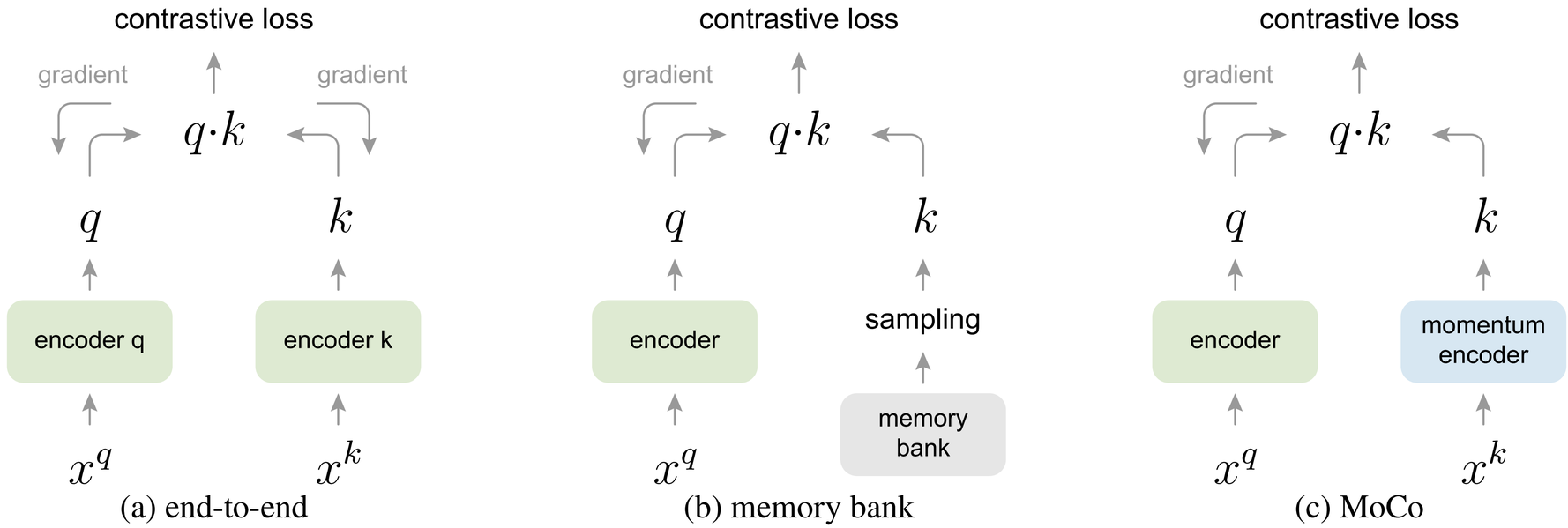


- The points on the sphere will be then linearly separable!

Wang and Isola, “Understanding Contrastive Representation Learning through Alignment and Uniformity on the Hypersphere”. In ICML 2020.

Momentum contrast

- Contrastive learning requires large sets of negative pairs
  - in the order of thousands...



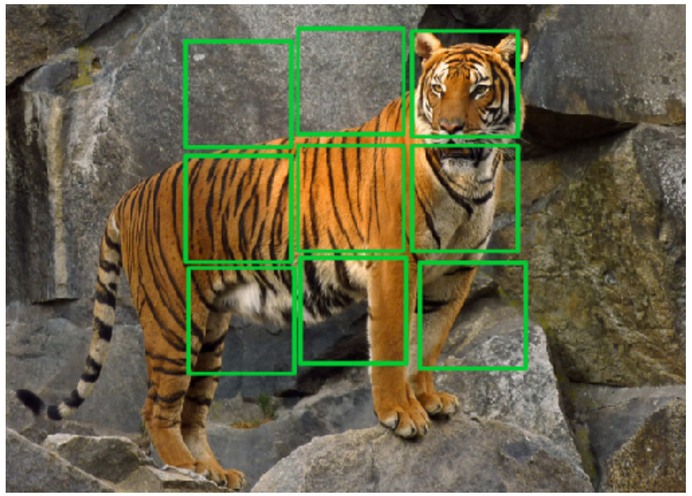
He et al., “Momentum Contrast for Unsupervised Visual Representation Learning”, CVPR 2020.

How to evaluate features?

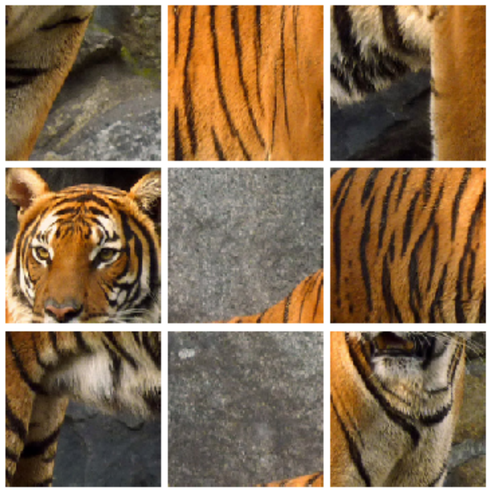
- Fine-tuning on the downstream tasks:
  - either all or only few last layers.
- Linear probing:
  - train a linear classifier on top of the learned features.
- k-nearest-neighbour (k-NN) classification:
  - select k nearest neighbours in the learned embeddings space;
  - select the dominant label as the class prediction.

Jigsaw puzzle

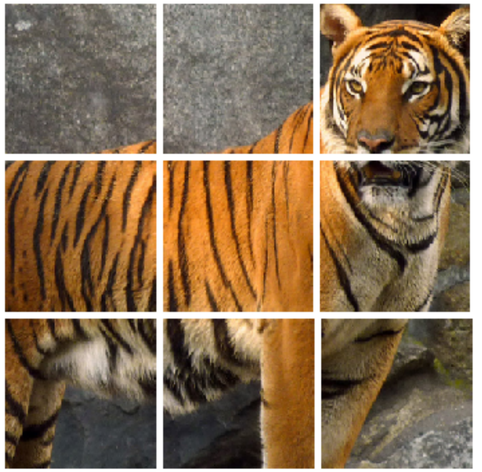
1. Extract patches



2. Input: shuffle



3. Output: reconstruct

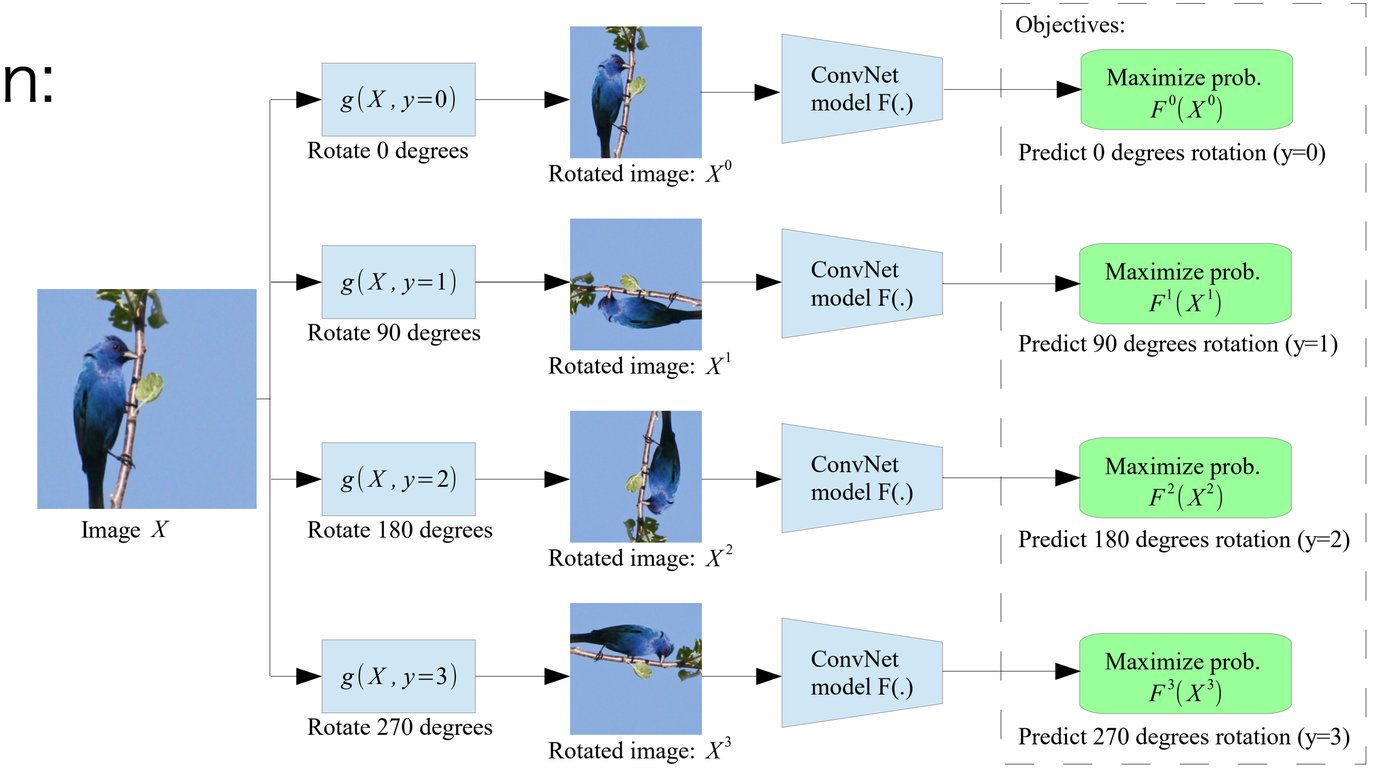


- Solving this task requires the model to learn spatial relation of image patches.

Noroozi and Favaro, “Unsupervised learning of visual representations by solving jigsaw puzzles”. In ECCV, 2016.

Rotation

- Predicting image rotation:

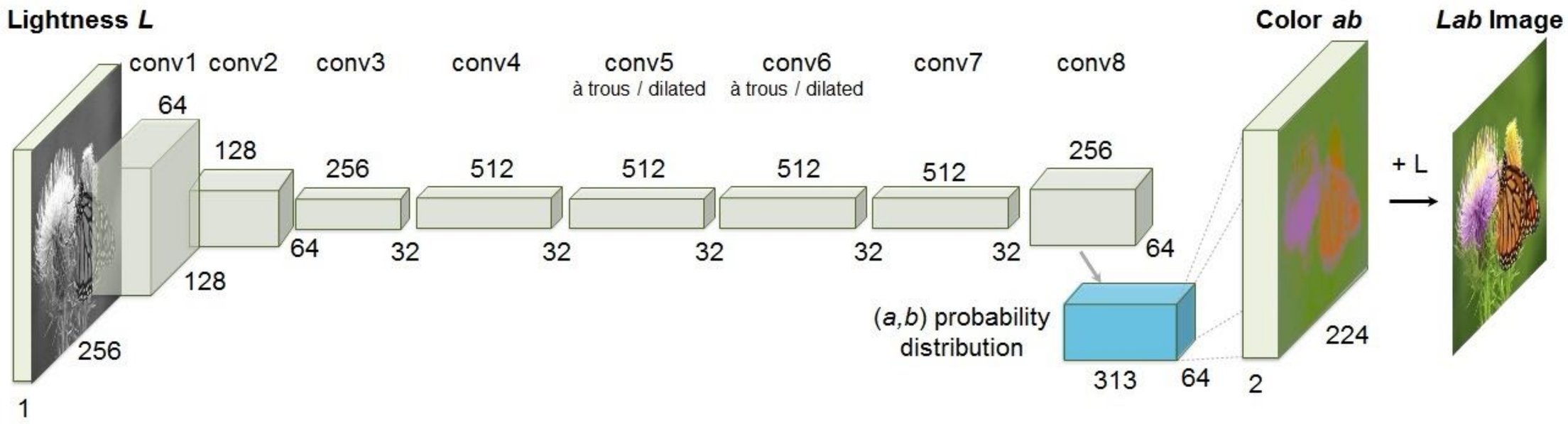


- This leverages the photographic bias in common image sets
  - i.e. photographed objects have prevalent orientation.

Gidaris et al., “Unsupervised representation learning by predicting image rotations”. In ICLR, 2018.

Colorization

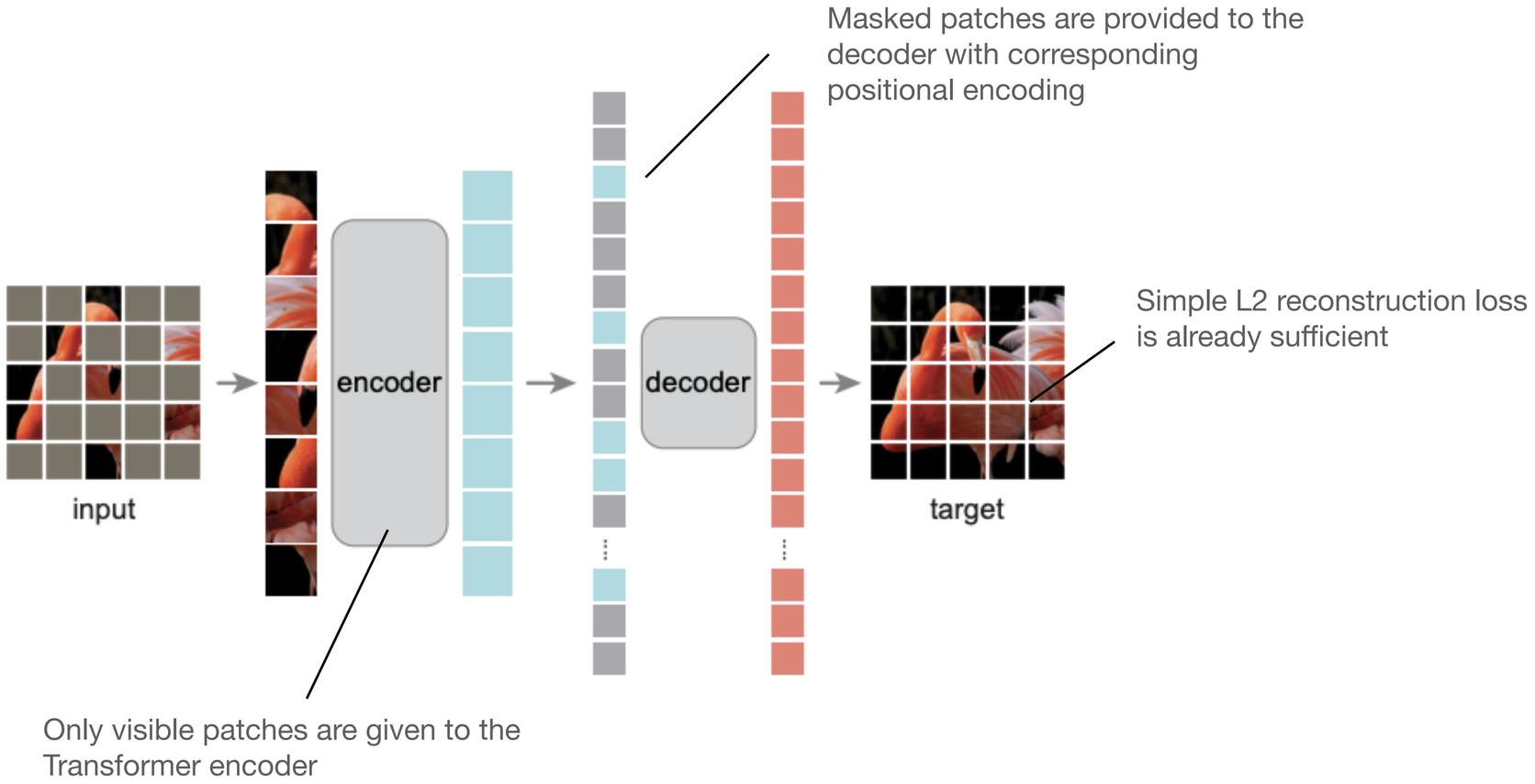
- Predicting the original colour of black-and-white images:
  - Intuition: proper colourisation requires semantic understanding of the image.



Zhang et al., “Colorful image colorization”. In ECCV, 2016.

Masked Autoencoders

Unsupervised learning with Transformers and a reconstruction loss:



He et al., “Masked Autoencoders Are Scalable Vision Learners”. In CVPR 2022.

Unsupervised learning

The list goes on...

- Chen et al., “A Simple Framework for Contrastive Learning of Visual Representations” (2020).
- Wei et al., “Masked Feature Prediction for Self-Supervised Visual Pre-Training” (2022).
- DINO: Caron et al., “Emerging Properties in Self-Supervised Vision Transformers” (2021).
- Caron et al., “Location-Aware Self-Supervised Transformers” (2023).
- Jabri et al., “Space-time correspondence as a contrastive random walk” (2020).
- Araslanov et al., “Dense unsupervised learning for video segmentation” (2021).

and many more...

Unsupervised learning

DINO: Caron et al., “Emerging Properties in Self-Supervised Vision Transformers” (2021).

Unsupervised learning

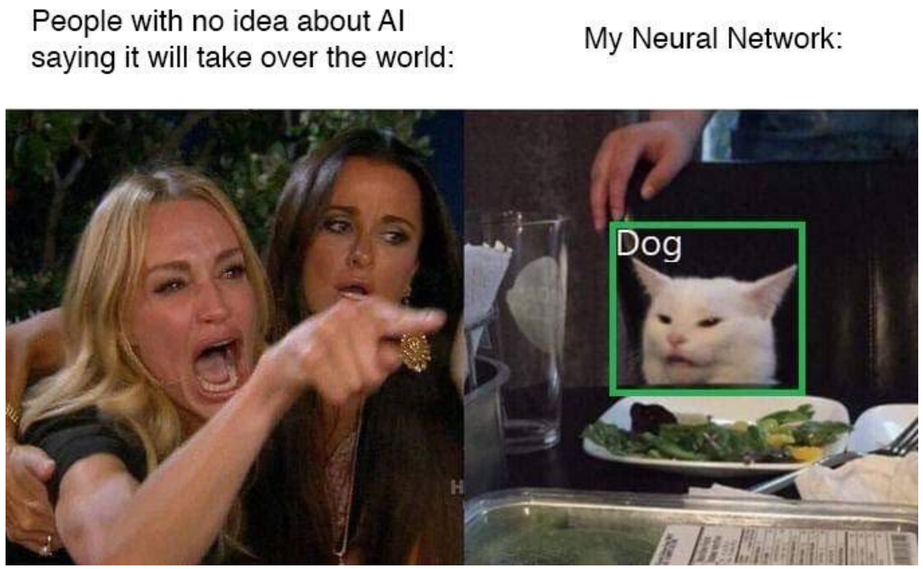
DINO: Caron et al., “Emerging Properties in Self-Supervised Vision Transformers” (2021).

Conclusion

- Unsupervised learning dominates research landscape
  - We can train more accurate models with less supervision.
- Requires large computational resources (dozens of high-end GPUs).
- Yet do not scale well with the amount of data (saturation).
- Many open questions:
  - What is a good proxy task?
  - How to make computational requirements manageable?
  - How (and/or why) does it work?

Feeling challenged?

- We work on many exciting open research problems.
- Get in touch if you're interested!
  - Guided research, job as student research assistant, etc.
- Next semester:
  - New practical course:
    Geometric scene understanding
  - First meeting (Zoom): February 8, 11:00



This is our last lecture

- Next up: Q&A on February 21 – see Moodle for details.
  - ask me anything about the lecture material and/or exam;
  - you can post questions in advance on Moodle.

Computer Vision III:

Semi- and Unsupervised learning

Nikita Araslanov

24.01.2023