Vision and Sign Language

Marek Hruz

Department of Cybernetics, Faculty of Applied Sciences

University of West Bohemia in Pilsen

Czech Republic

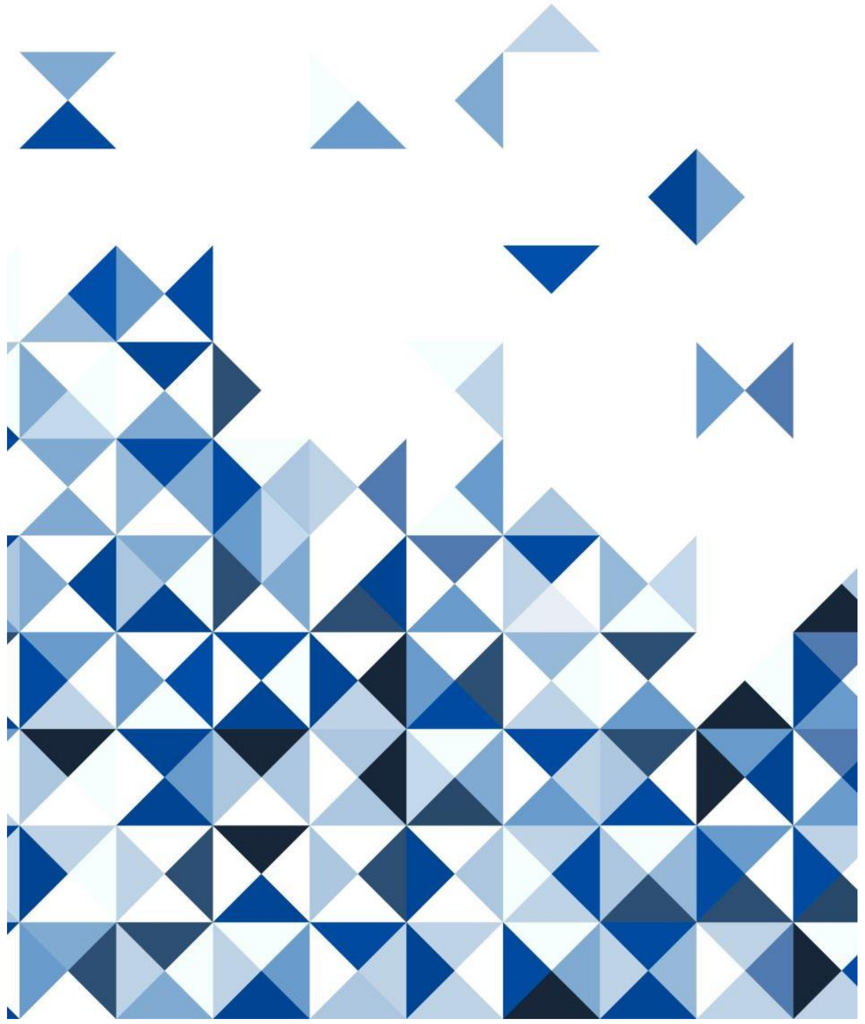
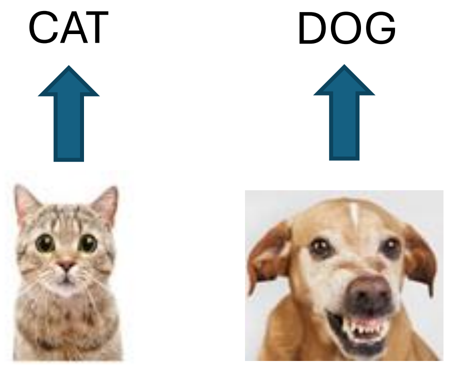


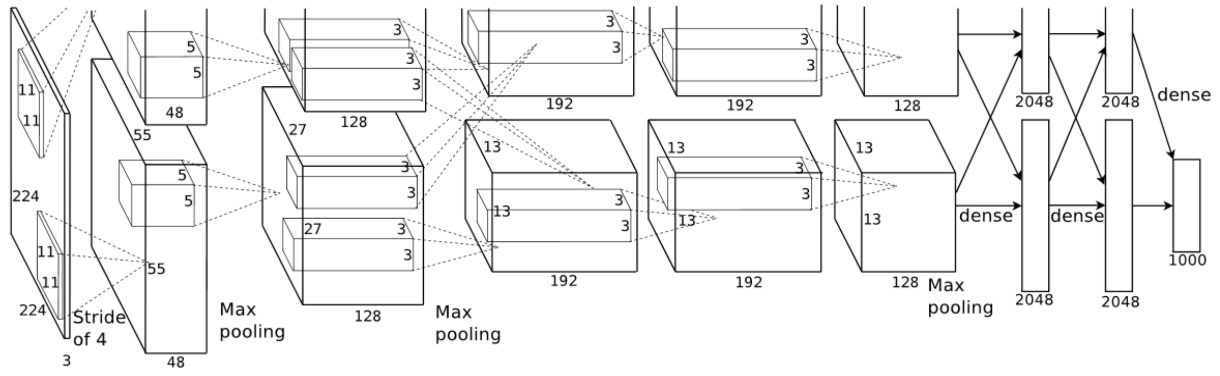
Image Classification
Convolution FTW?

Image Classification

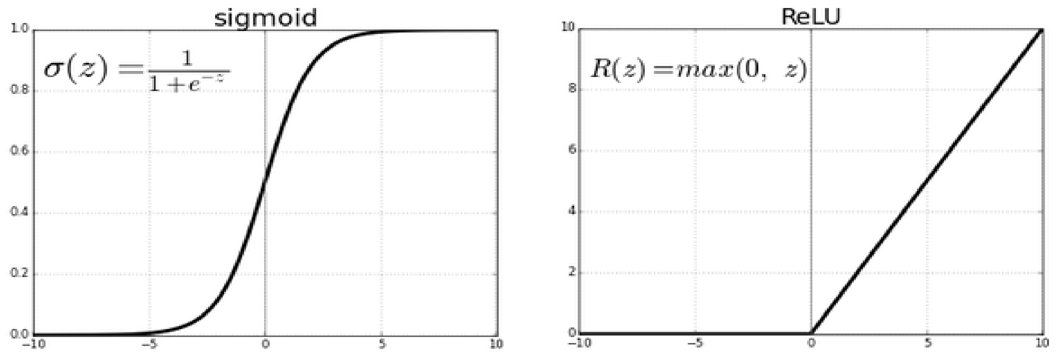


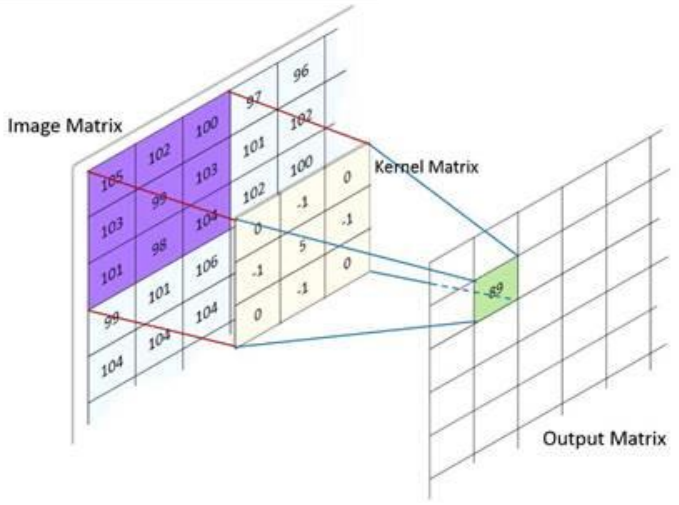
- Taks of assigning a class label to an image
- Historically, handcrafted features and trained a classifier
- Works reasonably well for small, non-complex, well-defined data
- To allow the next big leap in computer vision a new dataset was developed
- ImageNet - an image dataset organized according to the WordNet hierarchy
  - Total number of non-empty synsets: 21,841
  - Total number of images: 14,197,122
  - Number of images with bounding box annotations: 1,034,908
- SIFT + FVs in 2011 achieved Top-1 accuracy of 50.9% (1000 classes)

AlexNet 2012



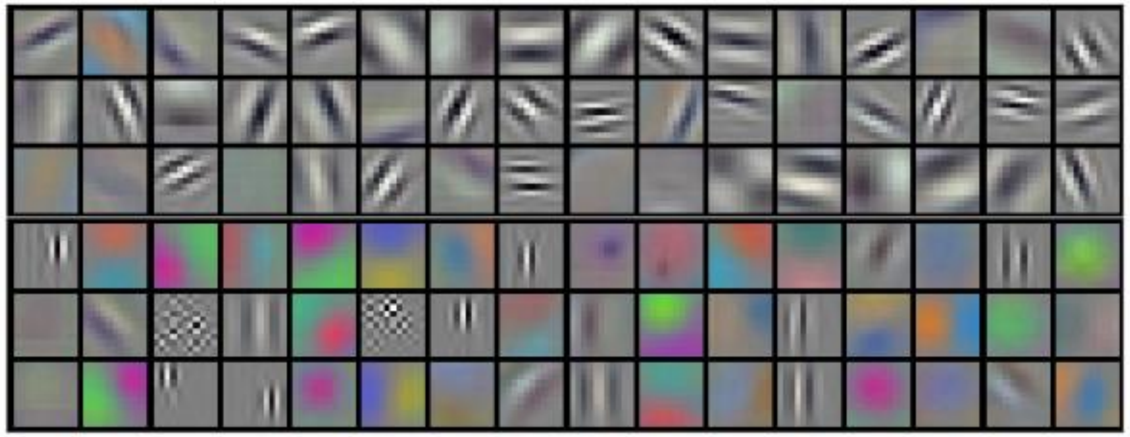
- Introduces Convolutional Neural Networks to the task of ImageNet classification
- The CNN is distributed on 2 GPUs
- Main features:
  - Novel, deep CNN architecture
  - ReLU non-linearity
  - Overlapping max-pooling
  - Data augmentation for overfitting reduction
  - Dropout
- Achieves Top-1 accuracy of 63.3%

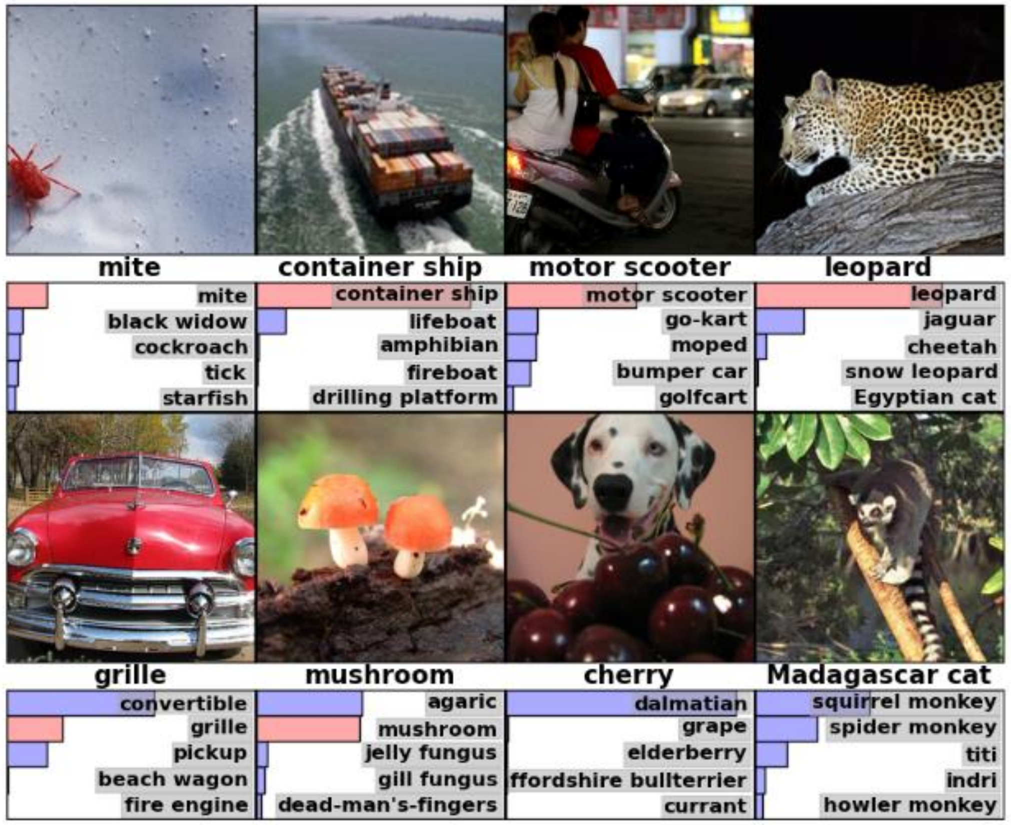


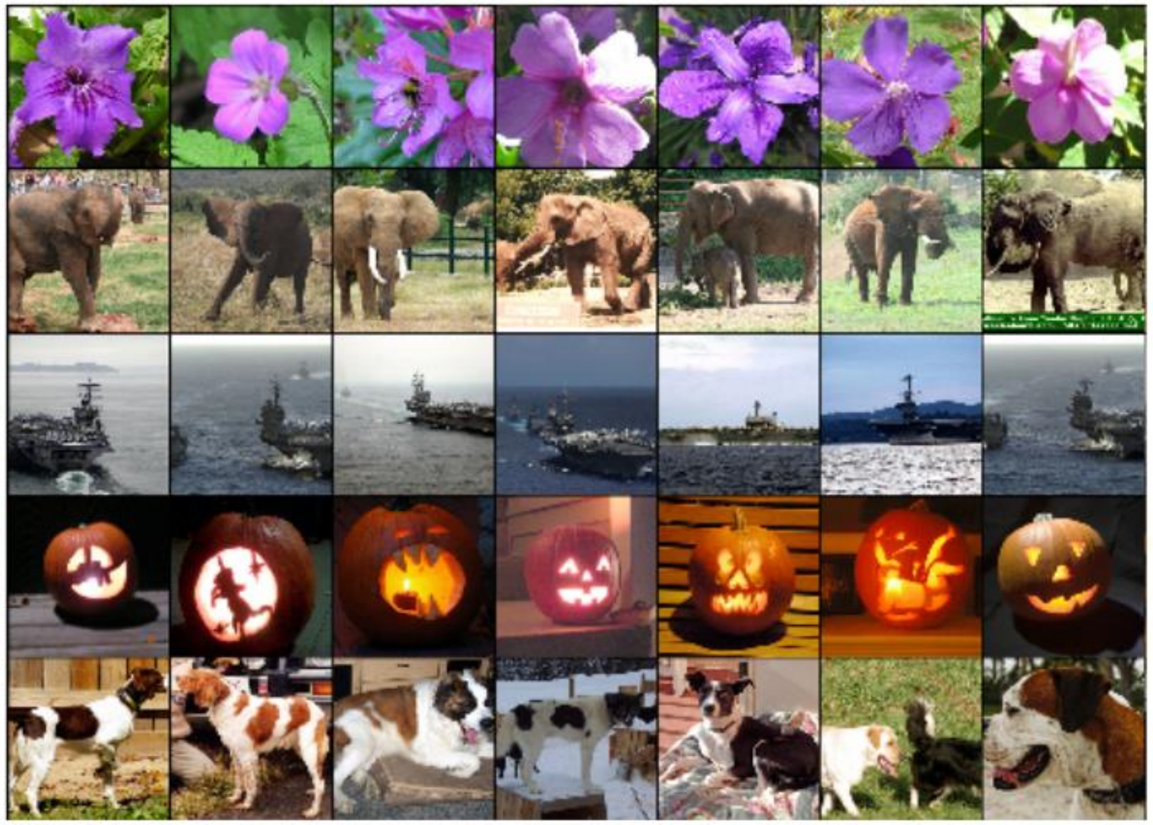


CNN implications

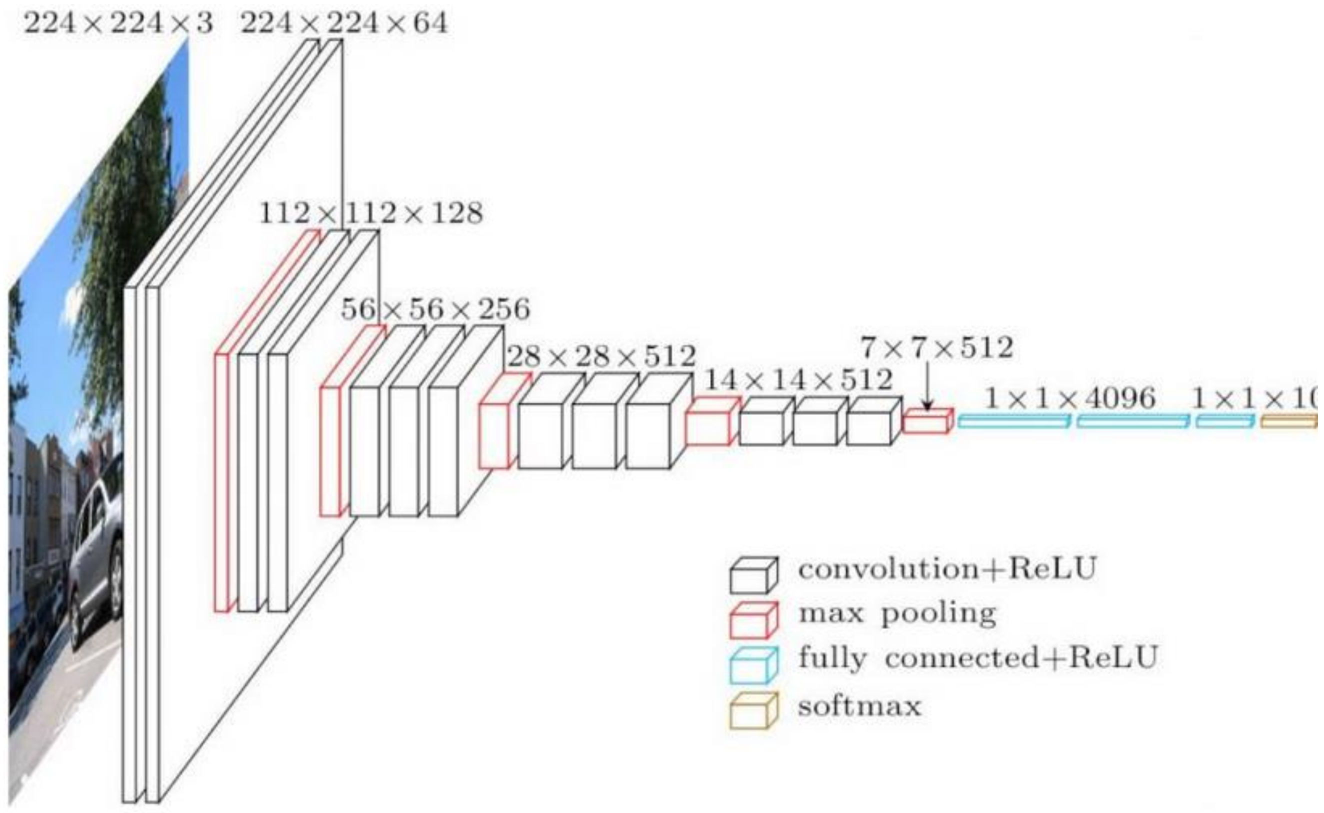
- Convolutional filters can be visualized and explained
- The shift from algorithm design to data preparation
- Semantic representation in deep layers







VGG 2014

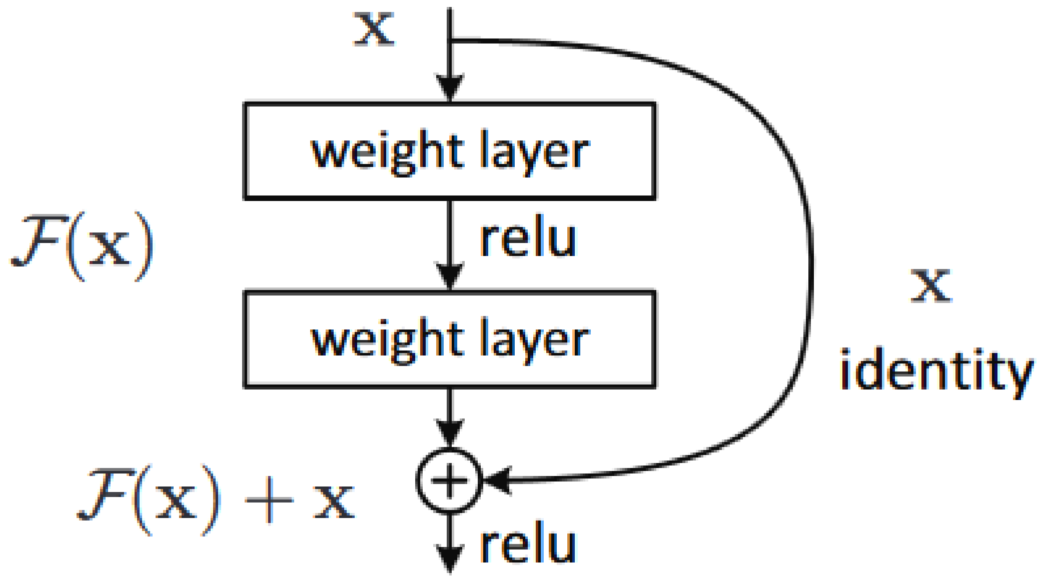


- From Google
- Scheme for deeper models
- Small sizes of convolution kernels (3x3 instead of 11x11)
- Back to non-overlapping pooling
- Achieves Top-1 accuracy of 74.4%

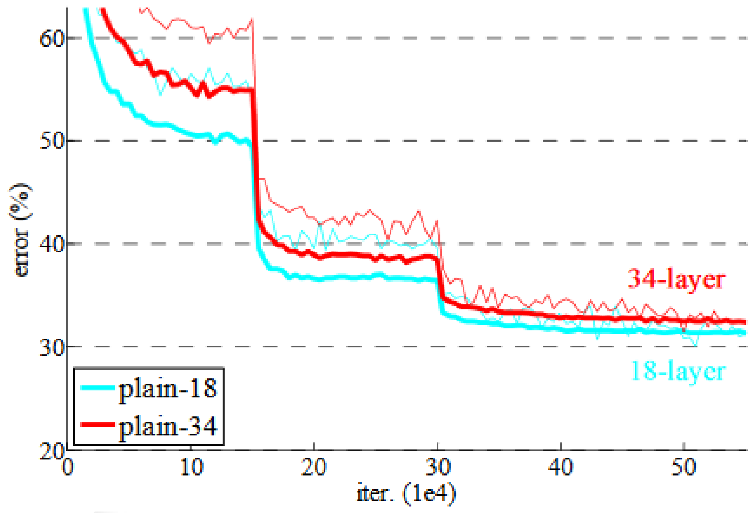
ResNet 2015

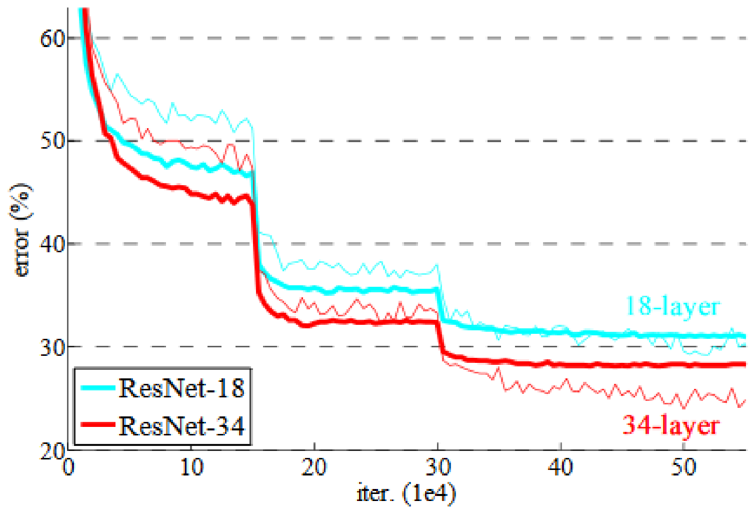
- Addresses the problem of **vanishing** and **exploding** gradients
- Introduces residual (skip) connections
- They allow a better flow of the gradient
- Achieves Top-1 accuracy

  of 78.6%



x
weight layer
$\mathcal{F}(\mathbf{x})$
relu
weight layer
x
identity
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu

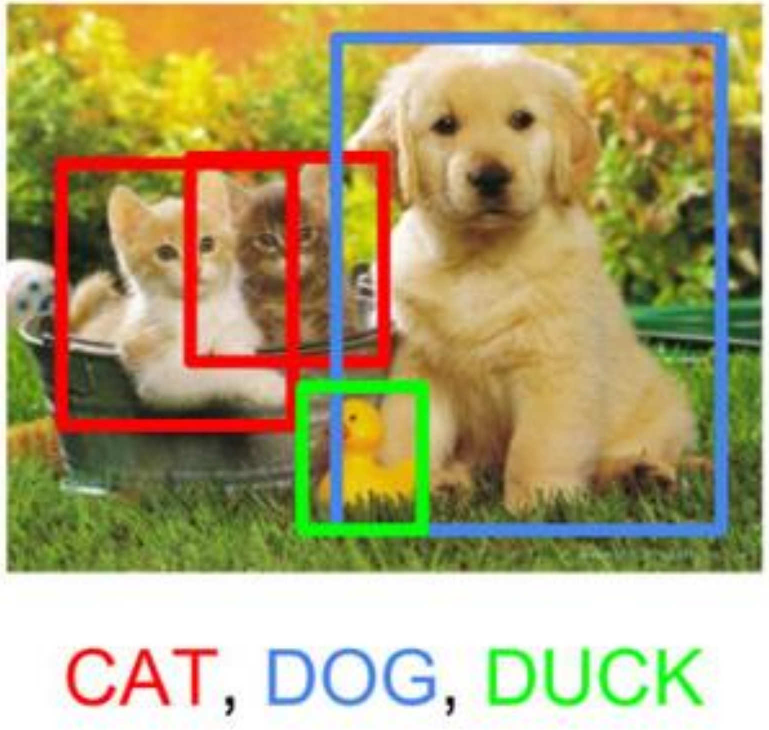




Convolutions can Detect Objects?

Object Detection

- Task of recognizing and localizing an object
- Historically, two stages:
    - Region proposal
    - Region classification



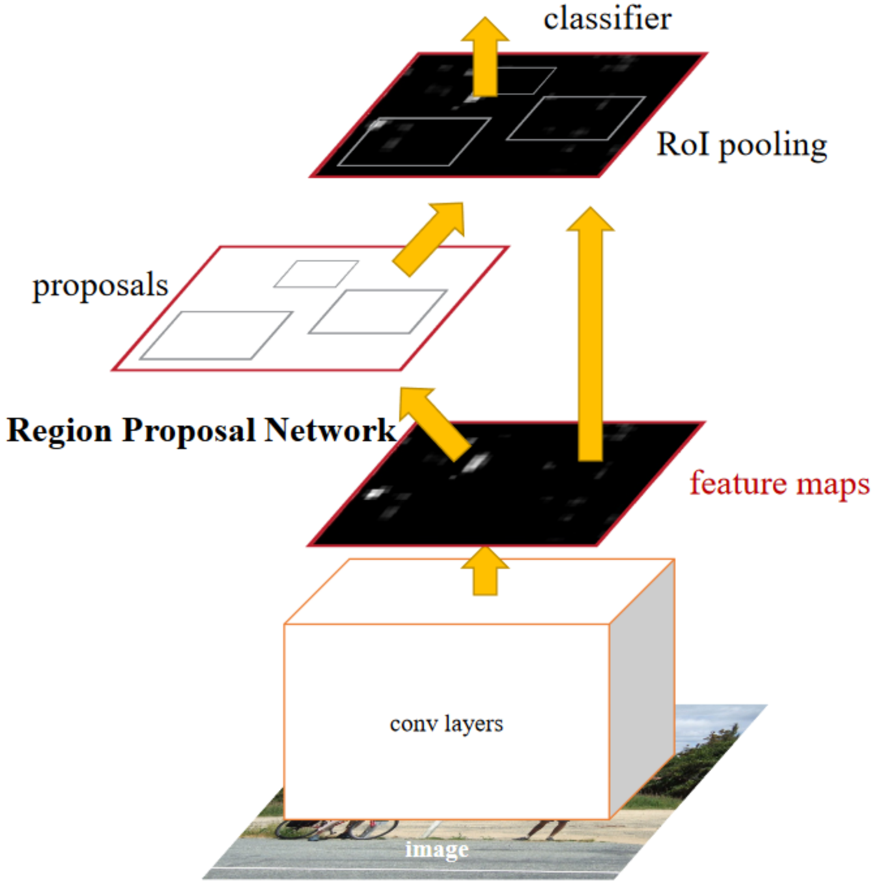
- MS COCO
    - Object segmentation
    - Recognition in context
    - Superpixel stuff segmentation
    - 330K images (>200K labeled)
    - 1.5 million object instances
    - 80 object categories
    - 91 stuff categories
    - 5 captions per image
    - 250,000 people with key points

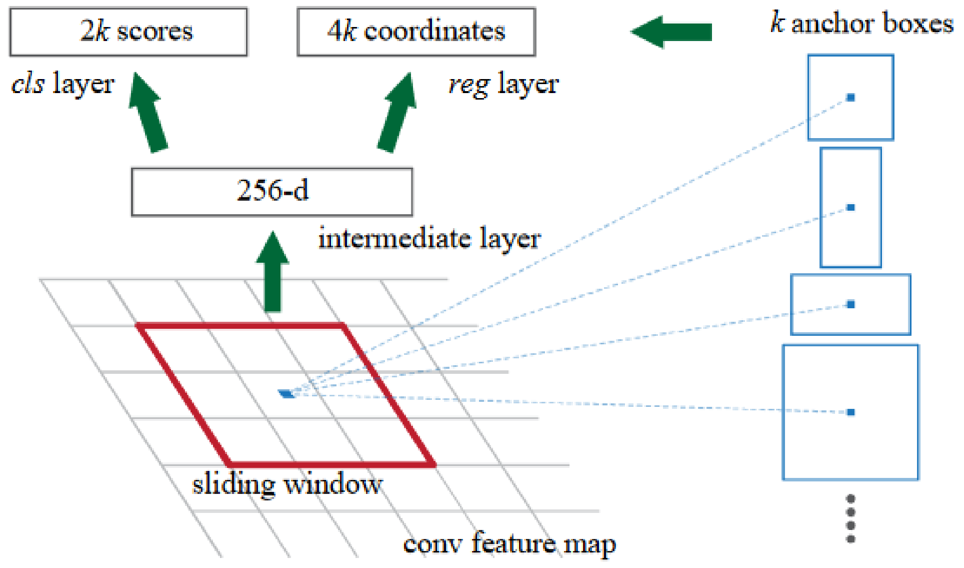
- Pascal VOC
    - 9,993 annotated images

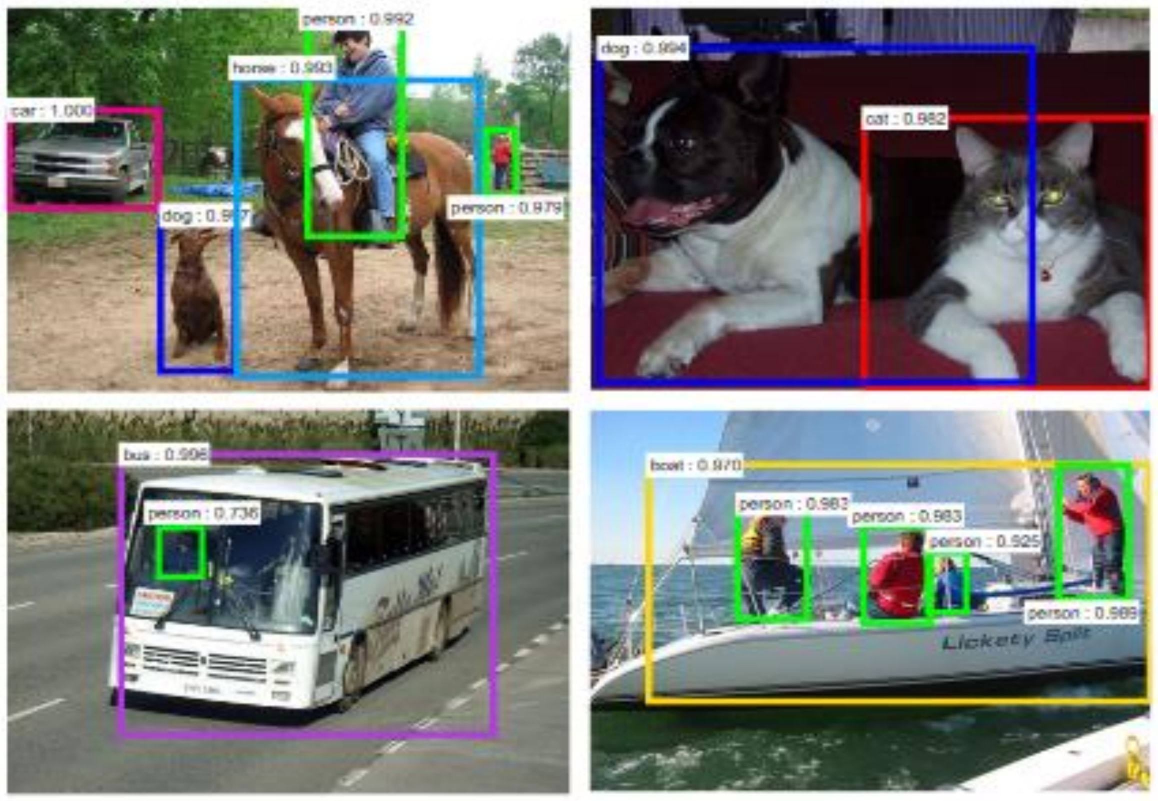
- Objects365
    - 365 categories
    - 2 million images
    - 30 million bounding boxes

Faster R-CNN 2015

mAP@0.5 = 42.7% for COCO

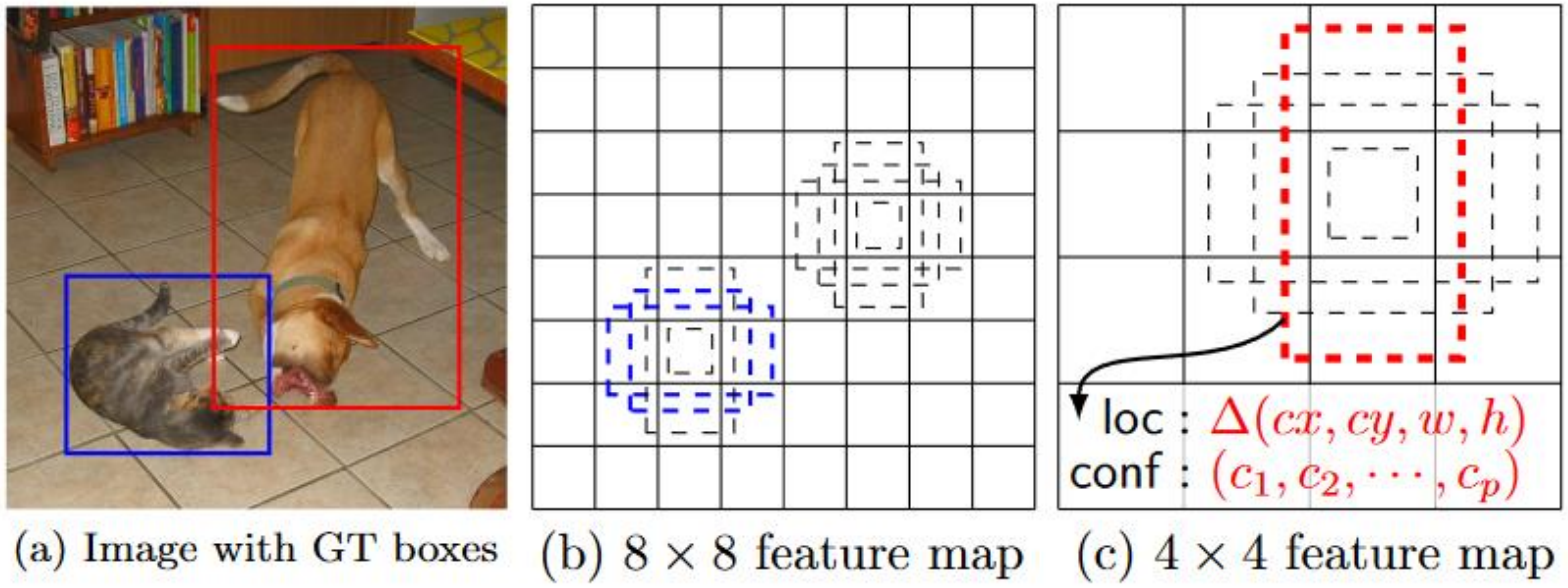


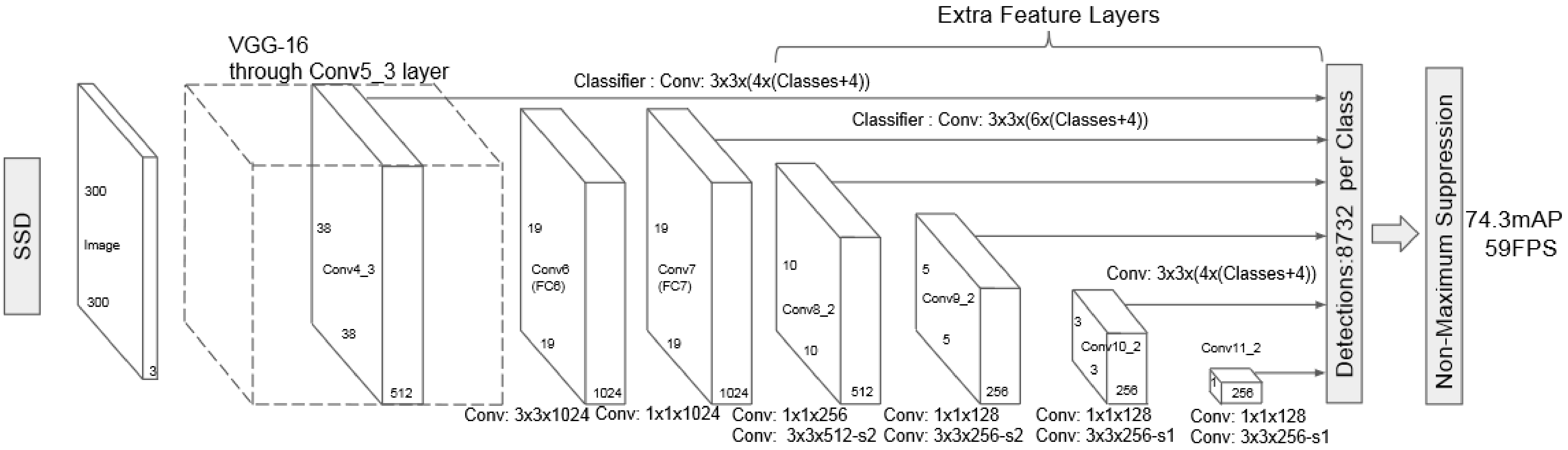




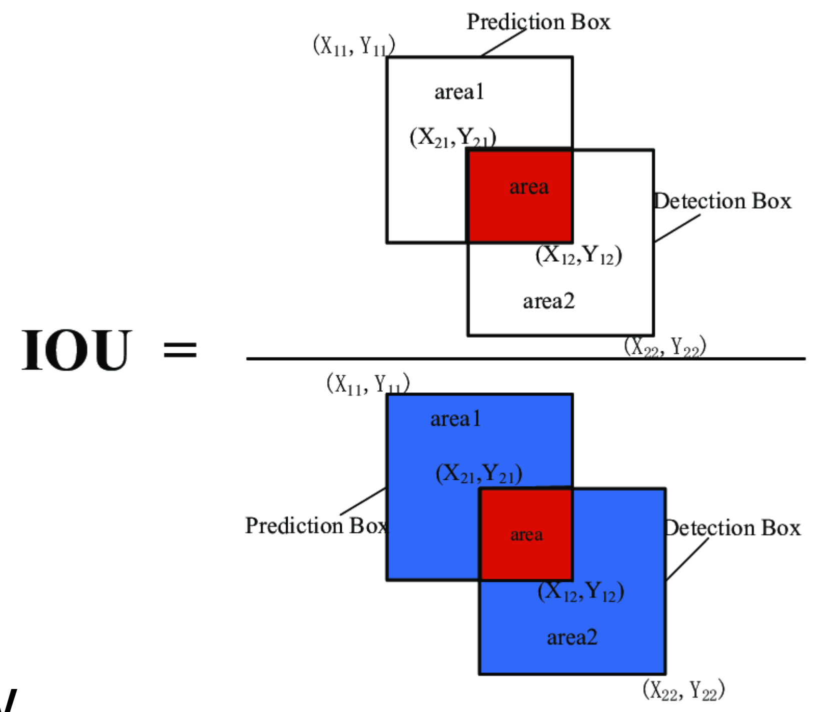
SSD early 2016

mAP@0.5 = 46.5% for COCO





Learning and Inference



- If there is sufficient IoU of the GT box and the anchor (default) box, then it is considered a positive sample.
- Too many negative boxes $\longrightarrow$ only take a few.
- Loss is the sum of classification and regression error.
- During inference more boxes can be predicted at the same position $\longrightarrow$ non-maximum suppression

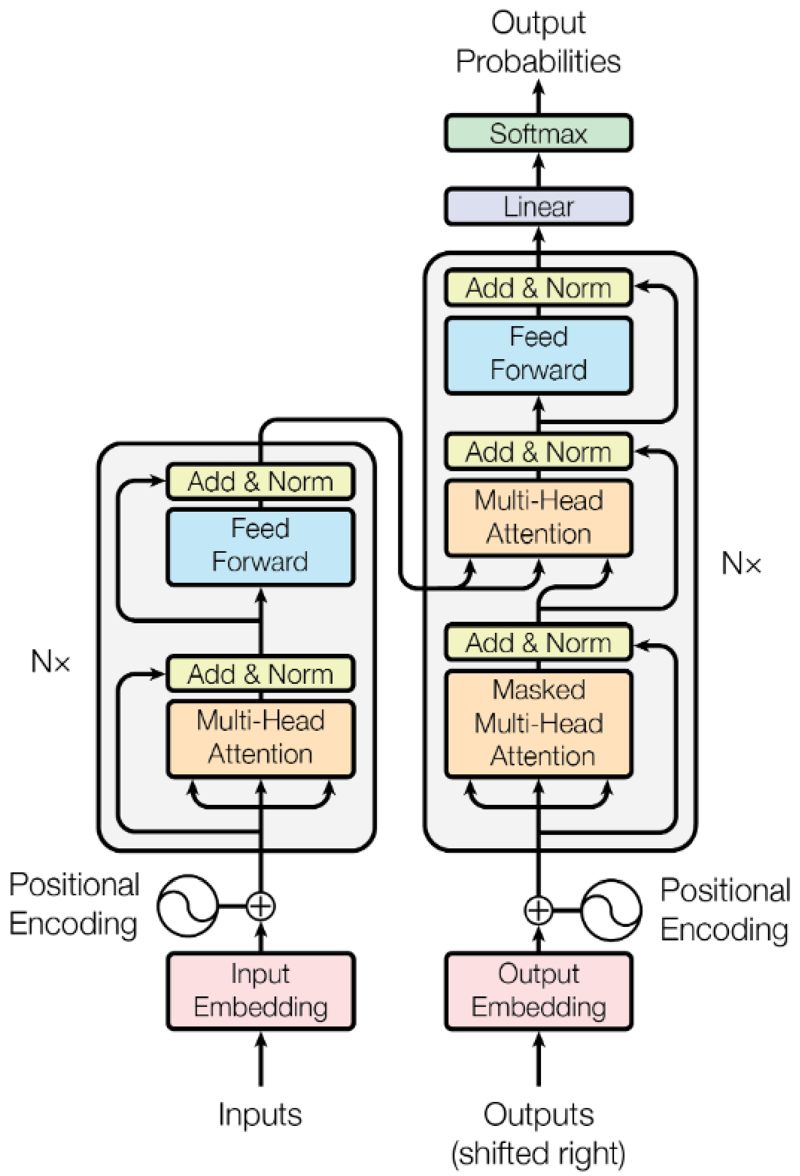
(Vision) Transformers
Next big leap?

Transformers

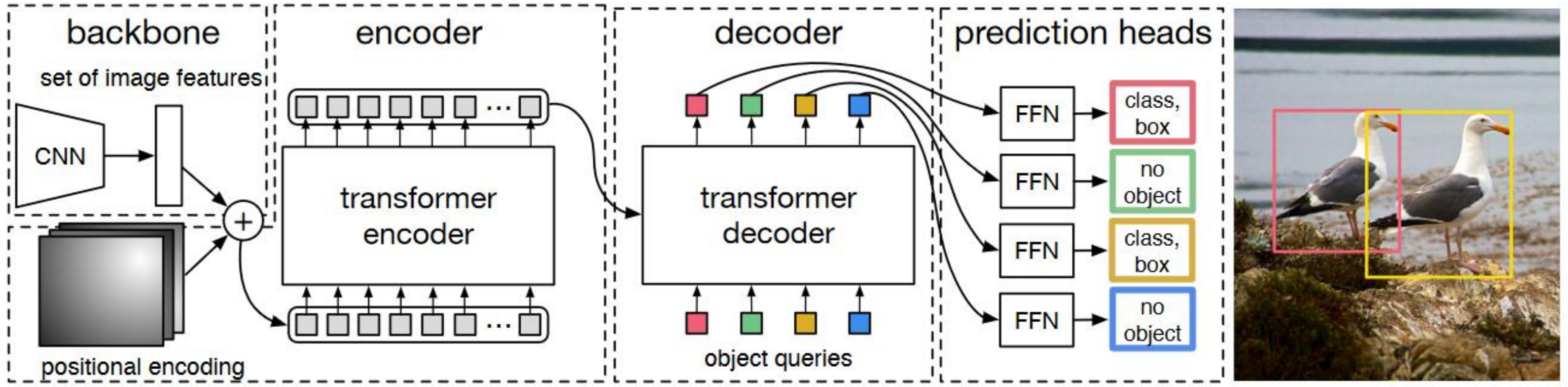
- Attention Is All You Need
- Paper from **June 2017**.
- Introduces the idea of using exclusively an **Attention mechanism** in sequence processing deep models.
- Removes modeling of a **state**.
- The *Encoder* sees the whole sequence at once (or a part of it that the memory allows).
- The *Decoder* can work either:
    - **Autoregressively** (the currently output TOKEN is dependent on the previous ones)

  or

    - **Non-autoregressively** (all TOKENs are output at once dependent on each other)



Detection Transformer



- End-to-End Object Detection with Transformers
- Paper from May 2020. ***DETR*.**
- Uses a (pre-trained) **CNN** to extract high-level visual features.
- Processes these features as a sequence by a Transformer.
- Employs a **2D Positional Encoding** to help model the geometry in an image.
- First half of PE encodes the x position, second half the y position.
- A non-auto regressive decoder predicts class and box of an object.

Emergent properties

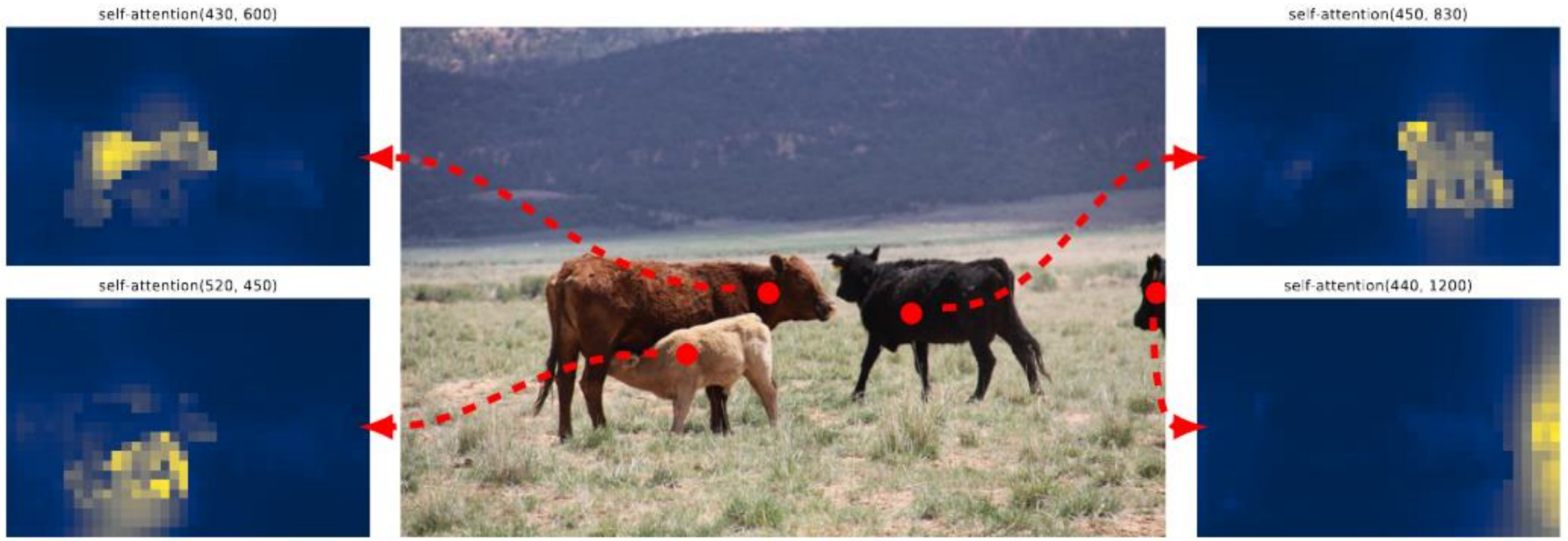


Fig. 3: Encoder self-attention for a set of reference points. The encoder is able to separate individual instances. Predictions are made with baseline DETR model on a validation set image.

Panoptic Segmentation (almost) for free

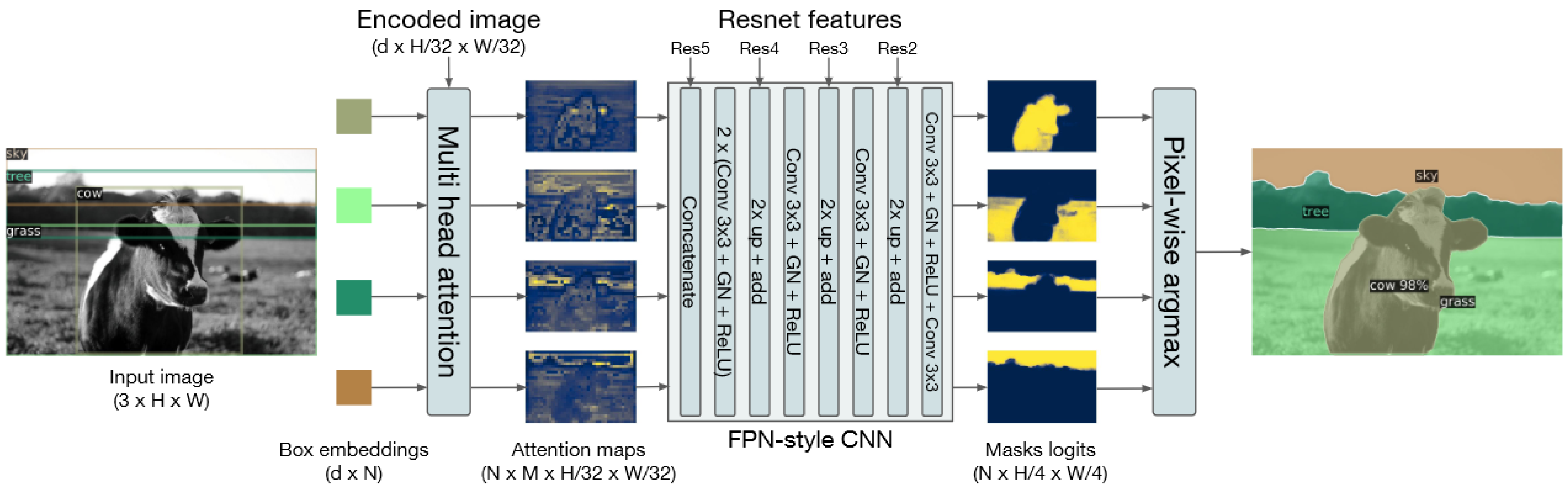


Fig. 8: Illustration of the panoptic head. A binary mask is generated in parallel for each detected object, then the masks are merged using pixel-wise argmax.

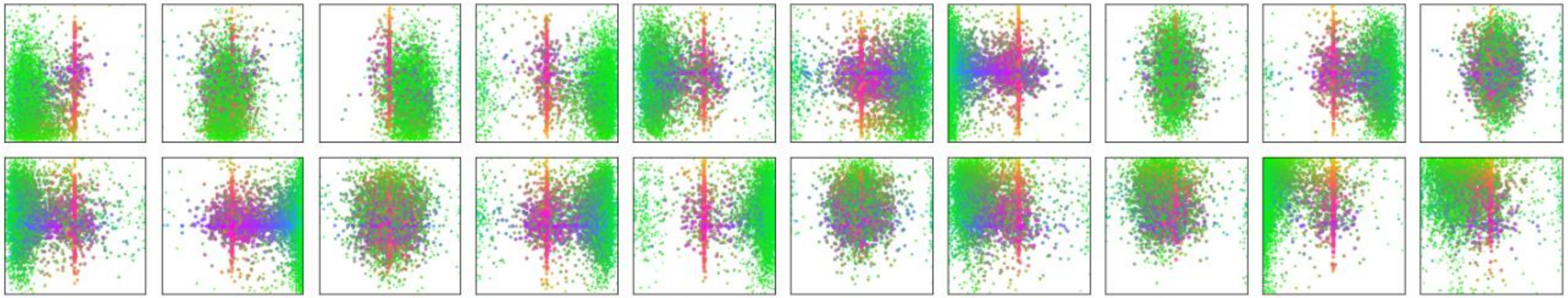


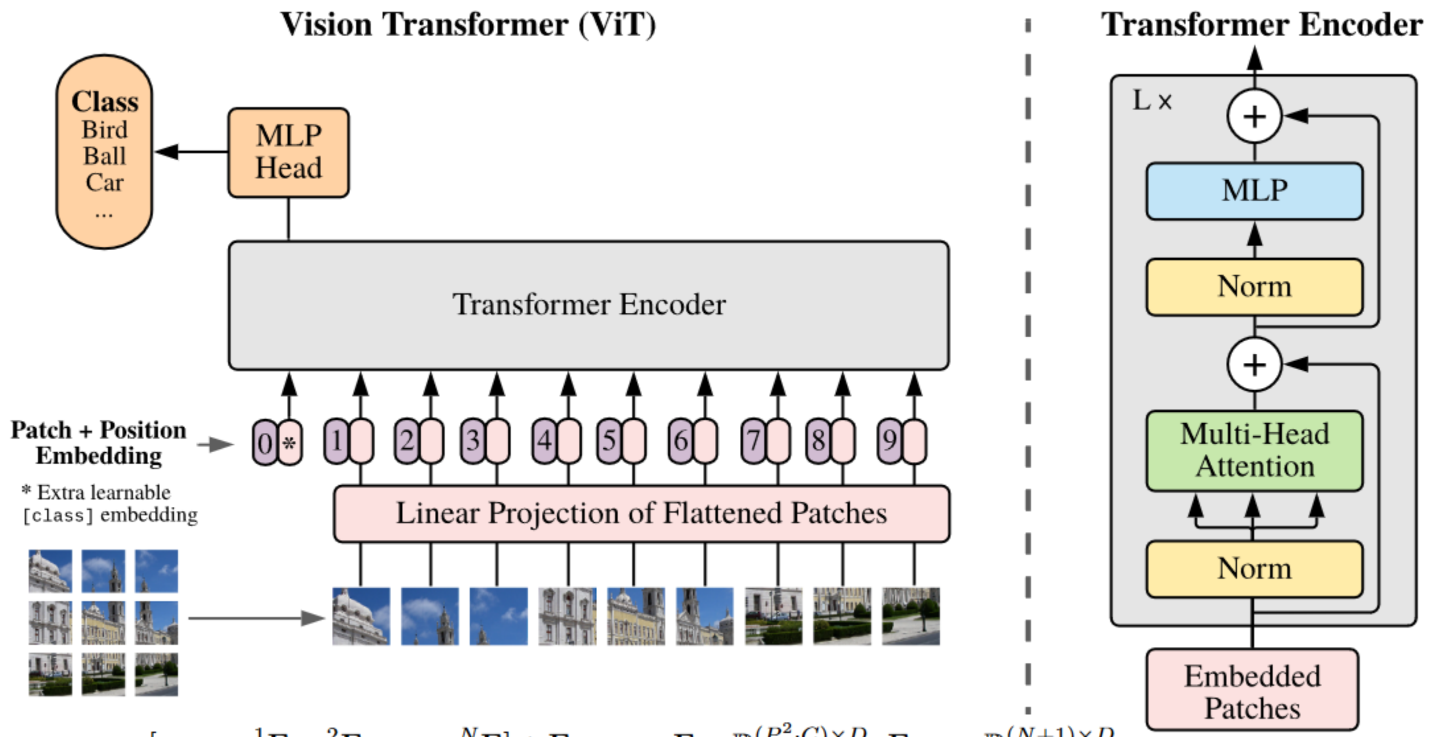
Fig. 7: Visualization of all box predictions on all images from COCO 2017 val set for 20 out of total $N = 100$ prediction slots in DETR decoder. Each box prediction is represented as a point with the coordinates of its center in the 1-by-1 square normalized by each image size. The points are color-coded so that green color corresponds to small boxes, red to large horizontal boxes and blue to large vertical boxes. We observe that each slot learns to specialize on certain areas and box sizes with several operating modes. We note that almost all slots have a mode of predicting large image-wide boxes that are common in COCO dataset.

Learning and Inference

- There is a maximum number of objects that can be detected (N=100).
- The predictions have to be associated with GT.
- Hungarian loss (a.k.a. linear sum assignment).
- The cost matrix for Hung. Loss is computed from Generalized IoU.
- The loss is then (almost) the same as in Faster RCNN.
- There is an “empty” class whose boxes must not overlap with GT boxes.

Vision Transformer

- An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale



$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}};\, \mathbf{x}_p^1\mathbf{E};\, \mathbf{x}_p^2\mathbf{E};\cdots;\, \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \qquad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C)\times D},\ \mathbf{E}_{pos} \in \mathbb{R}^{(N+1)\times D}$$

$$\mathbf{z}'_\ell = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \qquad \ell = 1\ldots L$$

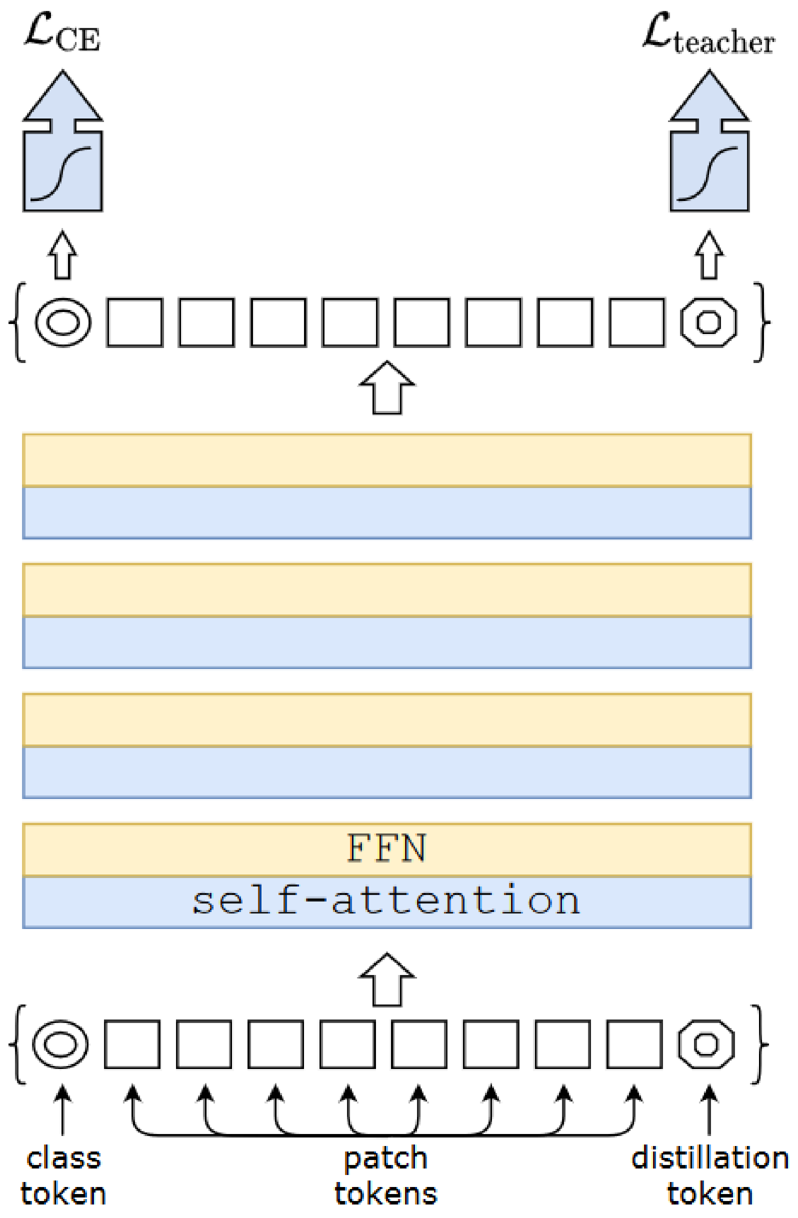
$$\mathbf{z}_\ell = \mathrm{MLP}(\mathrm{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, \qquad \ell = 1\ldots L$$

$$\mathbf{y} = \mathrm{LN}(\mathbf{z}_L^0)$$

- Paper from October 2020. ***ViT***
- Pure Transformer architecture without convolutions.
- An image is divided into equal parts, the parts are flattened and used as an input into a “bert-like” architecture with a **CLASS token**.
- Much less inductive bias than CNNs.
- More parameters than CNNs, but larger throughput of images.

Distillation in Transformers

- Training data-efficient image transformers & distillation through attention
- Paper from **December 2020**. ***DEiT***
- A distillation token is used to incorporate knowledge from a teacher.
- The teacher is assumed to be a strong classifier (i.e., a large CNN).
- Since Transformers have generally higher throughput of images it makes sense to distil the information from CNN to Transformer.



Distillation details

$$\mathcal{L}_{\text{global}}^{\text{hardDistill}} = \frac{1}{2}\mathcal{L}_{\text{CE}}(\psi(Z_s), y) + \frac{1}{2}\mathcal{L}_{\text{CE}}(\psi(Z_s), y_{\text{t}}).$$

- Two variants – **soft** and **hard** labels from the teacher.

$$\mathcal{L}_{\text{global}} = (1-\lambda)\mathcal{L}_{\text{CE}}(\psi(Z_{\text{s}}), y) + \lambda\tau^2\text{KL}(\psi(Z_{\text{s}}/\tau), \psi(Z_{\text{t}}/\tau)).$$

- Interesting discoveries:
  - Class and Distillation tokens converge to different vectors.
  - At first (input) layer they are very dissimilar (cos=0.06) at the last level they are very similar (cos=0.93).
  - Using two randomly initialized class tokens converges to quasi-identical vectors (cos = 0.999).
  - Hard distillation is simpler and performs better.
  - Very good for finetuning to higher resolution.

Self-supervision (aka distillation without label)

- Emerging Properties in Self-Supervised Vision Transformers
- Paper from April 2021. ***DINO.***
- Self-supervised ViT features contain **explicit information** about the **semantic segmentation** of an image, which does not emerge as clearly with supervised ViTs, nor with convnets.
- These features are also excellent k-NN classifiers.

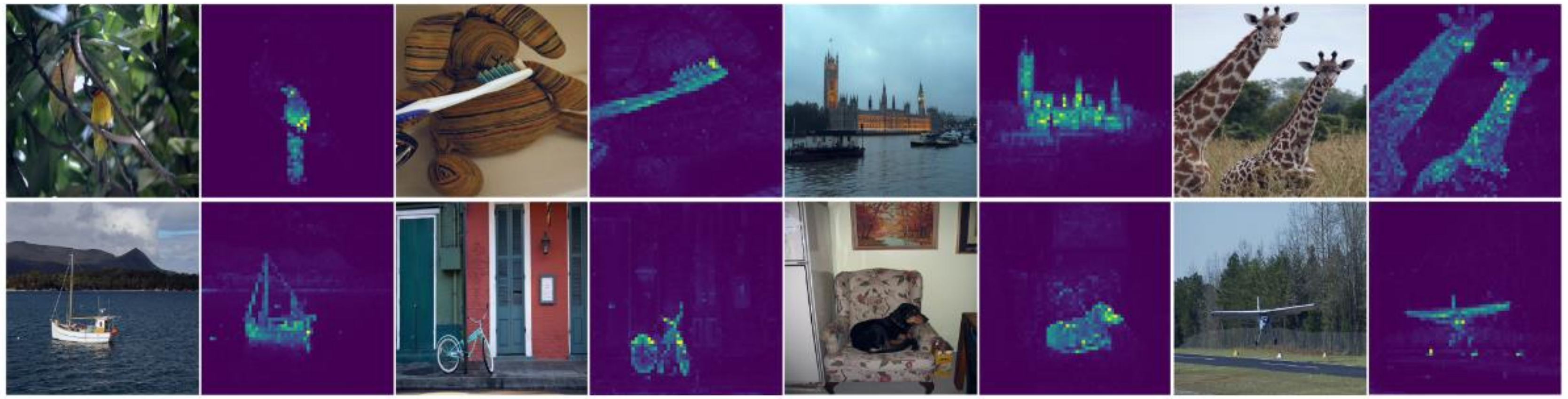
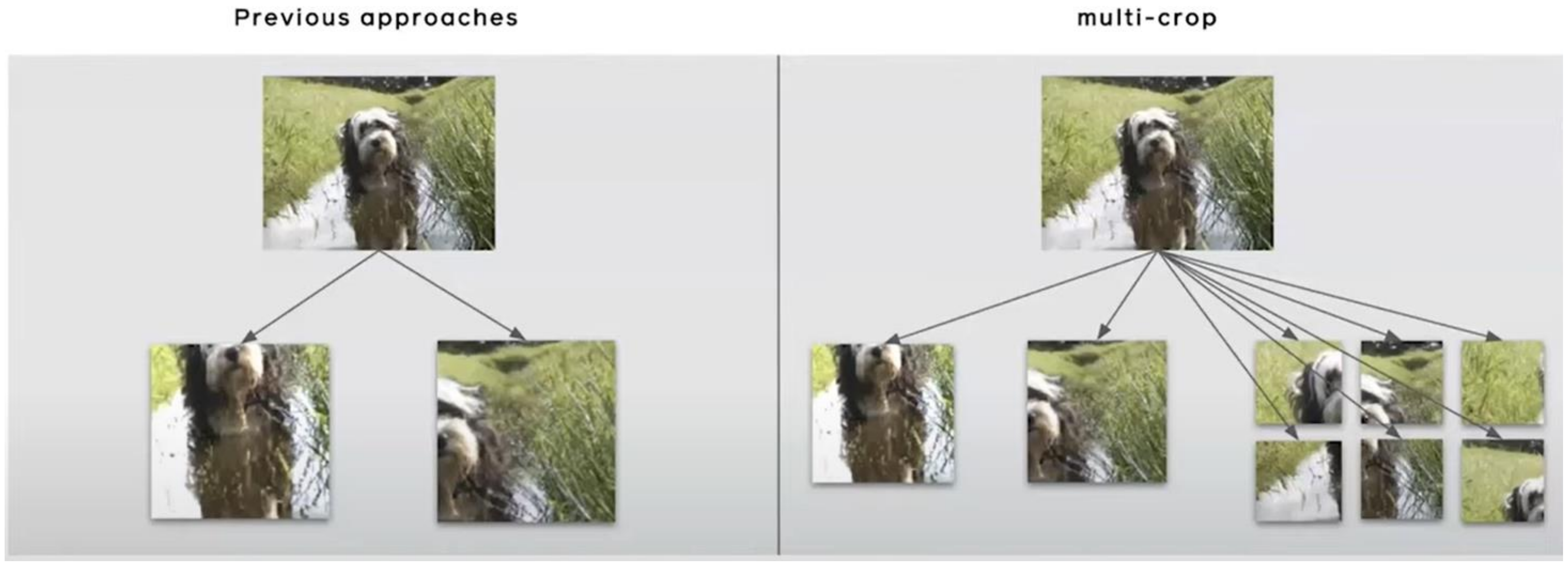


Figure 1: **Self-attention from a Vision Transformer with 8×8 patches trained with no supervision.** We look at the self-attention of the `[CLS]` token on the heads of the last layer. This token is not attached to any label nor supervision. These maps show that the model automatically learns class-specific features leading to unsupervised object segmentations.

Training

- Two global views and multiple local views are generated using MultiCrop augmentations.



Training

- Two global views and multiple local views are generated using MultiCrop augmentations.
- Global views are passed to the Teacher.
- All views (global+local) are passed to the Student.
- Cross-entropy between Teacher and Student is computed as loss.

$$\min_{\theta_s} \sum_{x \in \{x_1^g, x_2^g\}} \sum_{\substack{x' \in V \\ x' \neq x}} H(P_t(x), P_s(x')).$$

Training 2

$$\min_{\theta_s} \sum_{x \in \{x_1^g, x_2^g\}} \sum_{\substack{x' \in V \\ x' \neq x}} H(P_t(x), P_s(x')).$$

- The Student ‘s parameters are updated by gradient descent.
- The Teacher’s parameters are updated as exponential moving average.
- $\theta_t \leftarrow \lambda\theta_t + (1 - \lambda)\theta_s$
- The softmax of the Student and Teacher is sharpened.
- The Teacher’s softmax is also centered.

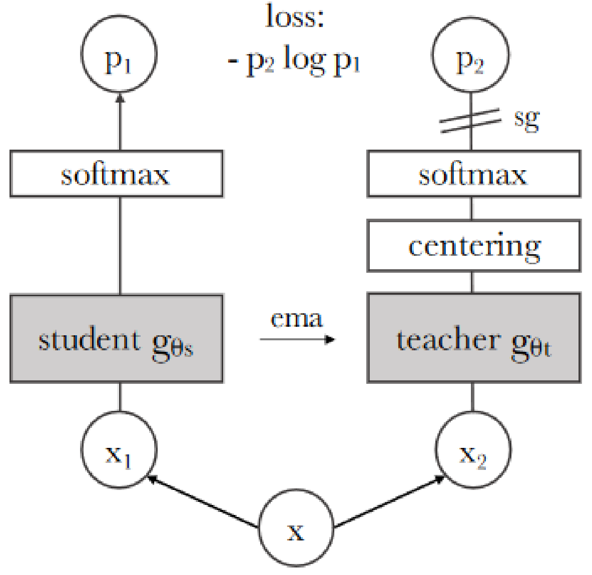


Figure 2: **Self-distillation with no labels.** We illustrate DINO in the case of one single pair of views (x_1, x_2) for simplicity. The model passes two different random transformations of an input image to the student and teacher networks. Both networks have the same architecture but different parameters. The output of the teacher network is centered with a mean computed over the batch. Each networks outputs a K dimensional feature that is normalized with a temperature softmax over the feature dimension. Their similarity is then measured with a cross-entropy loss. We apply a stop-gradient (sg) operator on the teacher to propagate gradients only through the student. The teacher parameters are updated with an exponential moving average (ema) of the student parameters.

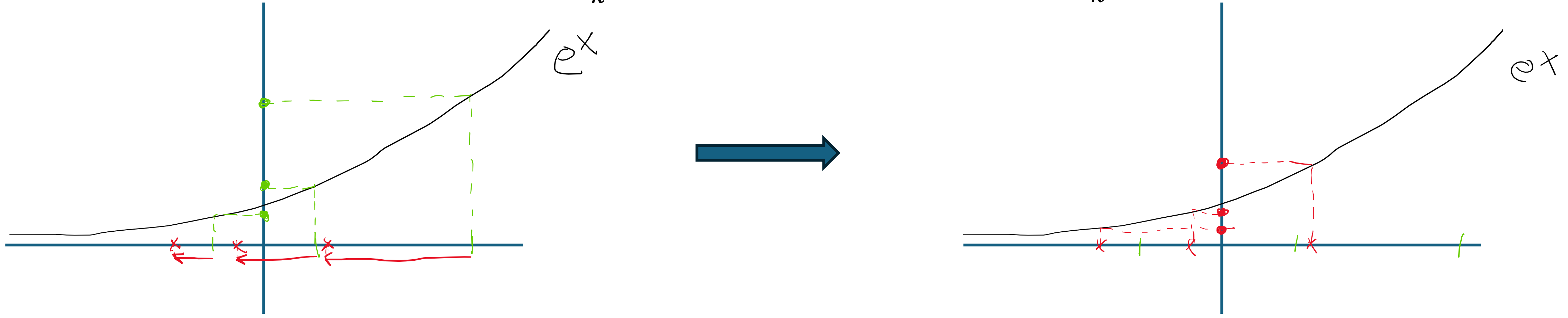
Algorithm 1 DINO PyTorch pseudocode w/o multi-crop.

```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

Centering

$$f(x_i) = \frac{e^{x_i}}{\sum_k e^{x_k}}$$

$$f_c(x_i) = \frac{e^{x_i - c_i}}{\sum_k e^{x_k - c_k}}$$



$$f_c(x_i) = \frac{e^{x_i-c_i}}{\sum_k e^{x_k-c_k}} = \frac{\frac{e^{x_i}}{e^{c_i}}}{\frac{e^{x_i}}{e^{c_i}} + \sum_{k-\{i\}} \frac{e^{x_k}}{e^{c_k}}} = \frac{e^{x_i}}{e^{x_i} + e^{c_i}\sum_{k-} \frac{e^{x_k}}{e^{c_k}}}$$

$$f(x_i) \lesseqqgtr f_c(x_i)$$

$$\frac{e^{x_i}}{\sum_k e^{x_k}} \lesseqqgtr \frac{e^{x_i}}{e^{x_i} + e^{c_i}\sum_{k-} \frac{e^{x_k}}{e^{c_k}}}$$

$$\frac{e^{x_i}}{e^{x_i} + \sum_{k-} e^{x_k}} \lesseqqgtr \frac{e^{x_i}}{e^{x_i} + e^{c_i}\sum_{k-} \frac{e^{x_k}}{e^{c_k}}}$$

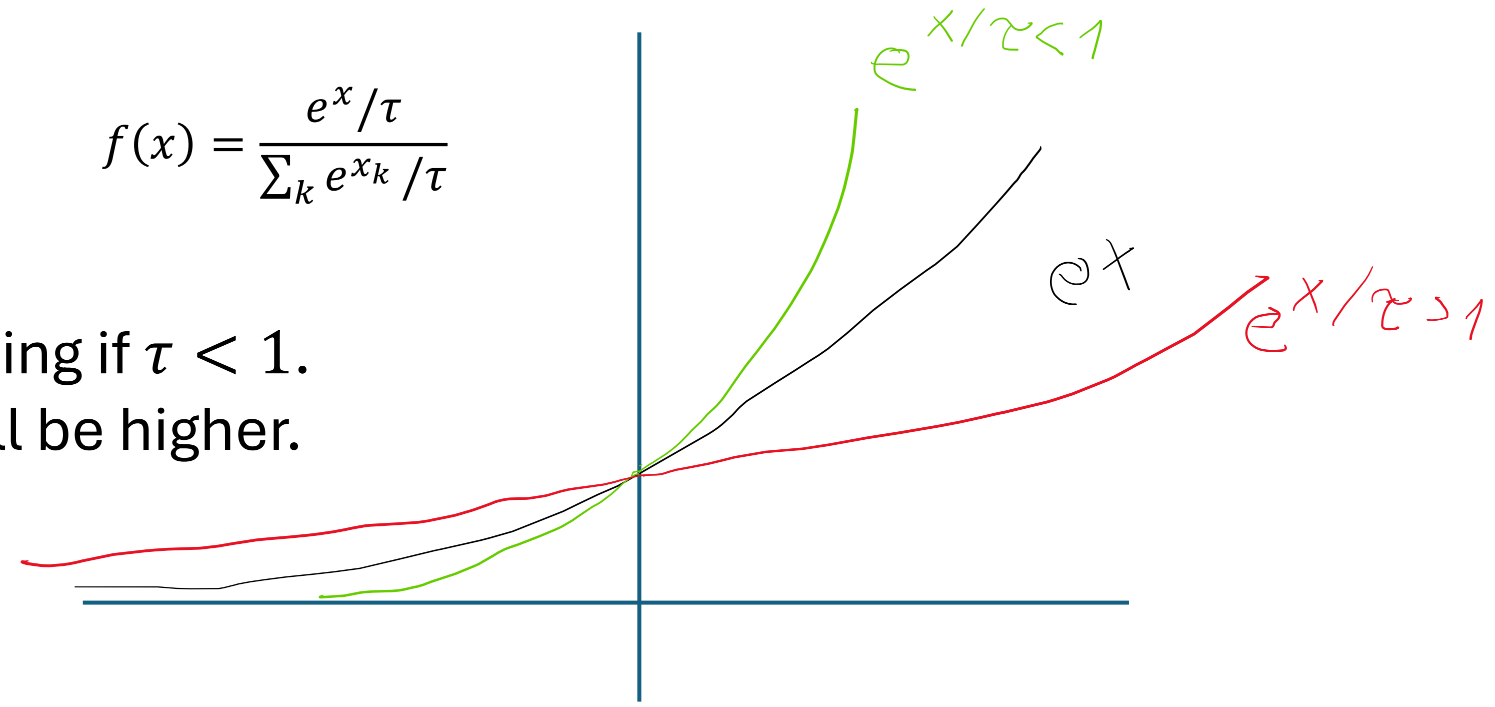
$$\sum_{k-} e^{x_k} \gtreqqless e^{c_i} \sum_{k-} \frac{e^{x_k}}{e^{c_k}}$$

$$\sum_{k-} \frac{e^{x_k}}{e^{c_i}} \gtreqqless \sum_{k-} \frac{e^{x_k}}{e^{c_k}}$$

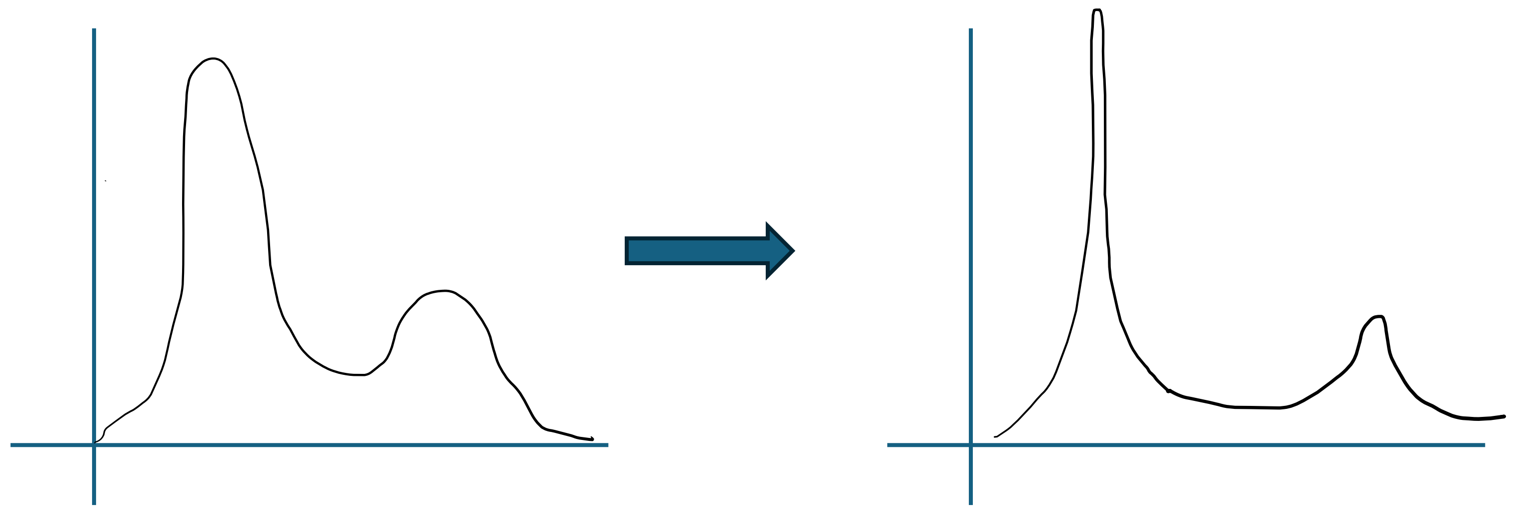
- Centering enforces a uniform distribution.
- The distribution is along all the inputs (i.e., images)
- This means we want our classifier to make different decisions given different inputs.
- Ideal for uniformly distributed data.
- The centering is applied to the teacher softmax (i.e., target for the student).

Sharpening

$$f(x) = \frac{e^{x}/\tau}{\sum_k e^{x_k}/\tau}$$



- We can only call it sharpening if $\tau < 1$.
- After softmax the peaks will be higher.
- Both teacher and student are sharpened, but with different constants.



Training details

- ViT from scratch.
- The same architecture for teacher and student.
- Learning rate
  - 10 epochs linear warm-up
  - after decay with cosine schedule
- Weight decay has also cosine schedule from 0.04 to 0.4
- Teacher – Polyak-Ruppert averaging with exponential decay.
- $\theta_t \leftarrow \lambda\theta_t + (1 - \lambda)\theta_s$, λ follows cosine schedule from 0.996 to 1.
- Teacher τ has linear warm-up from 0.04 to 0.07 in 30 epochs.
- Student τ is set to 0.1.

DINO features

- A Head is put after the feature maps (ResNet) or CLS token.
- The Head has three layers with hidden dimension 2048 followed by l2 normalization.
- Then a weight-normalized FC layer with *K* dimensions.
- Different heads were tested.

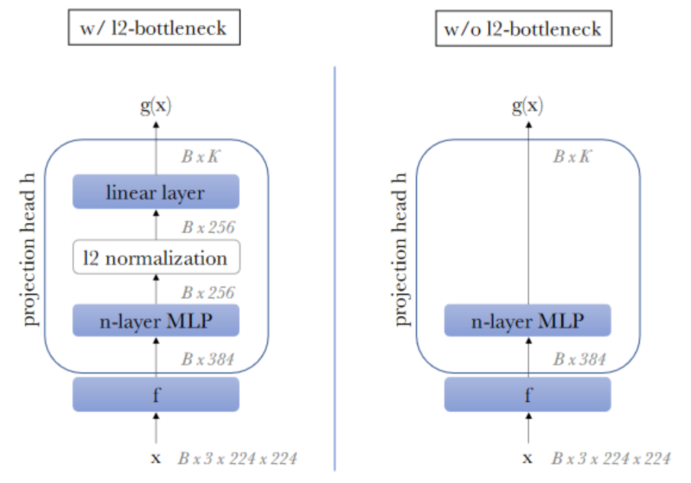
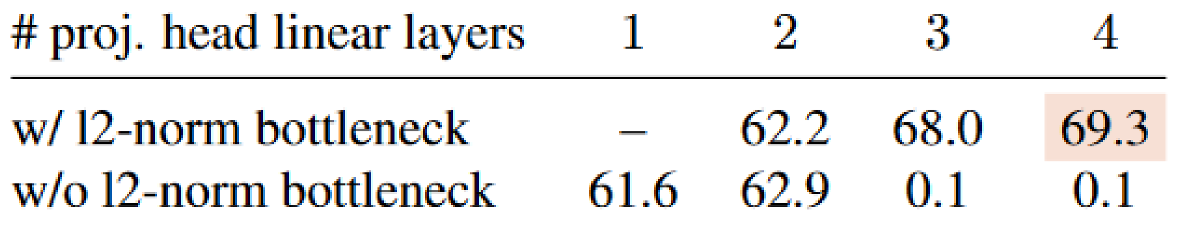
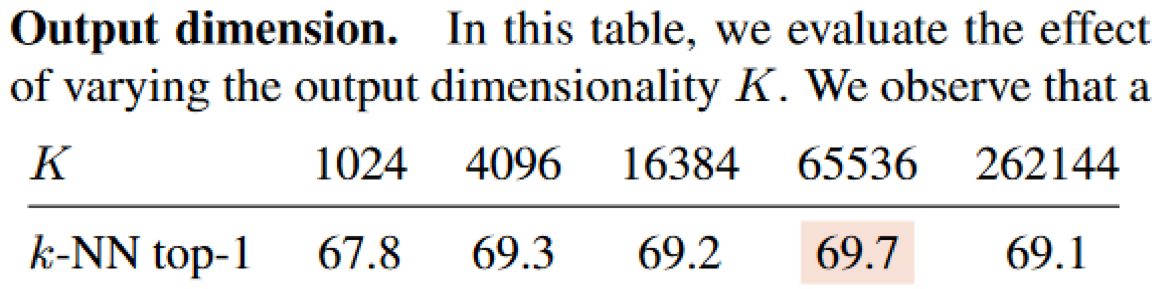


Figure 9: **Projection head design w/ or w/o l2-norm bottleneck.**



# proj. head linear layers	1	2	3	4
w/ l2-norm bottleneck	–	62.2	68.0	69.3
w/o l2-norm bottleneck	61.6	62.9	0.1	0.1



Output dimension. In this table, we evaluate the effect of varying the output dimensionality K. We observe that a

K	1024	4096	16384	65536	262144
k-NN top-1	67.8	69.3	69.2	69.7	69.1

Evaluation protocol

- Linear
  - A linear classifier with fixed backbone is trained on a val set.
  - Test set is classified using this classifier.
  - Drawbacks:
    - Needs time to learn.
    - Each task needs new learning.
    - Unstable learning – each task needs different parameters.
- k-NN
  - Each image in train set is processed by DINO.
  - Test sample is classified using 20-NN classification.
  - Stable through the tasks.

Method	Arch.	Param.	im/s	Linear	*k*-NN
Supervised	RN50	23	1237	79.3	79.3
SCLR [12]	RN50	23	1237	69.1	60.7
MoCov2 [15]	RN50	23	1237	71.1	61.9
InfoMin [67]	RN50	23	1237	73.0	65.3
BarlowT [81]	RN50	23	1237	73.2	66.0
OBoW [27]	RN50	23	1237	73.8	61.9
BYOL [30]	RN50	23	1237	74.4	64.8
DCv2 [10]	RN50	23	1237	75.2	67.1
SwAV [10]	RN50	23	1237	**75.3**	65.7
DINO	RN50	23	1237	**75.3**	**67.5**
Supervised	ViT-S	21	1007	79.8	79.8
BYOL* [30]	ViT-S	21	1007	71.4	66.6
MoCov2* [15]	ViT-S	21	1007	72.7	64.4
SwAV* [10]	ViT-S	21	1007	73.5	66.3
DINO	ViT-S	21	1007	**77.0**	**74.5**
Comparison across architectures					
SCLR [12]	RN50w4	375	117	76.8	69.3
SwAV [10]	RN50w2	93	384	77.3	67.3
BYOL [30]	RN50w2	93	384	77.4	–
DINO	ViT-B/16	85	312	78.2	76.1
SwAV [10]	RN50w5	586	76	78.5	67.1
BYOL [30]	RN50w4	375	117	78.6	–
BYOL [30]	RN200w2	250	123	79.6	73.9
DINO	ViT-S/8	21	180	79.7	**78.3**
SCLRv2 [13]	RN152w3+SK	794	46	79.8	73.1
DINO	ViT-B/8	85	63	**80.1**	77.4

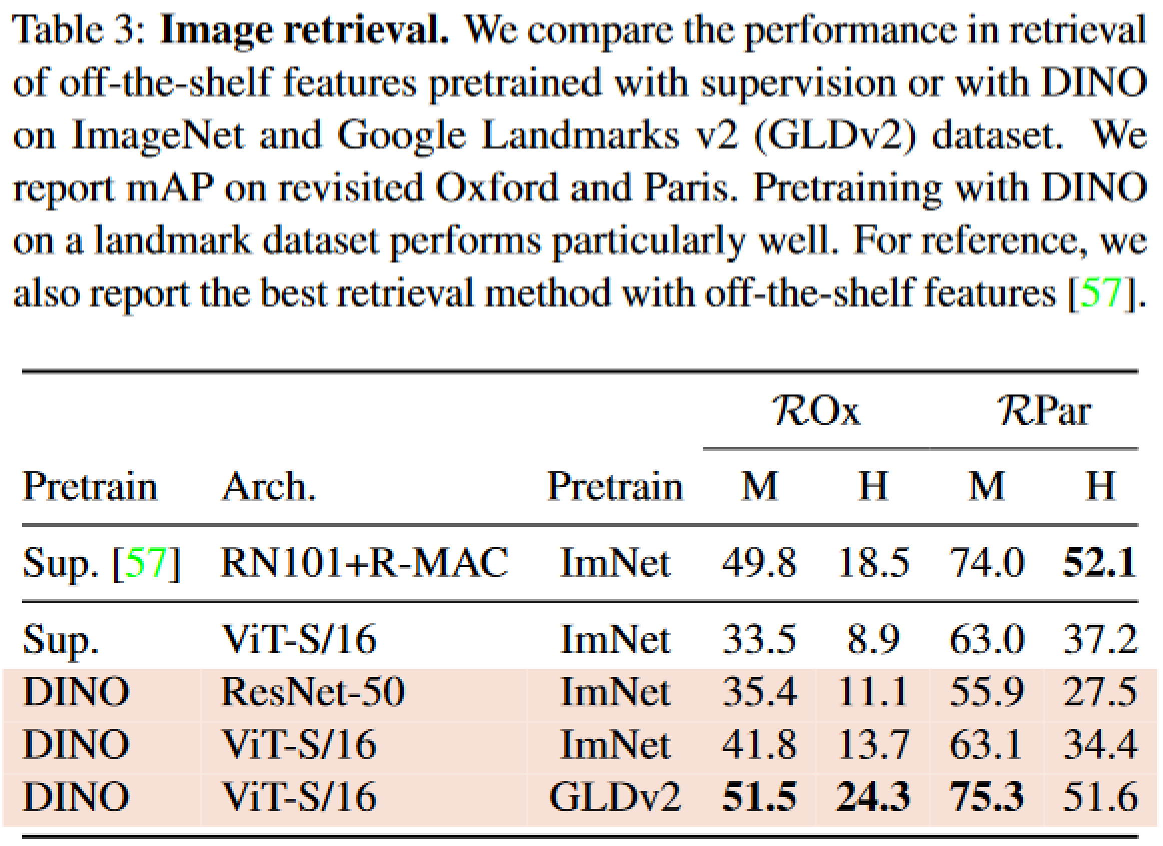


Table 3: **Image retrieval.** We compare the performance in retrieval of off-the-shelf features pretrained with supervision or with DINO on ImageNet and Google Landmarks v2 (GLDv2) dataset. We report mAP on revisited Oxford and Paris. Pretraining with DINO on a landmark dataset performs particularly well. For reference, we also report the best retrieval method with off-the-shelf features [57].

			$\mathcal{R}$Ox		$\mathcal{R}$Par	
Pretrain	Arch.	Pretrain	M	H	M	H
Sup. [57]	RN101+R-MAC	ImNet	49.8	18.5	74.0	**52.1**
Sup.	ViT-S/16	ImNet	33.5	8.9	63.0	37.2
DINO	ResNet-50	ImNet	35.4	11.1	55.9	27.5
DINO	ViT-S/16	ImNet	41.8	13.7	63.1	34.4
DINO	ViT-S/16	GLDv2	**51.5**	**24.3**	**75.3**	51.6

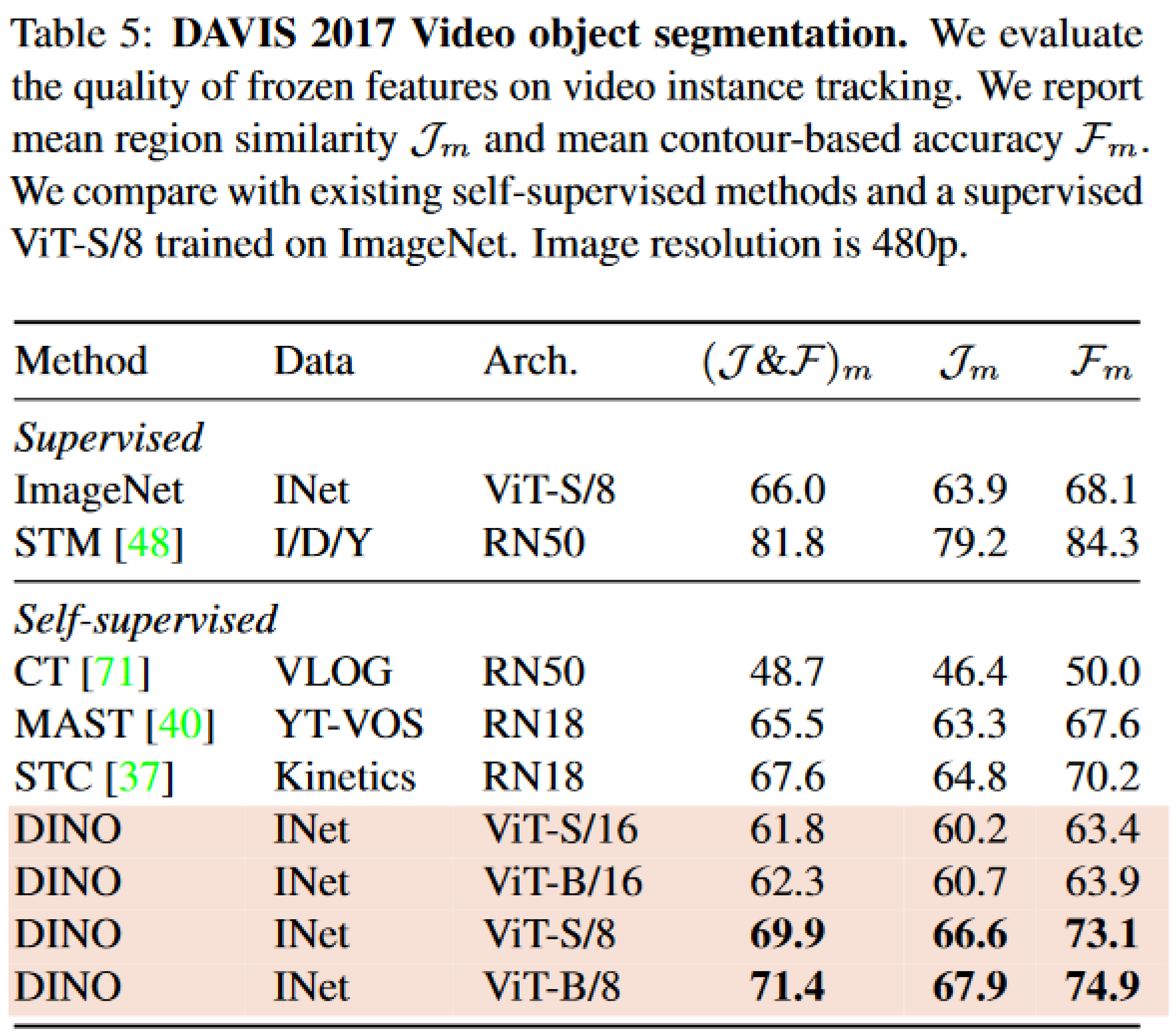


Table 5: **DAVIS 2017 Video object segmentation.** We evaluate the quality of frozen features on video instance tracking. We report mean region similarity $\mathcal{J}_m$ and mean contour-based accuracy $\mathcal{F}_m$. We compare with existing self-supervised methods and a supervised ViT-S/8 trained on ImageNet. Image resolution is 480p.

Method	Data	Arch.	$(\mathcal{J}\&\mathcal{F})_m$	$\mathcal{J}_m$	$\mathcal{F}_m$
Supervised					
ImageNet	INet	ViT-S/8	66.0	63.9	68.1
STM [48]	I/D/Y	RN50	81.8	79.2	84.3
Self-supervised					
CT [71]	VLOG	RN50	48.7	46.4	50.0
MAST [40]	YT-VOS	RN18	65.5	63.3	67.6
STC [37]	Kinetics	RN18	67.6	64.8	70.2
DINO	INet	ViT-S/16	61.8	60.2	63.4
DINO	INet	ViT-B/16	62.3	60.7	63.9
DINO	INet	ViT-S/8	**69.9**	**66.6**	**73.1**
DINO	INet	ViT-B/8	**71.4**	**67.9**	**74.9**

https://davischallenge.org/images/DAVIS-2017-TrainVal.mp4

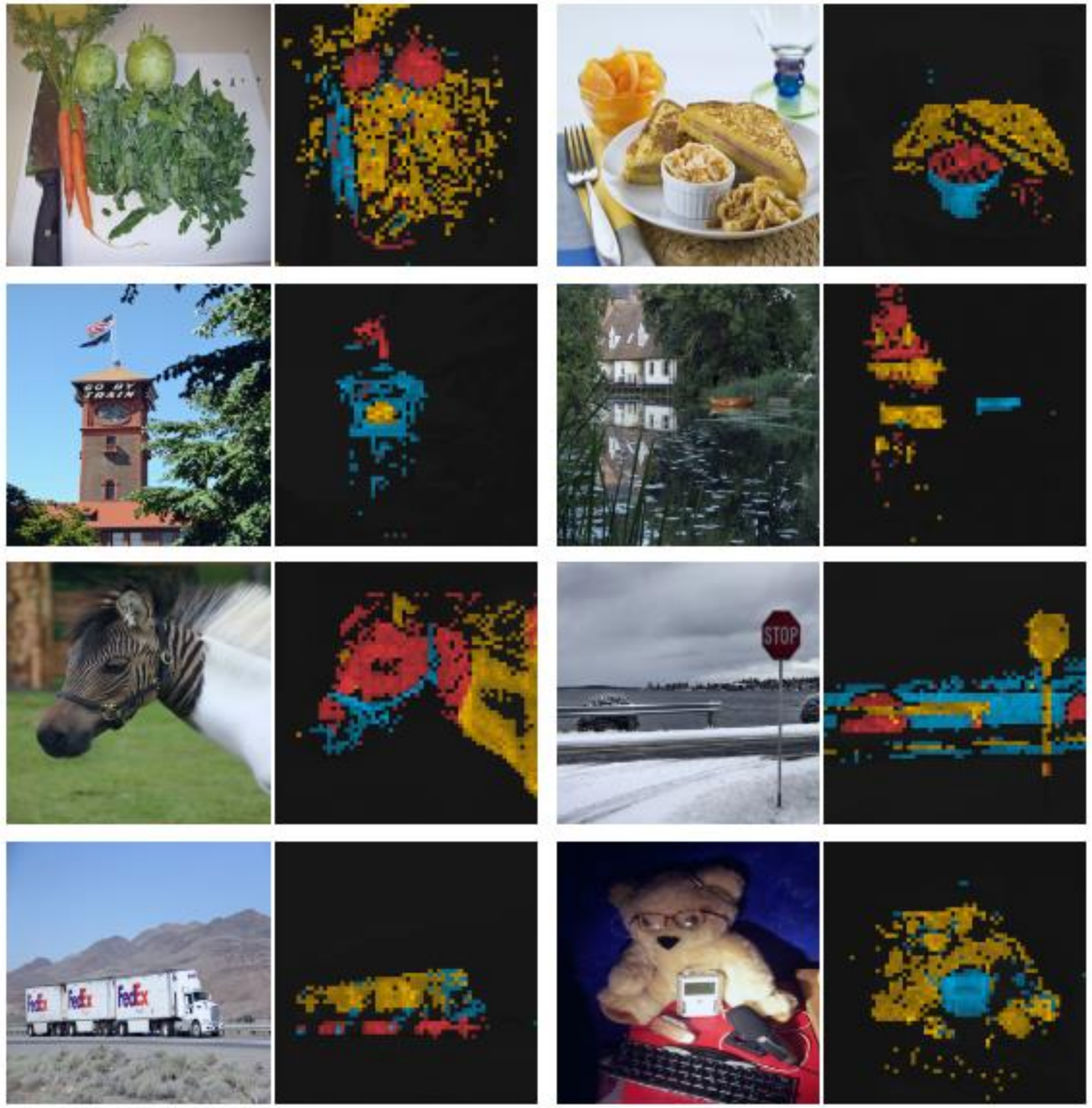
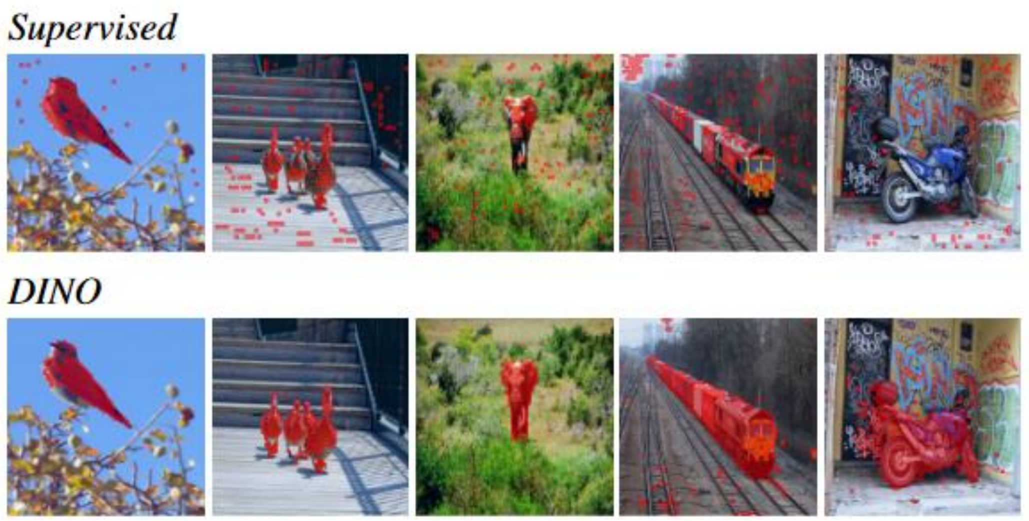


Figure 3: **Attention maps from multiple heads.** We consider the heads from the last layer of a ViT-S/8 trained with DINO and display the self-attention for `[CLS]` token query. Different heads, materialized by different colors, focus on different locations that represents different objects or parts (more examples in Appendix).

Ablation



	Random	Supervised	DINO
ViT-S/16	22.0	27.3	45.9
ViT-S/8	21.8	23.7	44.7

Figure 4: **Segmentations from supervised versus DINO.** We visualize masks obtained by thresholding the self-attention maps to keep 60% of the mass. On top, we show the resulting masks for a ViT-S/8 trained with supervision and DINO. We show the best head for both models. The table at the bottom compares the Jaccard similarity between the ground truth and these masks on the validation images of PASCAL VOC12 dataset.

Table 7: **Important component for self-supervised ViT pre-training.** Models are trained for 300 epochs with ViT-S/16. We study the different components that matter for the k-NN and linear ("Lin.") evaluations. For the different variants, we highlight the differences from the default DINO setting. The best combination is the momentum encoder with the multicrop augmentation and the cross-entropy loss. We also report results with BYOL [30], MoCo-v2 [15] and SwAV [10].

	Method	Mom.	SK	MC	Loss	Pred.	k-NN	Lin.
1	DINO	✓	✗	✓	CE	✗	72.8	76.1
2		✗	✗	✓	CE	✗	0.1	0.1
3		✓	✓	✓	CE	✗	72.2	76.0
4		✓	✗	✗	CE	✗	67.9	72.5
5		✓	✗	✓	MSE	✗	52.6	62.4
6		✓	✗	✓	CE	✓	71.8	75.6
7	BYOL	✓	✗	✗	MSE	✓	66.6	71.4
8	MoCov2	✓	✗	✗	INCE	✗	62.0	71.6
9	SwAV	✗	✓	✓	CE	✗	64.7	71.8

SK: Sinkhorn-Knopp, MC: Multi-Crop, Pred.: Predictor
CE: Cross-Entropy, MSE: Mean Square Error, INCE: InfoNCE

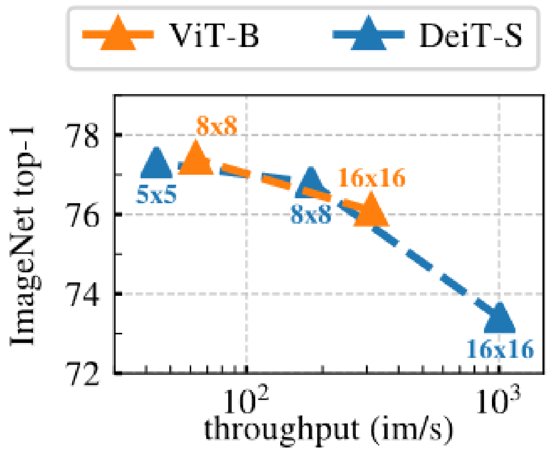


Figure 5: **Effect of Patch Size.** k-NN evaluation as a function of the throughputs for different input patch sizes with ViT-B and ViT-S. Models are trained for 300 epochs.

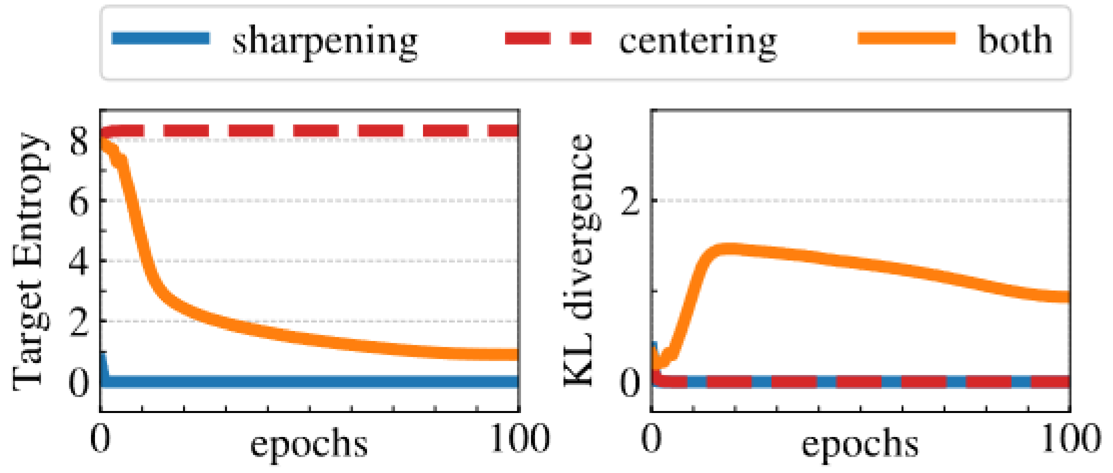
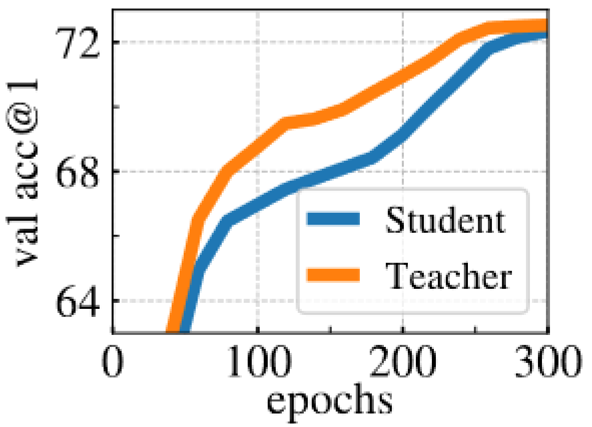


Figure 7: **Collapse study. (left)**: evolution of the teacher's target entropy along training epochs; **(right)**: evolution of KL divergence between teacher and student outputs.

$$H(P_t, P_s) = h(P_t) + D_{KL}(P_t|P_s).$$



Teacher	Top-1
Student copy	0.1
Previous iter	0.1
Previous epoch	66.6
Momentum	72.8

Figure 6: Top-1 accuracy on ImageNet validation with k-NN classifier. **(left)** Comparison between the performance of the momentum teacher and the student during training. **(right)** Comparison between different types of teacher network. The momentum encoder leads to the best performance but is not the only viable option.

Feel good images

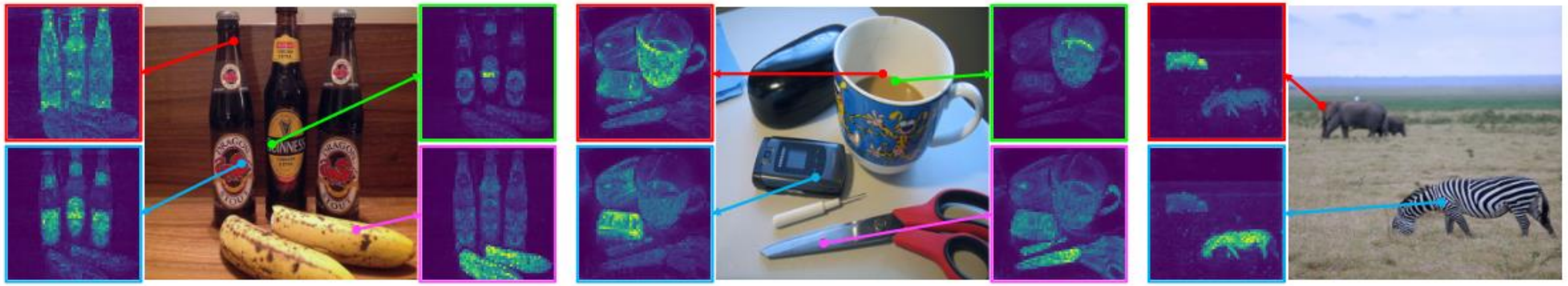
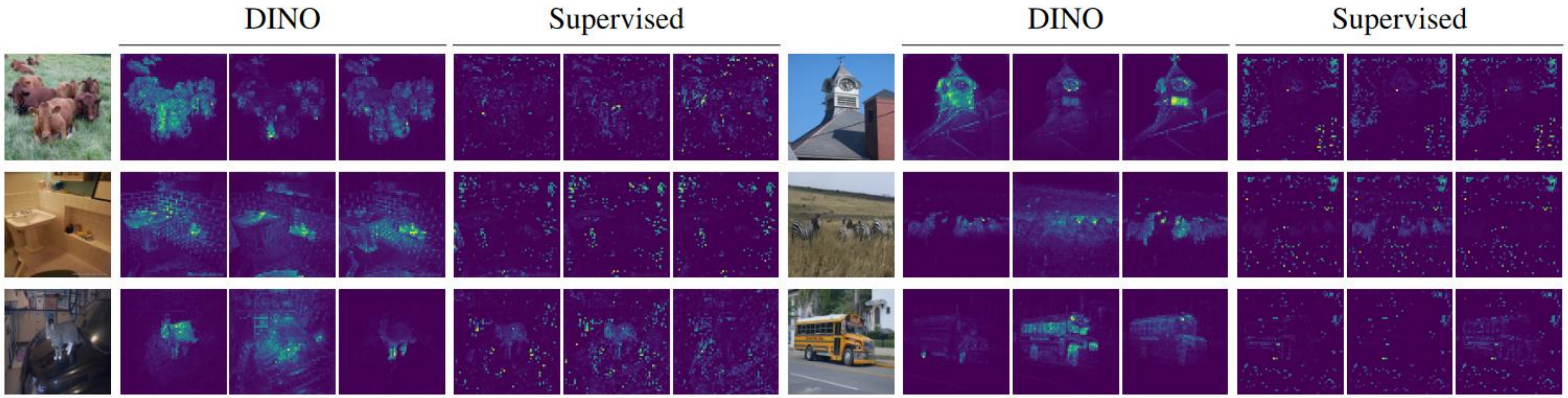


Figure 8: **Self-attention for a set of reference points.** We visualize the self-attention module from the last block of a ViT-S/8 trained with DINO. The network is able to separate objects, though it has been trained with no supervision at all.



Sign Language

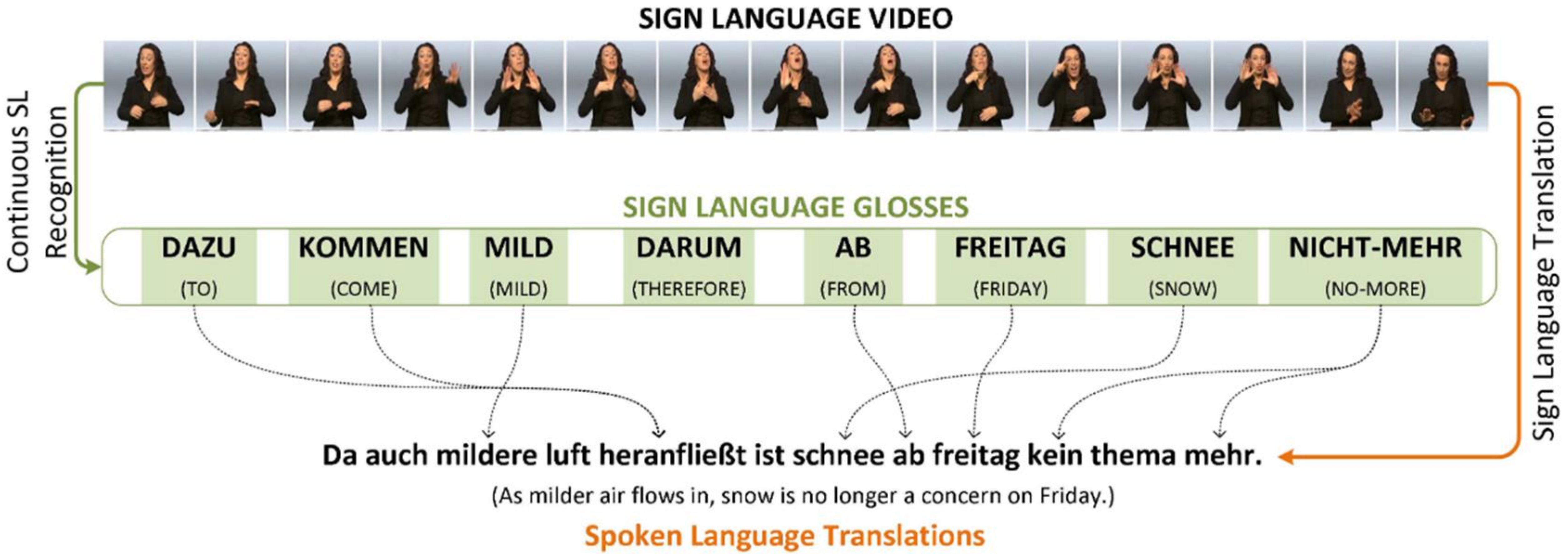
The Deep Learning Perspective

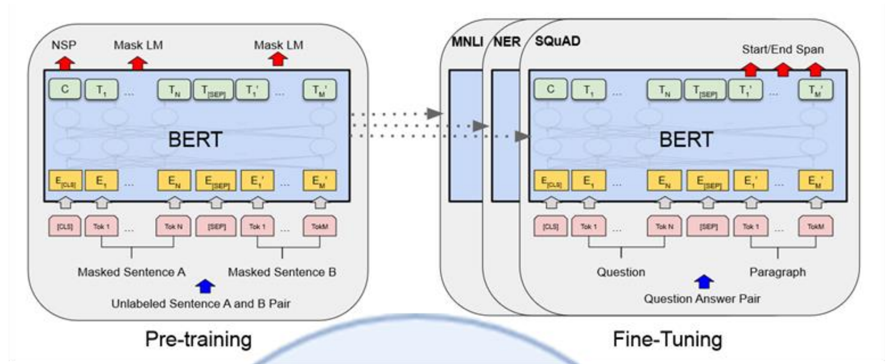
Sign Language (according to ChatGPT)

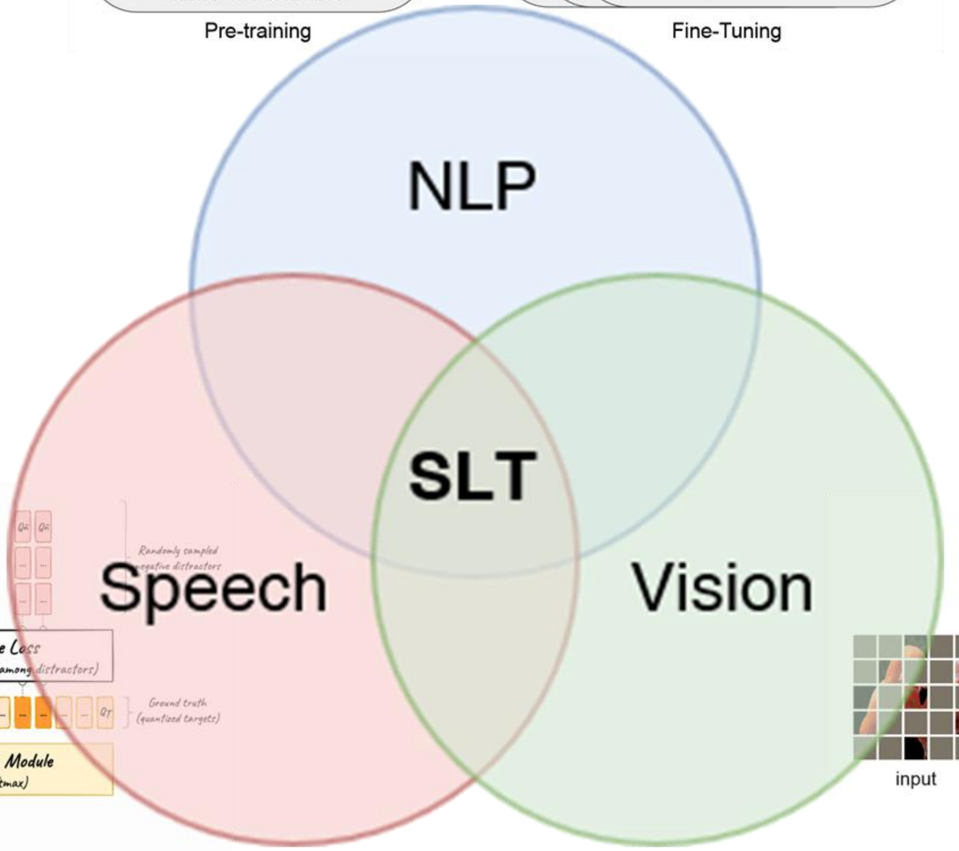
- Rich and complex form of communication.

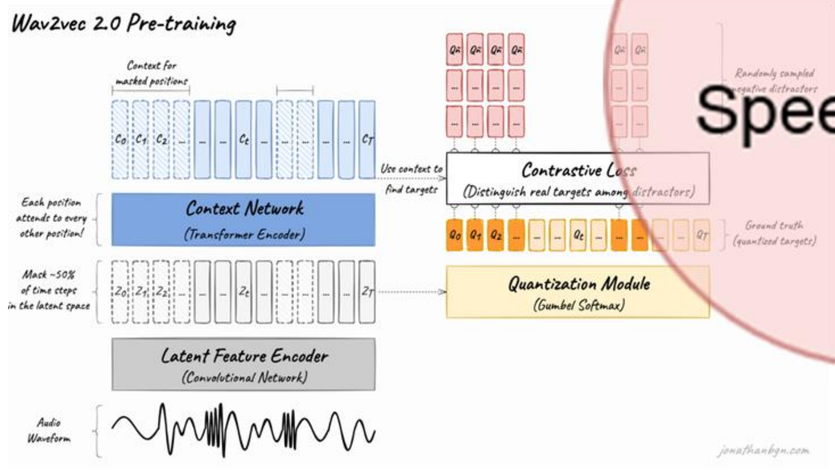
1. **Manual Components**: These are the elements of sign language that involve the hands and arms.
   1. **Handshapes:** Different configurations of the fingers and hands represent specific letters, words, or concepts. Each sign is formed by combining these handshapes in various ways.
   2. **Movements:** The motion and direction of the hands and arms play a crucial role in conveying meaning. Movements can indicate actions, locations, or transitions between ideas.
   3. **Locations:** The space around the signer's body is divided into specific locations, each representing different grammatical or lexical elements. These locations are used to indicate subjects, objects, or points of reference.
   4. **Palm orientation:** The way the palm faces can alter the meaning of a sign. Whether the palm is facing up, down, to the side, or inward can change the interpretation of the gesture.
2. **Non-manual Components:** These aspects of sign language involve facial expressions, body posture, and other visual cues that accompany manual signs.
   1. **Facial expressions**: Expressive facial movements are crucial in sign language, as they convey grammatical information, emotions, and emphasis. Different facial expressions can change the meaning or intensity of a sign, just as intonation does in spoken language.
   2. **Body language**: Body posture, stance, and movement provide additional context and meaning to signs. They can indicate the speaker's attitude, intention, or the relationship between different elements in a sentence.
   3. **Eye gaze**: Eye contact and direction play a significant role in sign language conversations. They signal turn-taking, indicate who is the subject or object of a sentence, and establish rapport between signers.

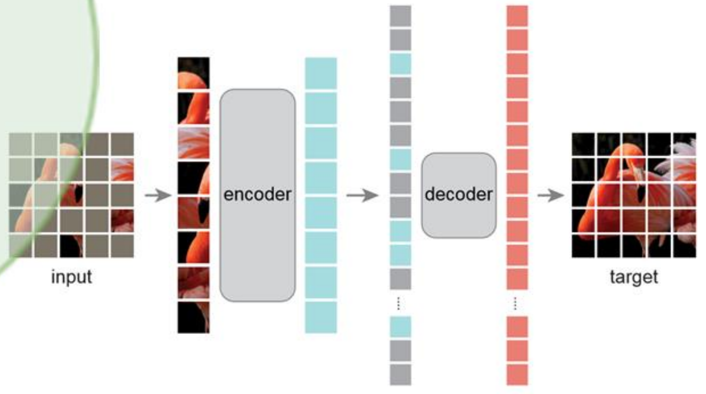
Sign Language Translation


SIGN LANGUAGE VIDEO
Continuous SL Recognition
Sign Language Translation
SIGN LANGUAGE GLOSSES
DAZU (TO)
KOMMEN (COME)
MILD (MILD)
DARUM (THEREFORE)
AB (FROM)
FREITAG (FRIDAY)
SCHNEE (SNOW)
NICHT-MEHR (NO-MORE)
Da auch mildere luft heranfließt ist schnee ab freitag kein thema mehr.
(As milder air flows in, snow is no longer a concern on Friday.)
Spoken Language Translations


NSP
Mask LM
Mask LM
BERT
Masked Sentence A
Masked Sentence B
Unlabeled Sentence A and B Pair
Pre-training
MNLI
NER
SQuAD
Start/End Span
BERT
Question
Paragraph
Question Answer Pair
Fine-Tuning

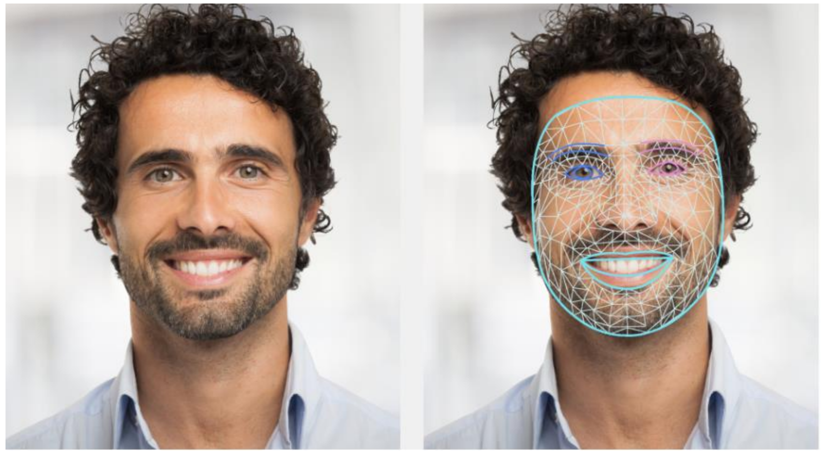
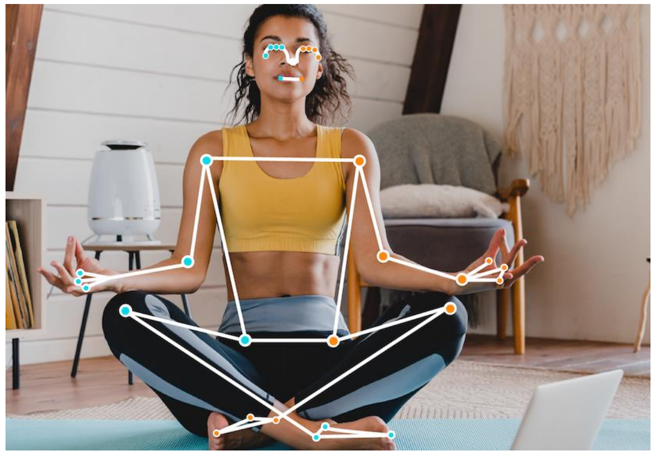
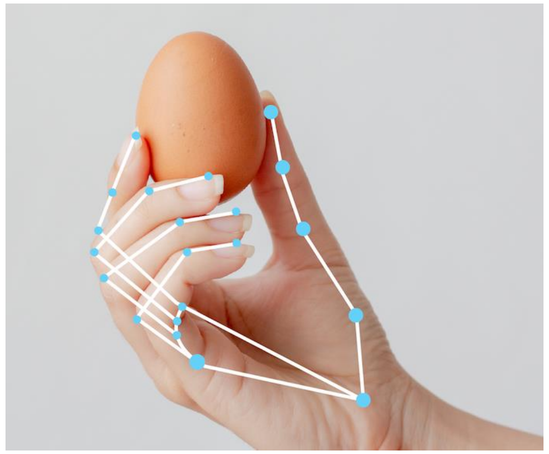

NLP
Speech
Vision
SLT


Wav2vec 2.0 Pre-training
Context for masked positions
Each position attends to every other position!
Context Network
(Transformer Encoder)
Mask ~50% of time steps in the latent space
Latent Feature Encoder
(Convolutional Network)
Audio Waveform
Use context to find targets
Contrastive Loss
(Distinguish real targets among distractors)
Randomly sampled negative distractors
Ground truth (quantized targets)
Quantization Module
(Gumbel Softmax)
jonathanbgn.com


input
encoder
decoder
target

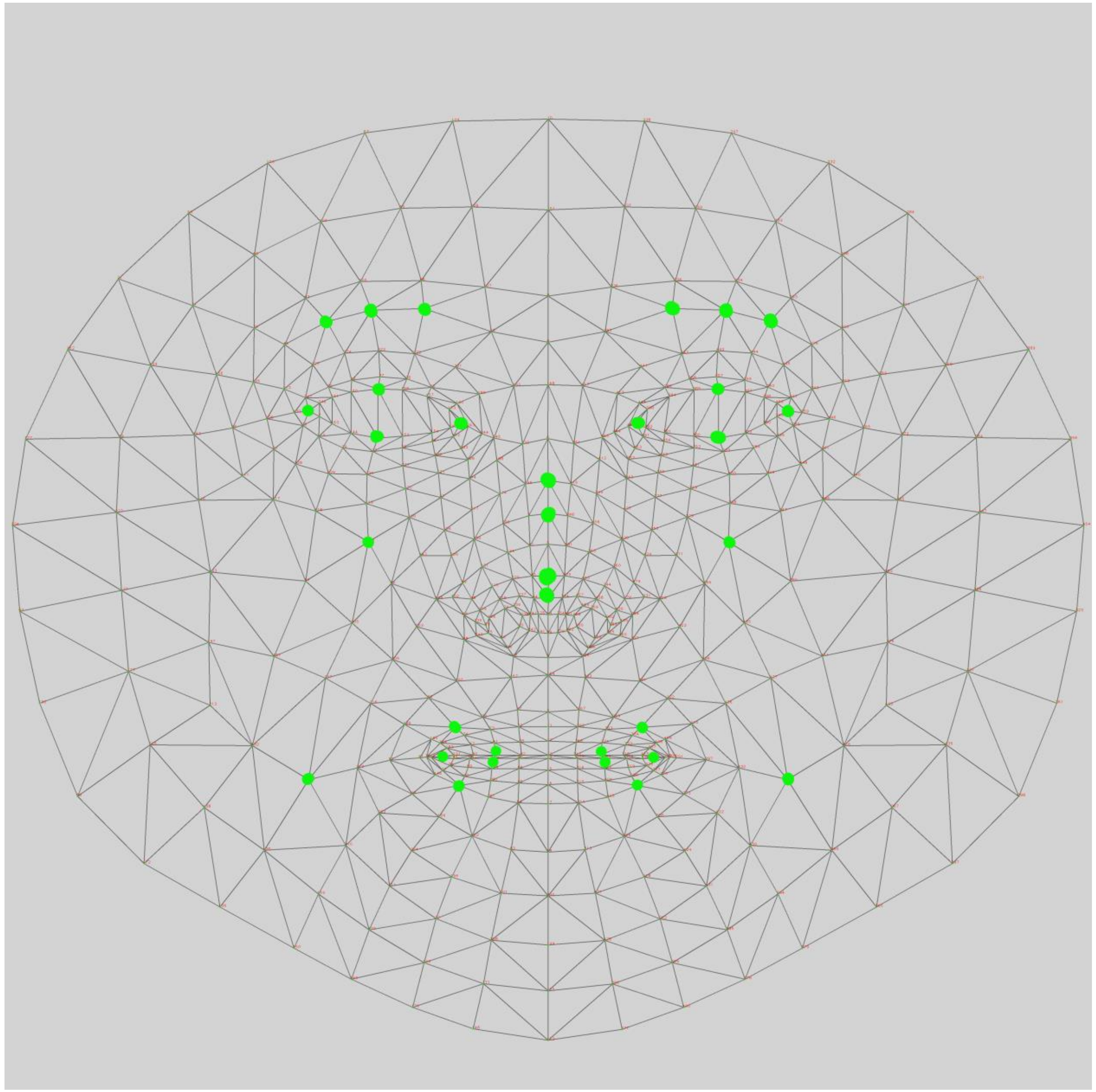
Pose Estimation

- Detection of Landmarks (e.g., joints) no Face, and Body
  - We focus on Face Landmark Detection, Pose Detection, and Hand Landmark Detection.

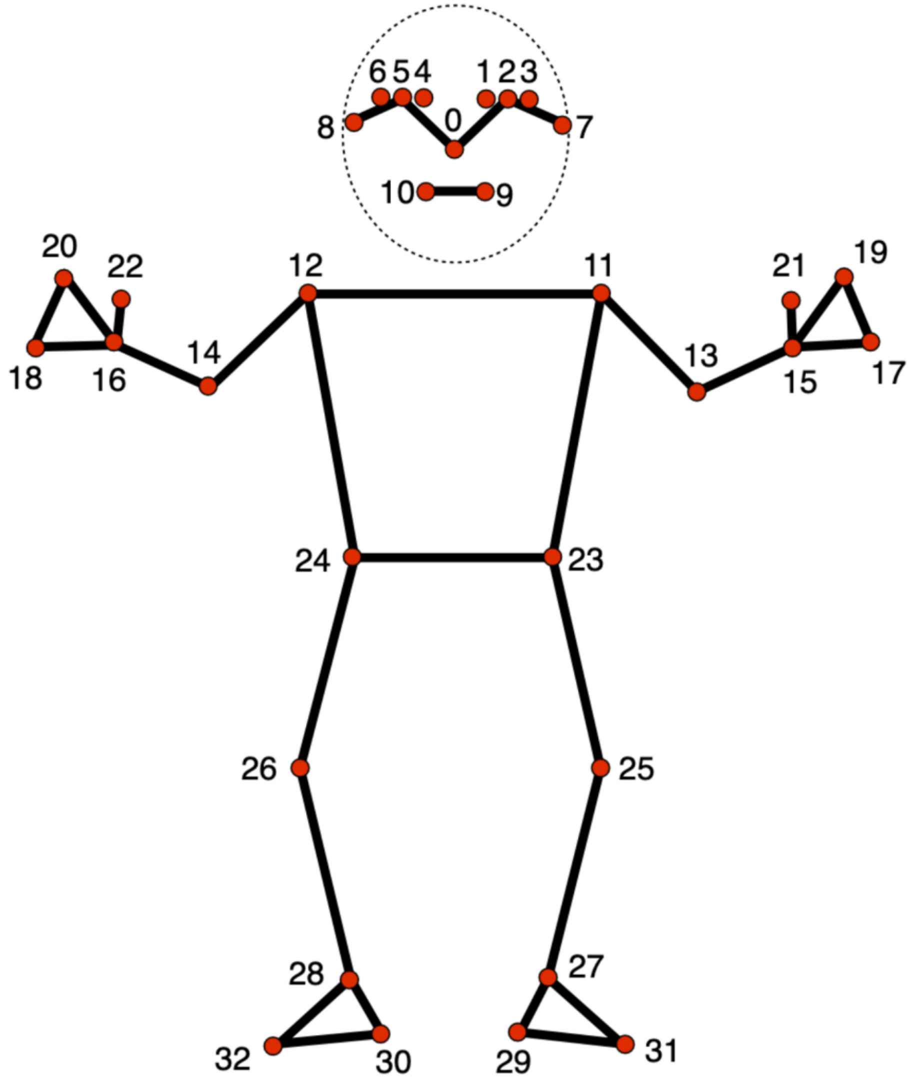




Pose Definition - Face

- The facial landmarks are very dense.
- Many are linearly dependent.
- Hand-picked important landmarks.
- We want to represent:
    - Eyebrows
    - Eyes
    - Mouth
    - Nose (for reference)

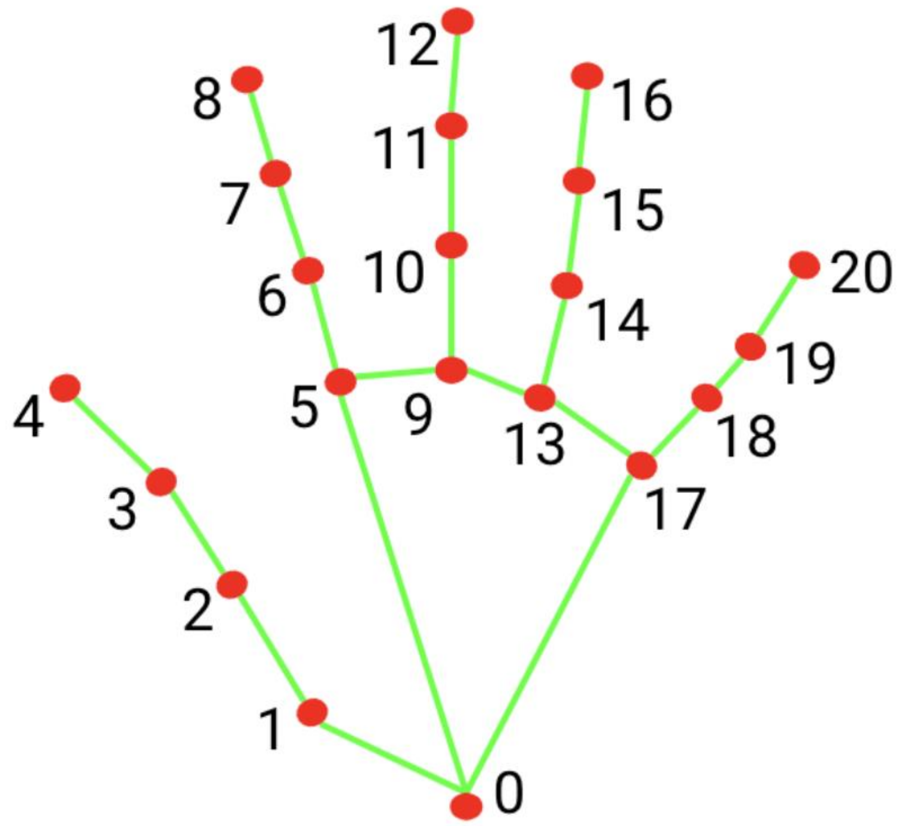


Pose Definition - Body



0 - nose	19 - left index
1 - left eye (inner)	20 - right index
2 - left eye	21 - left thumb
3 - left eye (outer)	22 - right thumb
4 - right eye (inner)	23 - left hip
5 - right eye	24 - right hip
6 - right eye (outer)	25 - left knee
7 - left ear	26 - right knee
8 - right ear	27 - left ankle
9 - mouth (left)	28 - right ankle
10 - mouth (right)	29 - left heel
11 - left shoulder	30 - right heel
12 - right shoulder	31 - left foot index
13 - left elbow	32 - right foot index
14 - right elbow	
15 - left wrist	
16 - right wrist	
17 - left pinky	
18 - right pinky	

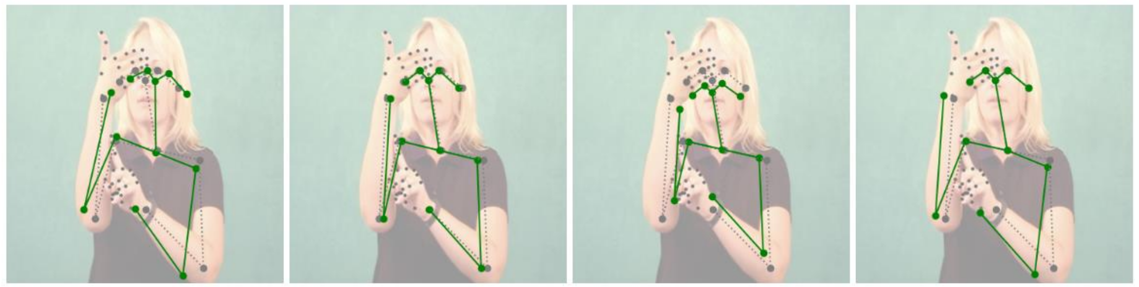
Pose Definition - Hand

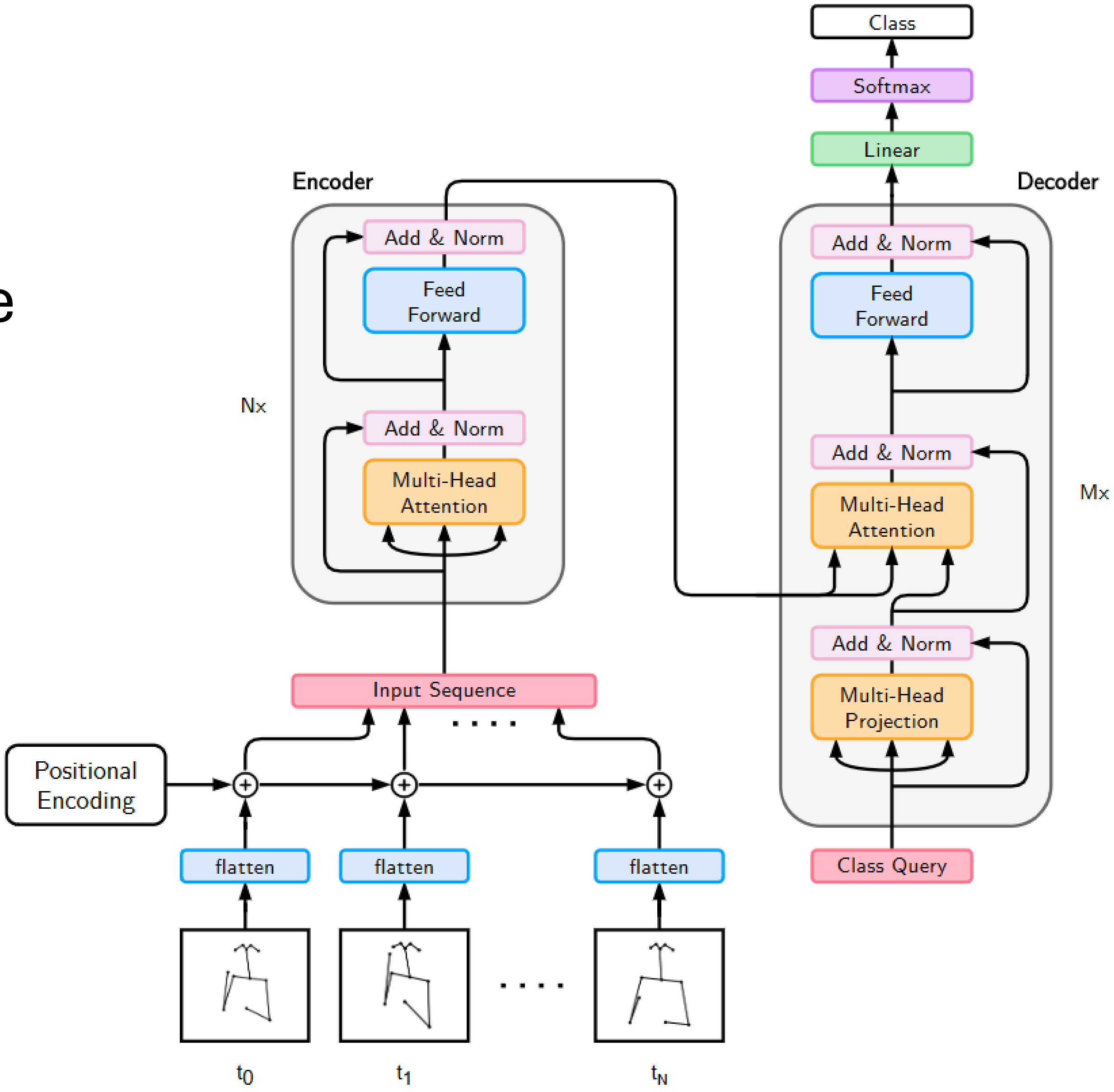


0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

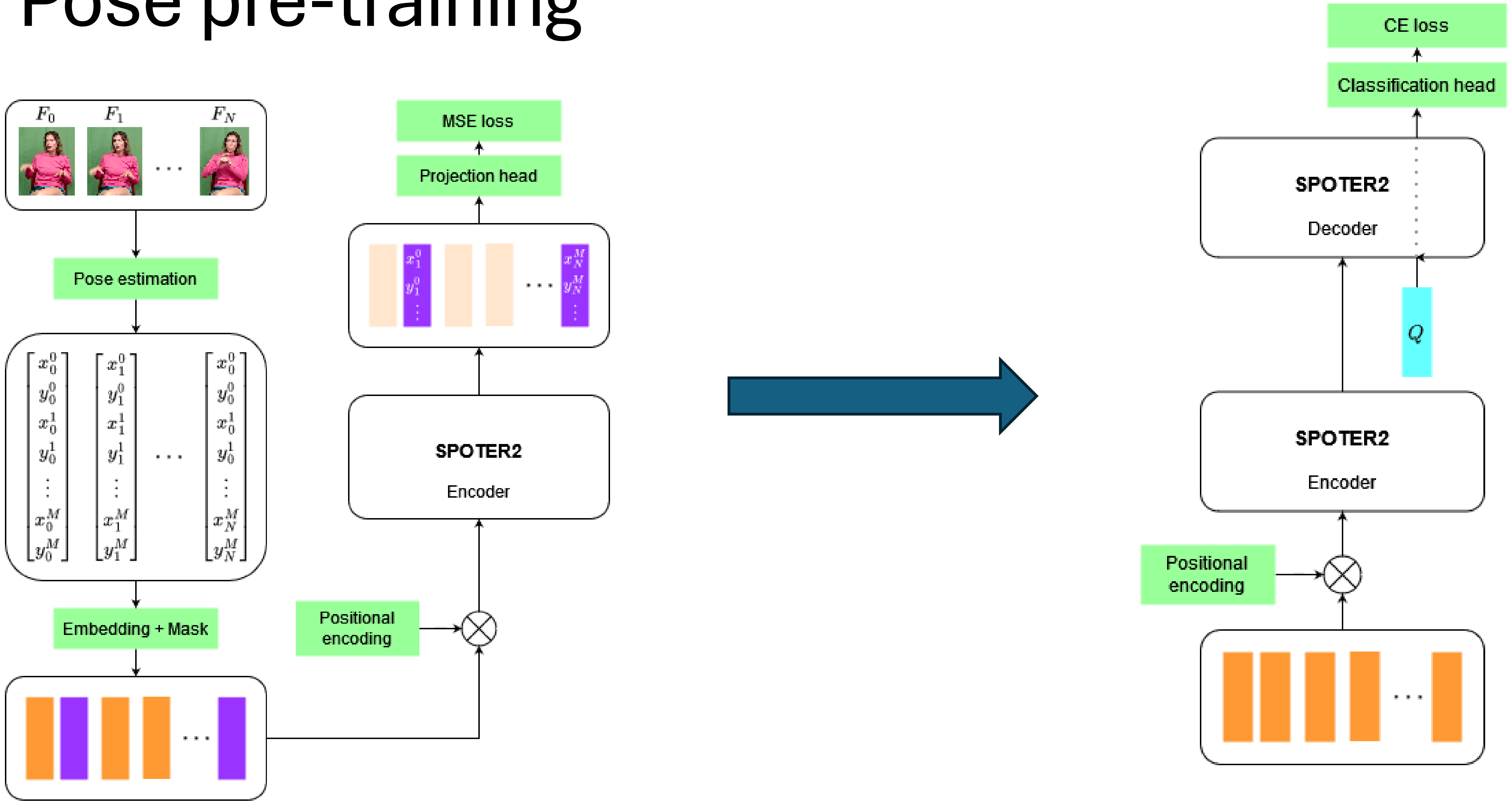
SPOTER architecture

- Pose based isolated sign language recognition.
- A clever way of normalization and augmentation.
- Single learned query decoder.

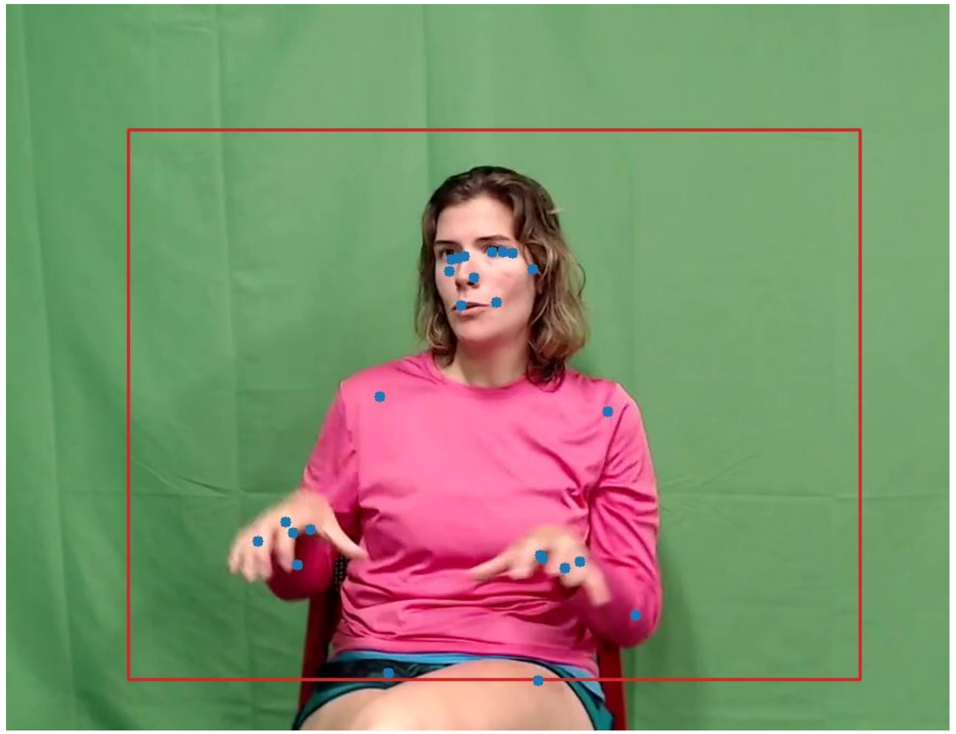


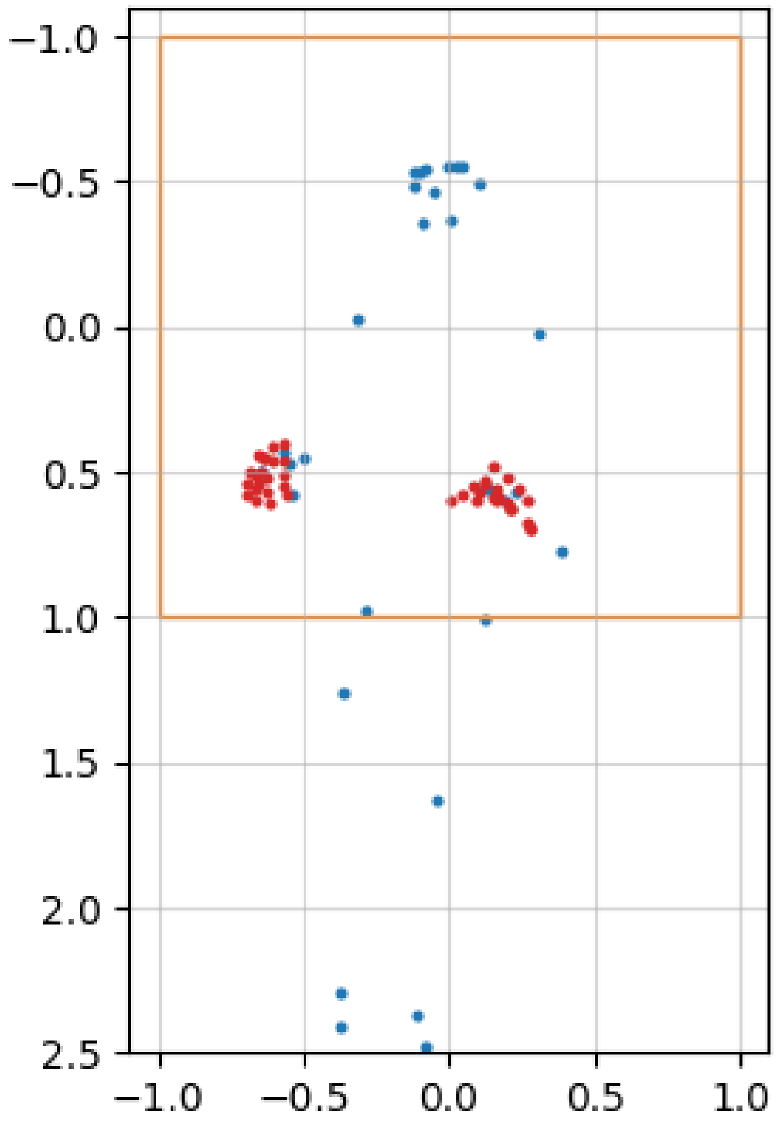


Pose pre-training

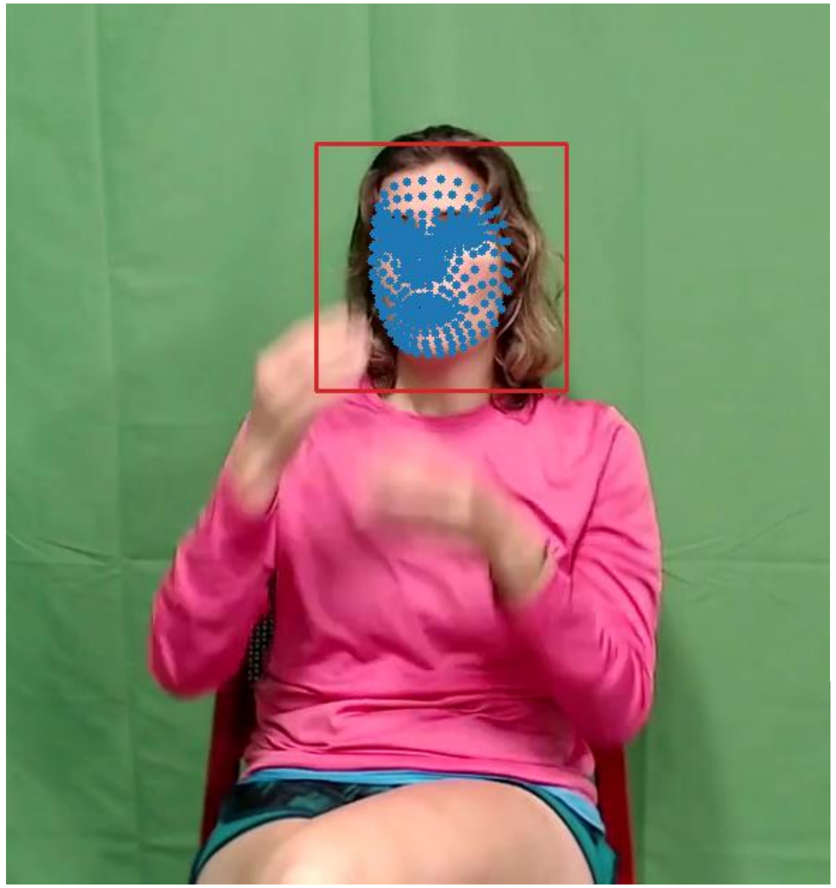


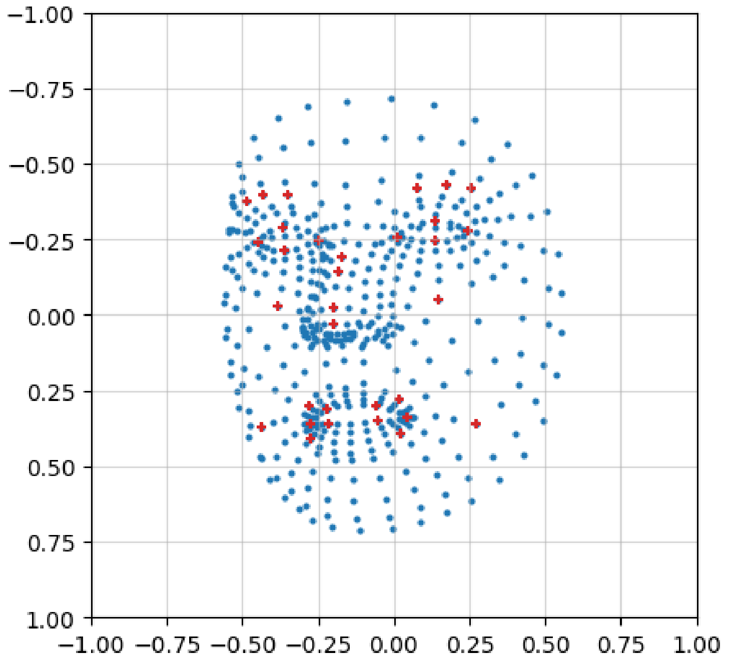
Pose normalization

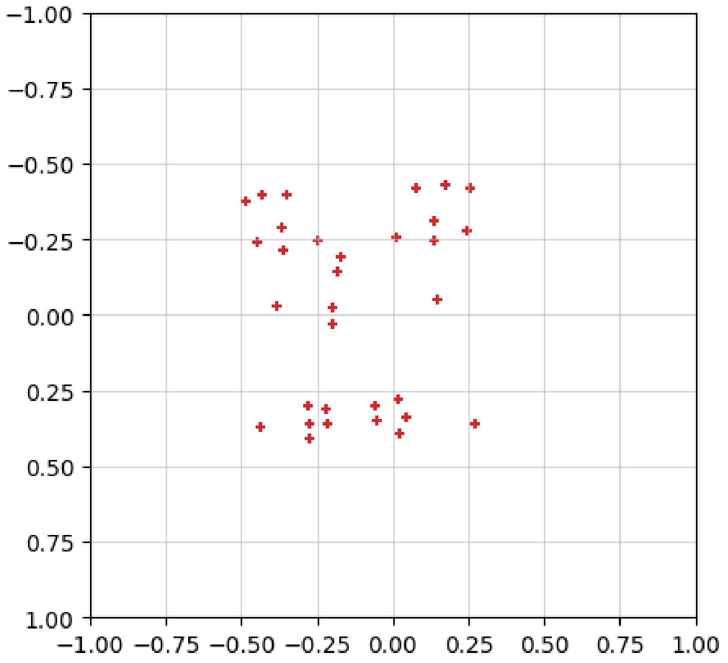




Face normalization

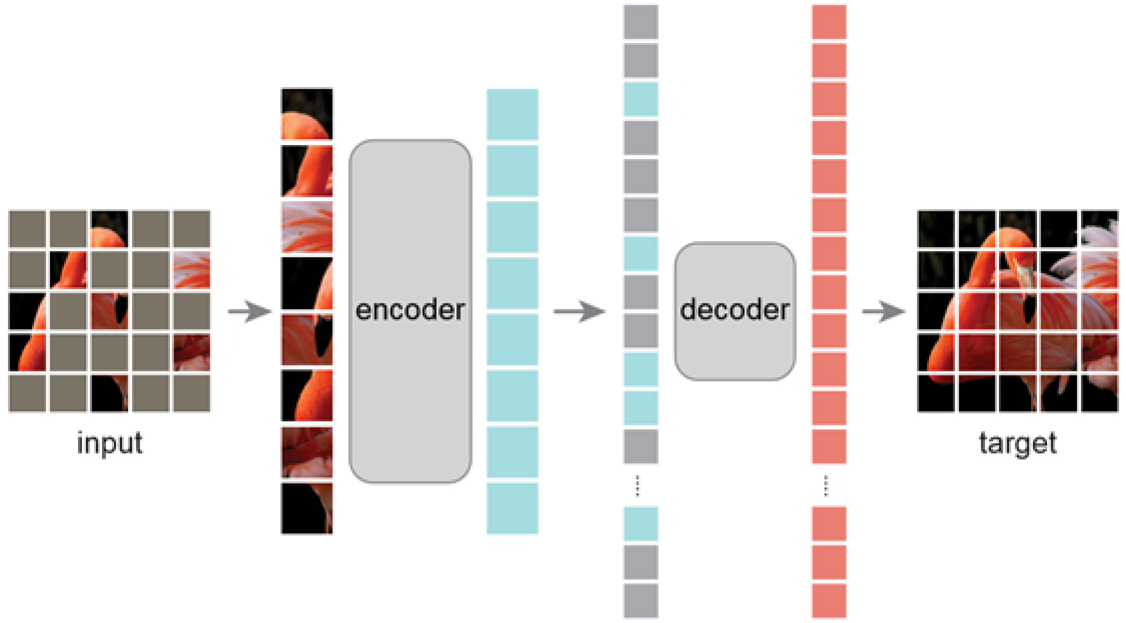


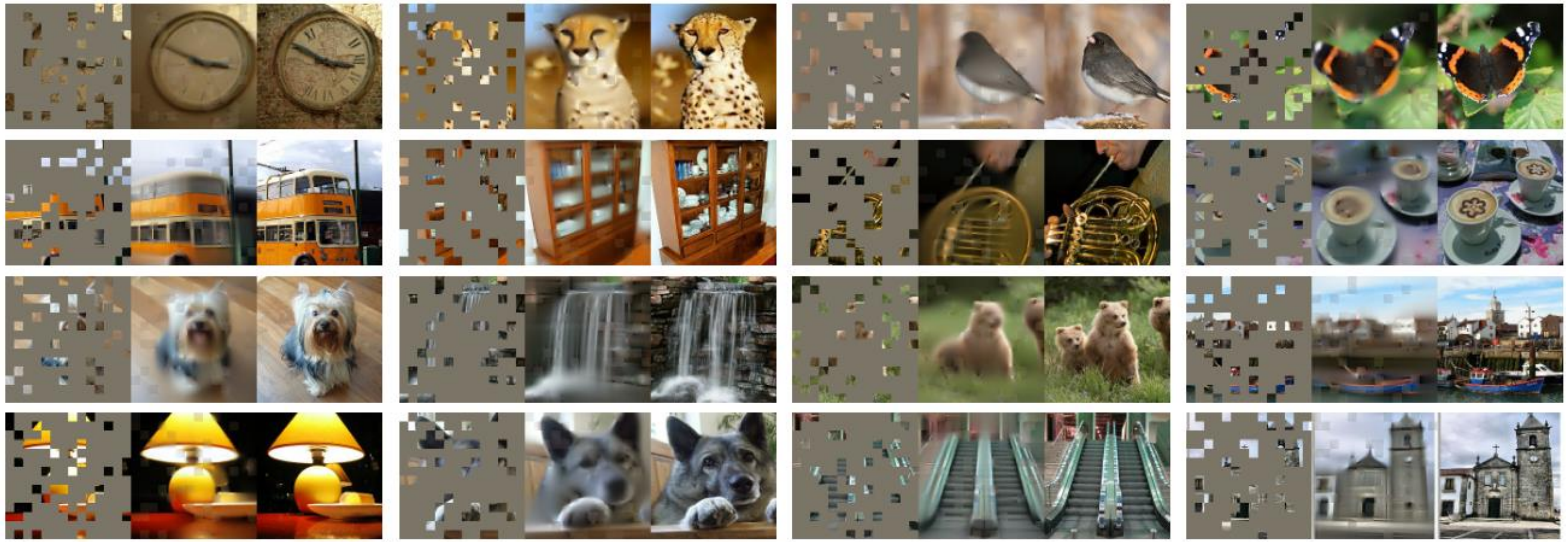


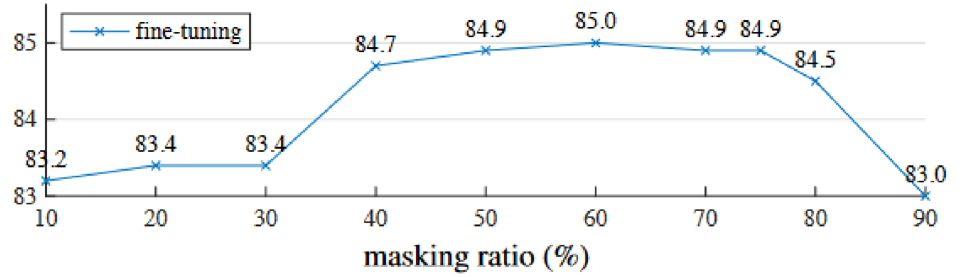


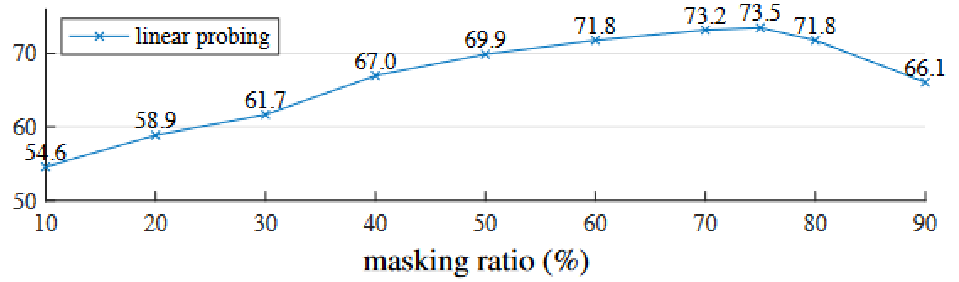
Visual pre-training

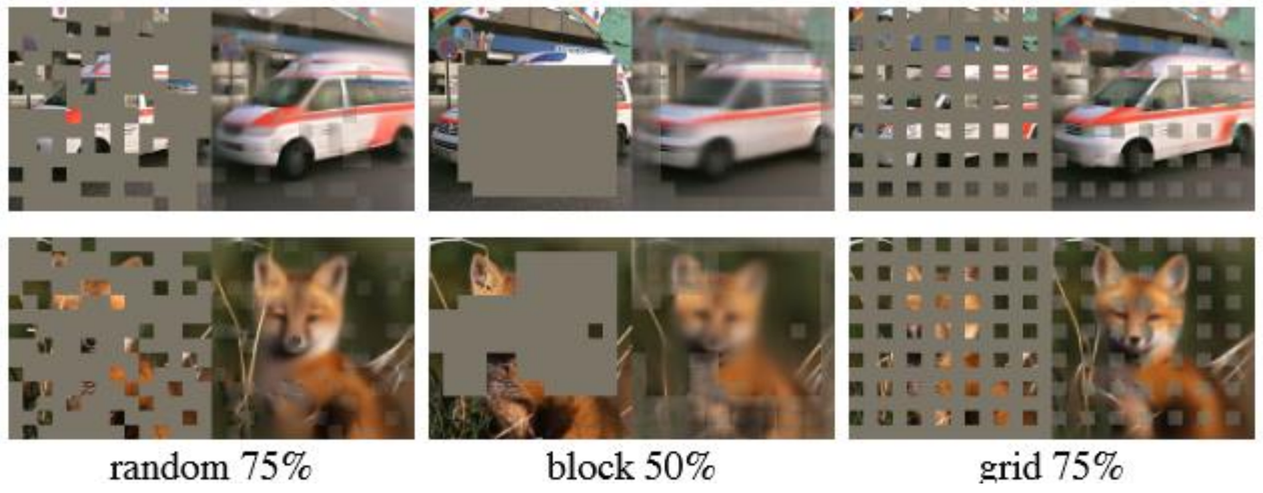
- Masked Autoencoders (Vision Transformers), and DINO
- Informed masking of images - mask only relevant parts.
- Given a pose, we can mask individual parts of the human body.
- Similar to masking whole words in sentences rather than random characters.

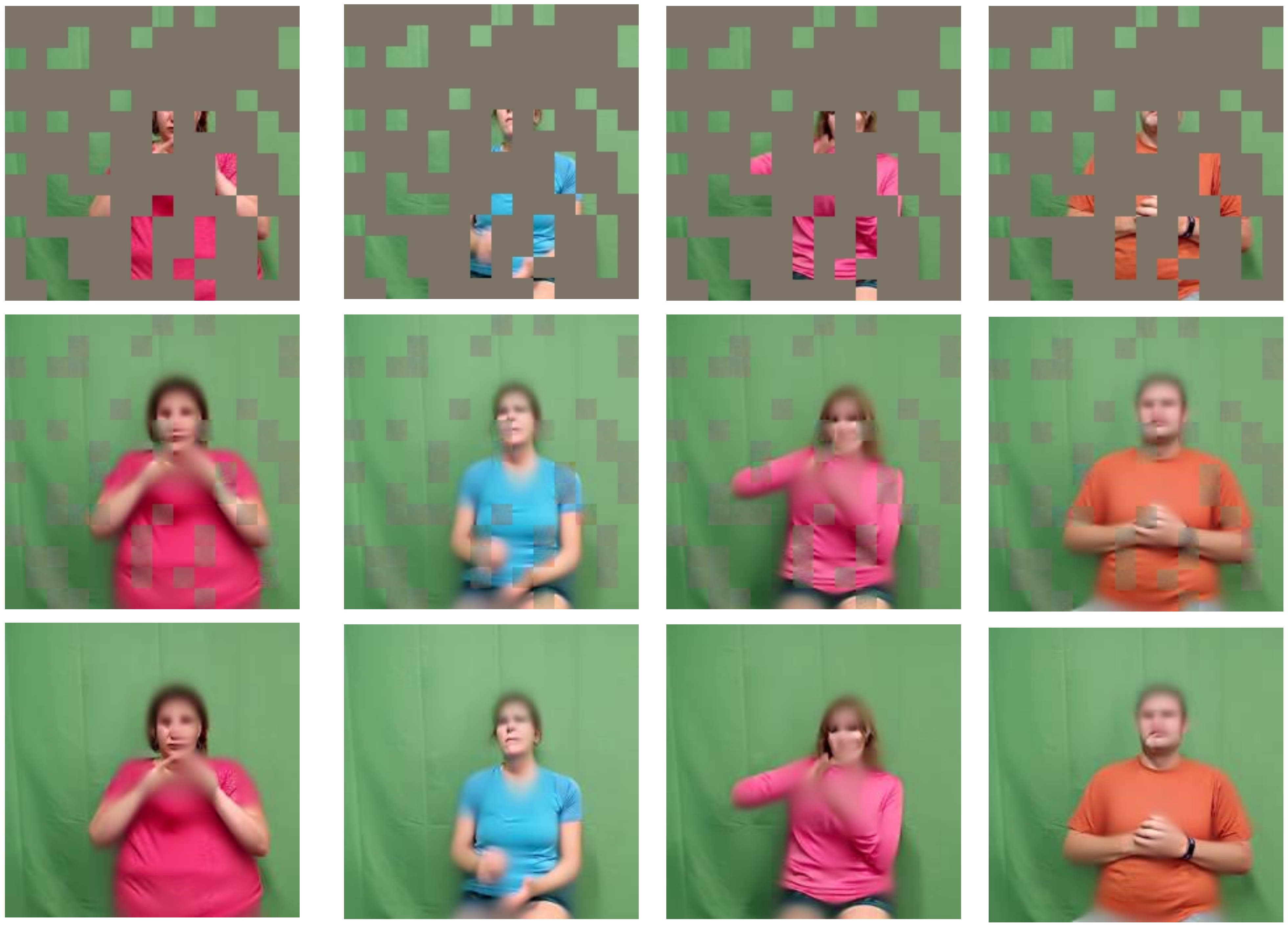





fine-tuning
85
84
83
83.2
83.4
83.4
84.7
84.9
85.0
84.9
84.9
84.5
83.0
10
20
30
40
50
60
70
80
90
masking ratio (%)


linear probing
70
60
50
54.6
58.9
61.7
67.0
69.9
71.8
73.2
73.5
71.8
66.1
10
20
30
40
50
60
70
80
90
masking ratio (%)


random 75%
block 50%
grid 75%



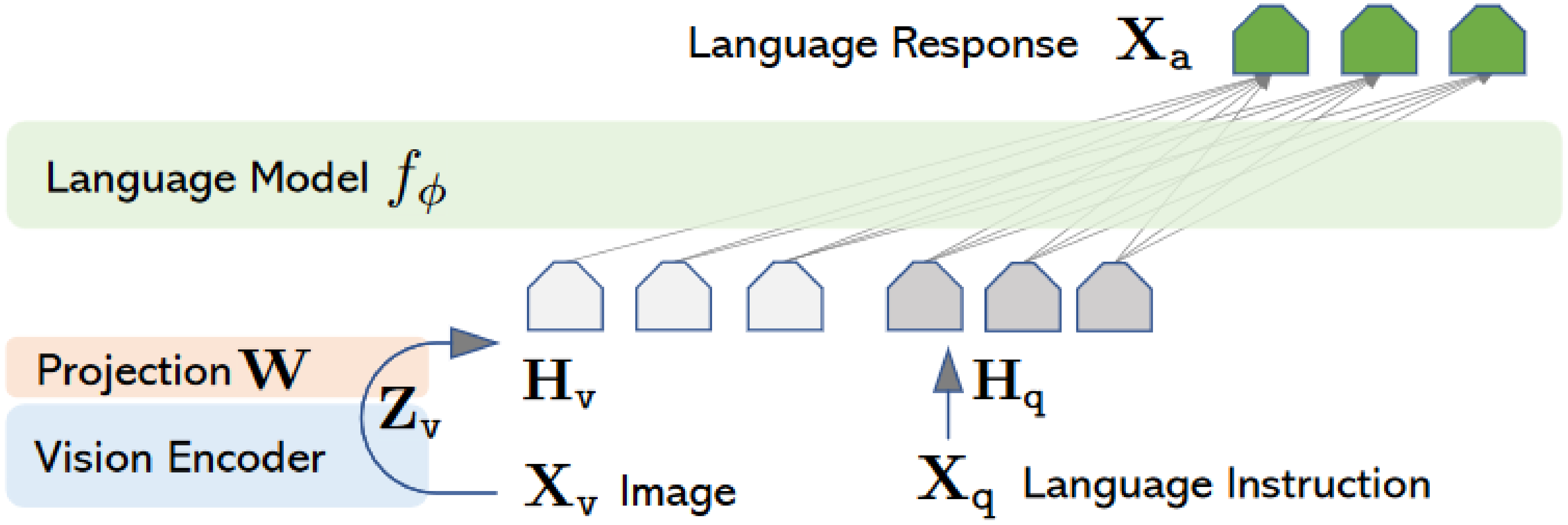
LLaMA

- Large Language Model from Meta.
- Transformer decoder architecture.
- Interesting properties:
  - Pre-normalization (RMSNorm)
  - SwiGLU activation
  - Rotary Embeddings

- LLaMA 2 space:
  - https://huggingface.co/spaces/huggingface-projects/llama-2-13b-chat
- LLaMa 3 blog:
  - https://huggingface.co/blog/llama3

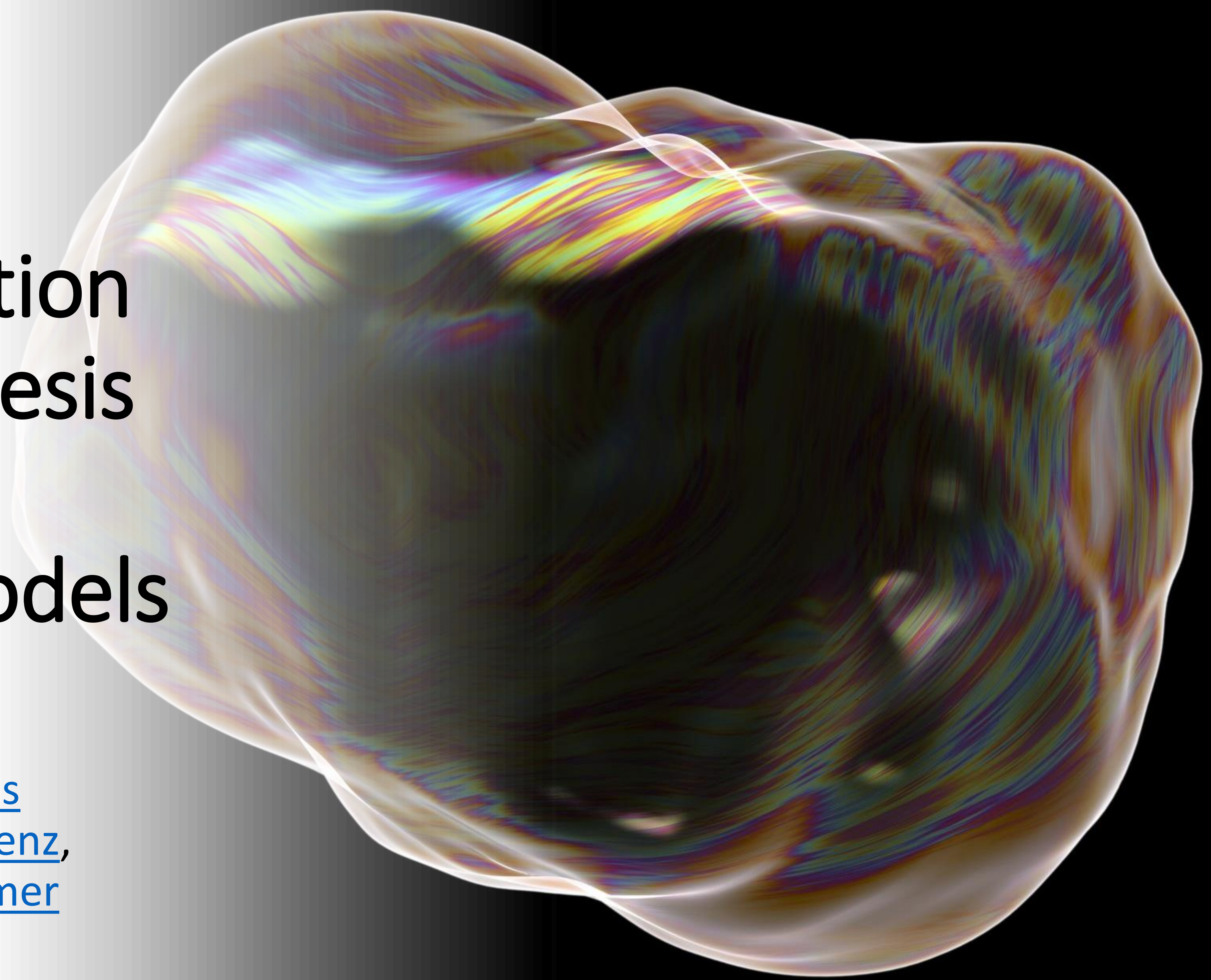
LLaVa

- Visual Instruction Tuning.
- We leave the LLM as is and learn a projection of the image features, so that the language model can reason about the image.



Stable Diffusion

Bonus material



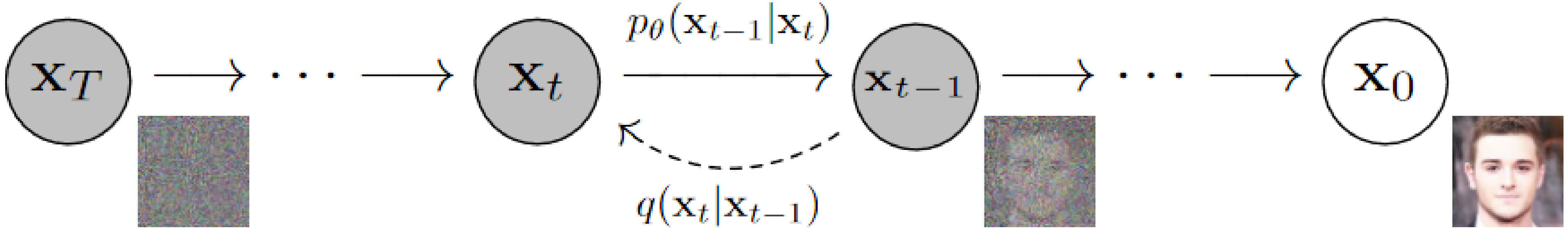
High-Resolution Image Synthesis with Latent Diffusion Models

a.k.a STABLE DIFFUSION

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer

Introduction

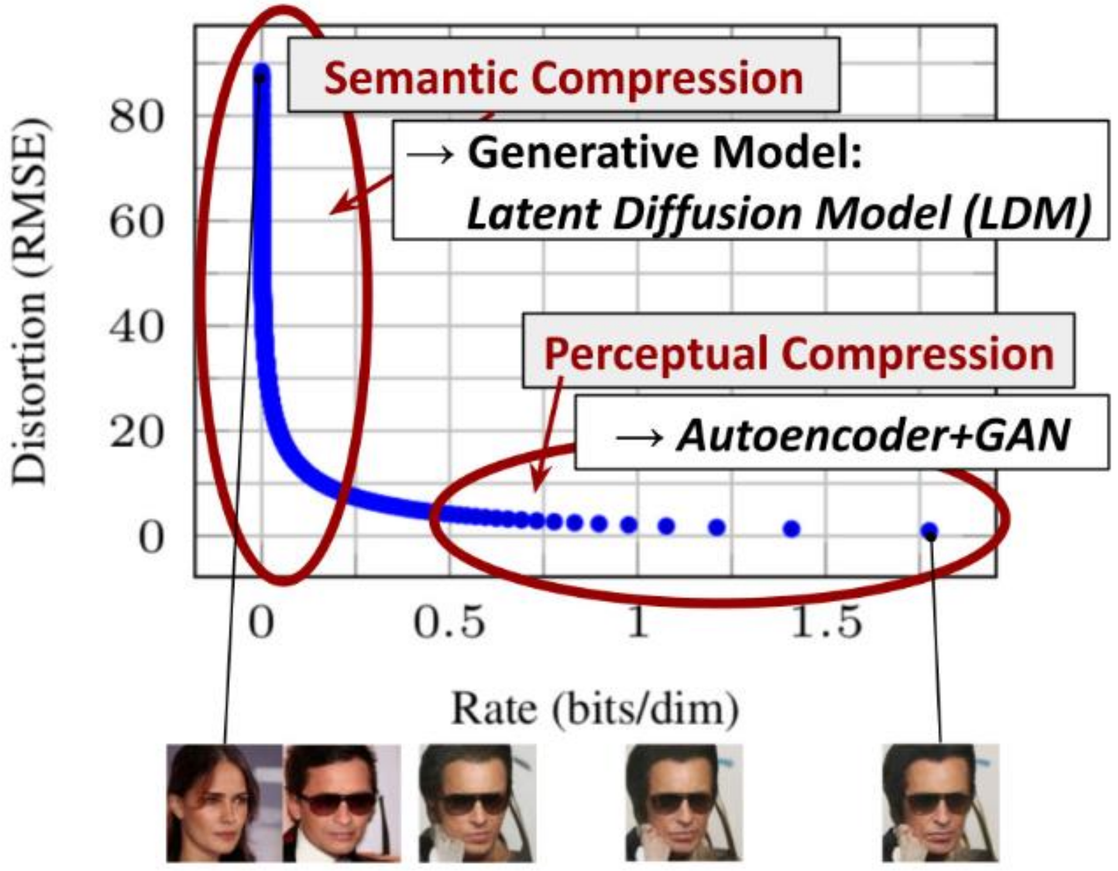
- Image synthesis model
- Democratizing High-Resolution Image Synthesis
  - Diffusion Models (DMs) are trained for many GPU days (150 – 1000 V100 days)
  - Inference is also expensive (50k samples takes 5 days on single A100)



$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$\mathbf{x}_0$

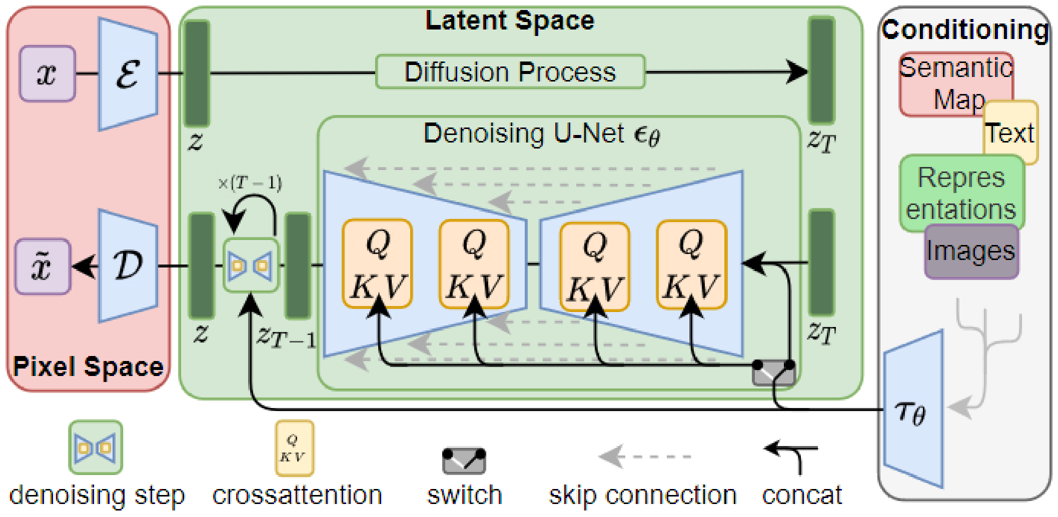
Departure to Latent Space

- Analysis of already trained DMs
- Perceptual Compression
  - Removes high-frequency details but still learns little semantic variation
- Semantic Compression
  - The actual generative model learns the semantic and conceptual composition of the data
- Latent diffusion models (LDMs)
  - as an effective generative model and a separate mild compression stage that only eliminates imperceptible details

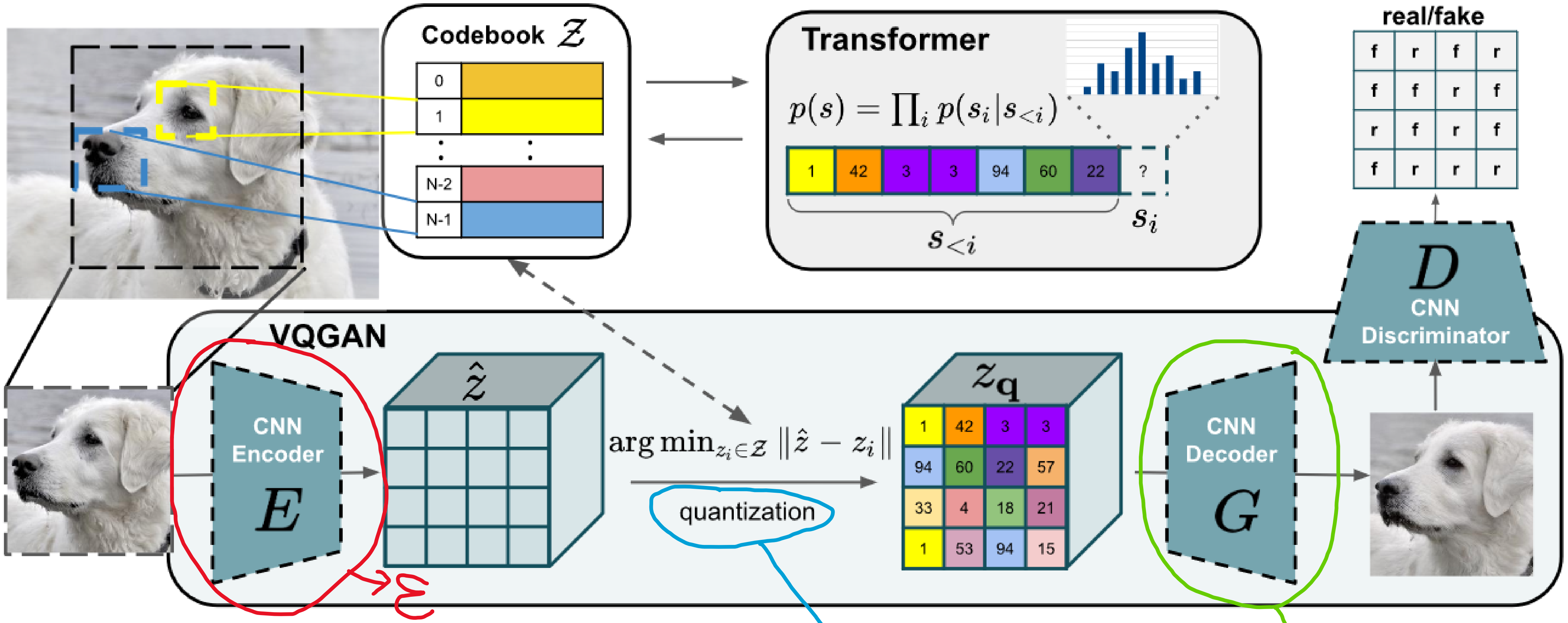


Training overview

- First, we train an autoencoder that provides a lower-dimensional (and thereby efficient) representational space that is perceptually equivalent to the data space
- There is one general autoencoder trained once
- Diffusion model is trained on the latent representation



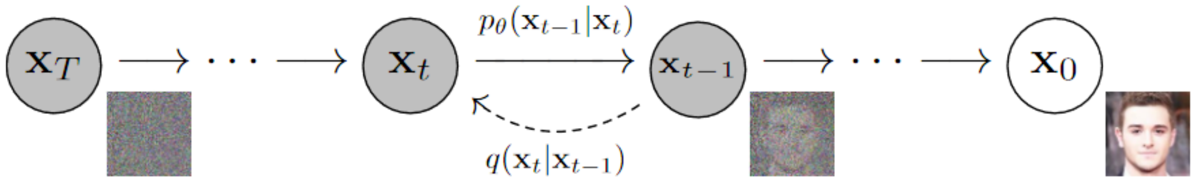
Perceptual Image Compression



Taming Transformers for High-Resolution Image Synthesis

Or KL to
normal dist.

Latent Diffusion Models


$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$\mathbf{x}_{t-1}$
$\mathbf{x}_0$

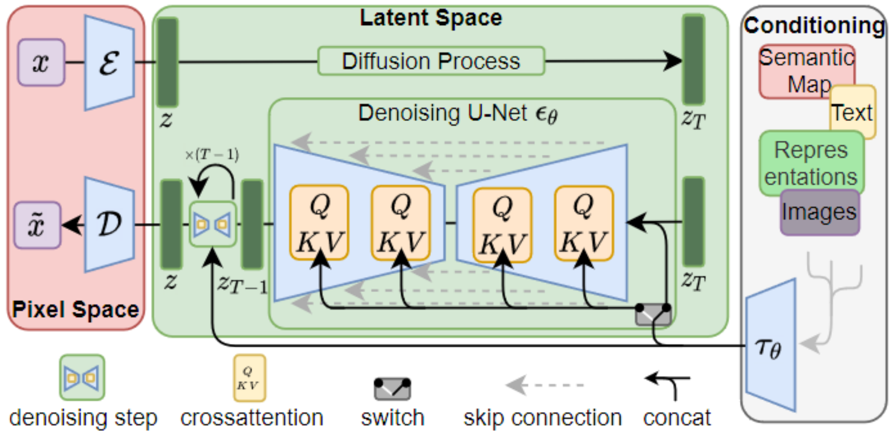
- Gradual denoising of a normally distributed value

$$L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\Big[\|\epsilon - \epsilon_\theta(x_t,t)\|_2^2\Big]$$

- In latent space the loss function takes the form of

$$L_{LDM} := \mathbb{E}_{\mathcal{E}(x),\epsilon\sim\mathcal{N}(0,1),t}\Big[\|\epsilon - \epsilon_\theta(z_t,t)\|_2^2\Big]$$

- ϵ_θ is a U-Net



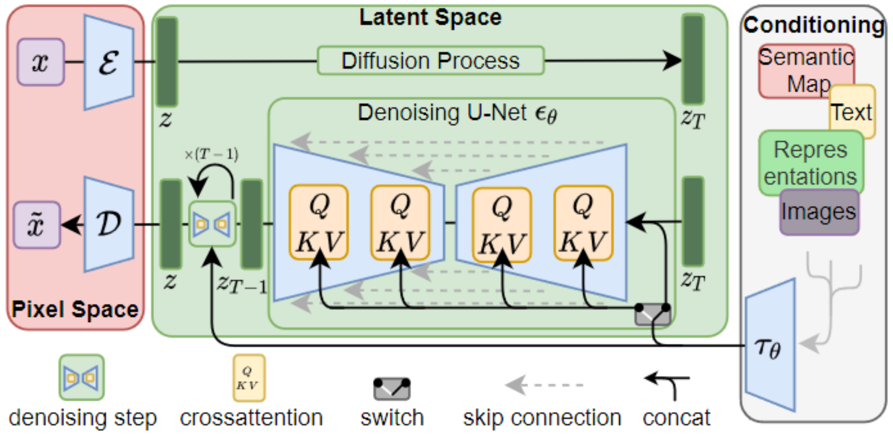
Conditioning Mechanism

- Implemented as a cross-attention mechanism in U-Net
- The conditioning (text, blurred image, segmentation map, …) is processed by an expert model τ_θ resulting in a vector representation
- E.g., for the text conditioning it might be a Transformer Encoder such as BERT

- Depending on the task, the τ_θ is either concatenated to the input latent representation (image-to-image)
- Or is used in the U-Net Conv modules

$$Q = W_Q^{(i)} \cdot \varphi_i(z_t), \; K = W_K^{(i)} \cdot \tau_\theta(y), \; V = W_V^{(i)} \cdot \tau_\theta(y)$$

$$L_{LDM} := \mathbb{E}_{\mathcal{E}(x), y, \epsilon \sim \mathcal{N}(0,1), t}\left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2\right]$$



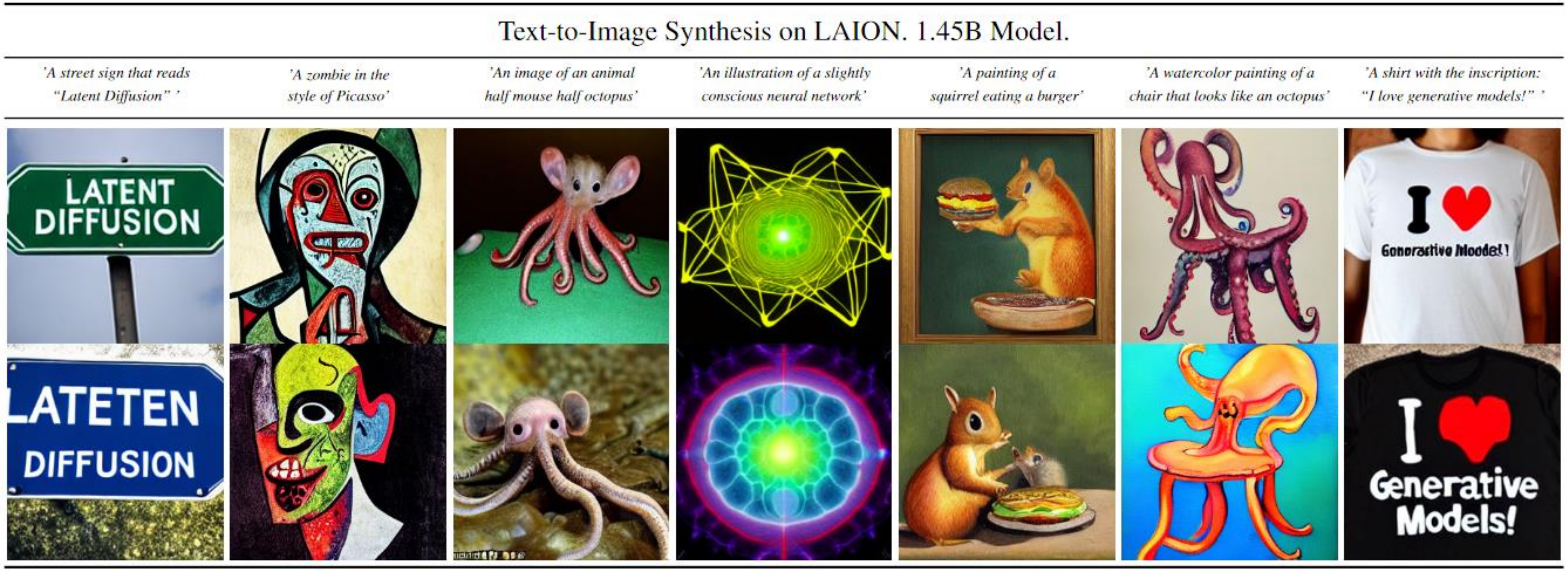

Text-to-Image Synthesis on LAION. 1.45B Model.
'A street sign that reads "Latent Diffusion" '
'A zombie in the style of Picasso'
'An image of an animal half mouse half octopus'
'An illustration of a slightly conscious neural network'
'A painting of a squirrel eating a burger'
'A watercolor painting of a chair that looks like an octopus'
'A shirt with the inscription: "I love generative models!" '
LATENT DIFFUSION
LATETEN DIFFUSION
I
Generative Models!

Figure 5. Samples for user-defined text prompts from our model for text-to-image synthesis, *LDM-8 (KL)*, which was trained on the LAION [78] database. Samples generated with 200 DDIM steps and $\eta = 1.0$. We use unconditional guidance [32] with $s = 10.0$.

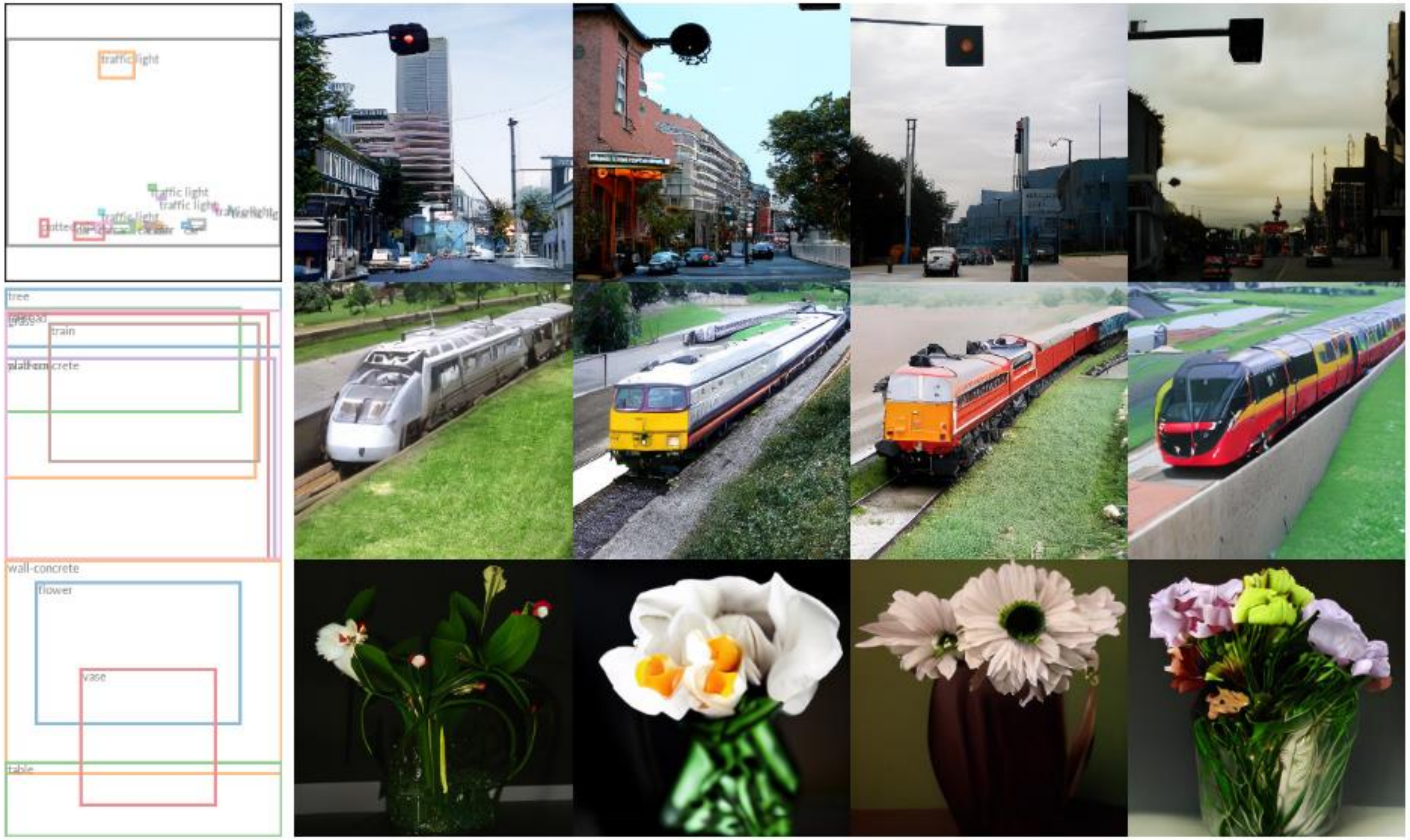


Figure 8. Layout-to-image synthesis with an *LDM* on COCO [4], see Sec. 4.3.1. Quantitative evaluation in the supplement D.3.

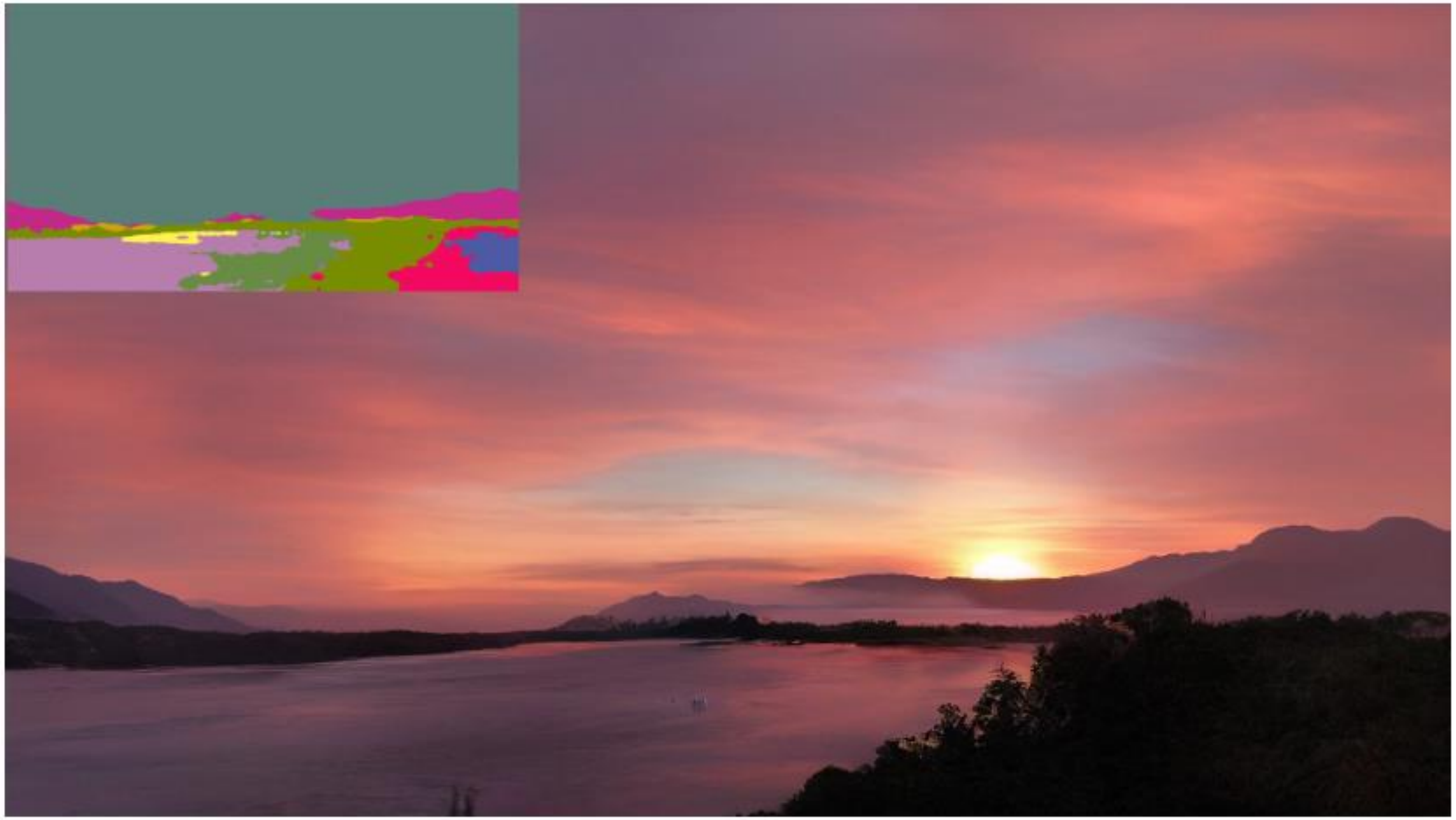


Figure 9. A *LDM* trained on 256^2 resolution can generalize to larger resolution (here: 512×1024) for spatially conditioned tasks such as semantic synthesis of landscape images. See Sec. 4.3.2.

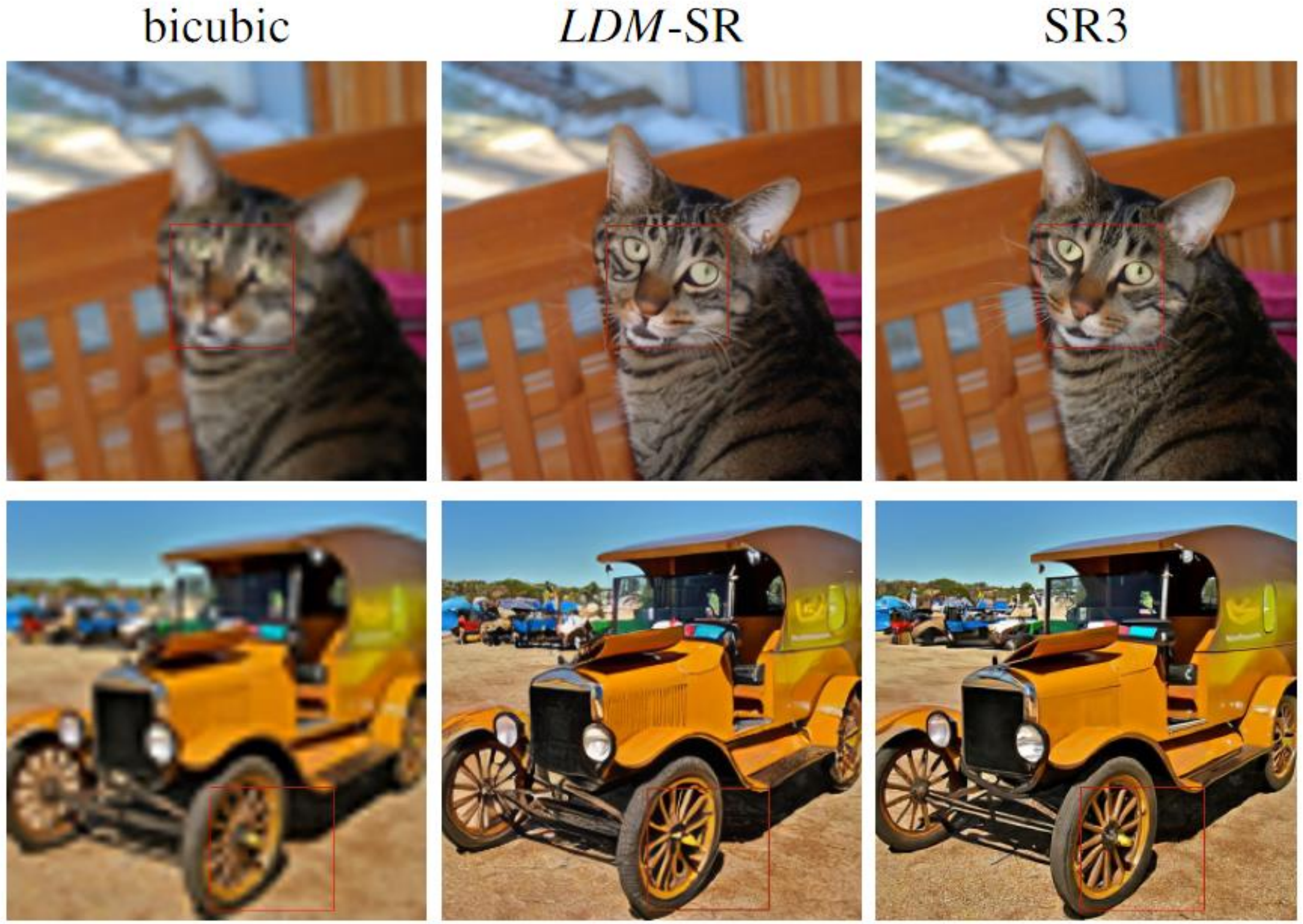


Figure 10. ImageNet 64→256 super-resolution on ImageNet-Val. *LDM-SR* has advantages at rendering realistic textures but SR3 can synthesize more coherent fine structures. See appendix for additional samples and cropouts. SR3 results from [72].

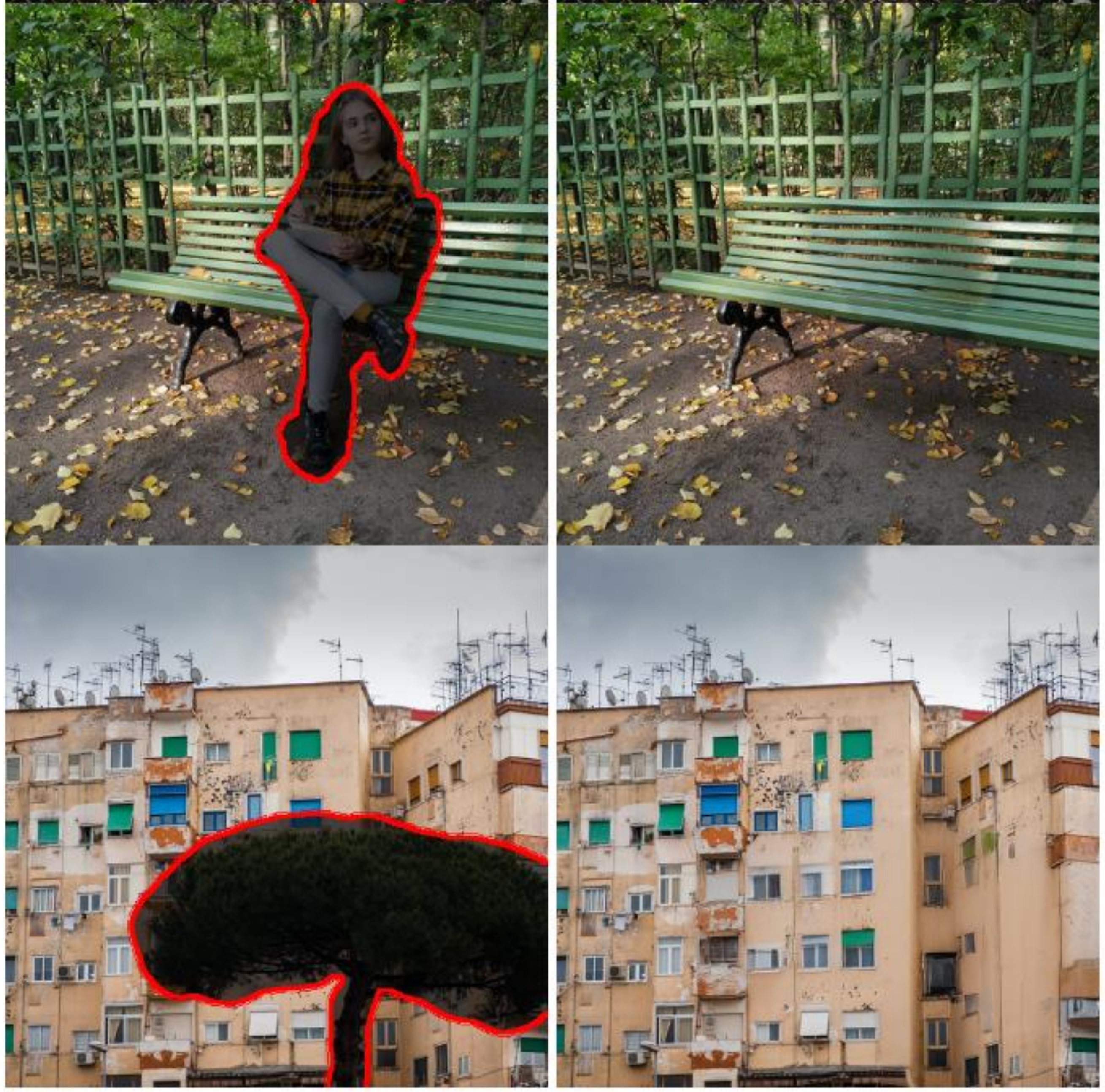


Figure 11. Qualitative results on object removal with our *big, w/ ft* inpainting model. For more results, see Fig. 22.

Questions?