Transformers in Vision

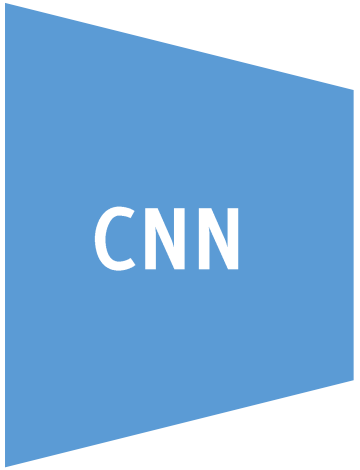
Giacomo Boracchi,
Politecnico di Milano, DEIB.
https://boracchi.faculty.polimi.it/

March 21st, 2025
Advanced Deep Learning, PhD course

Why CNNs are so good?

Stacks of trained convolutional layers have become the **dominant paradigm to solve visual recognition problems**

- Neural Networks are very powerful function approximation
- We know how to train these models on huge amount of data



CNNs have **built-in inductive biases** that are specific for images:

- **Spatial bias** (locality): convolutions operate on the filter support, thus the resulting features refer to contiguous image regions
- **Translation equivariance bias**: convolutions operate the same in each image regions. Very important for handling high resolution images or solving object-detection problems

On top of that, they enable **parallel and efficient processing.**

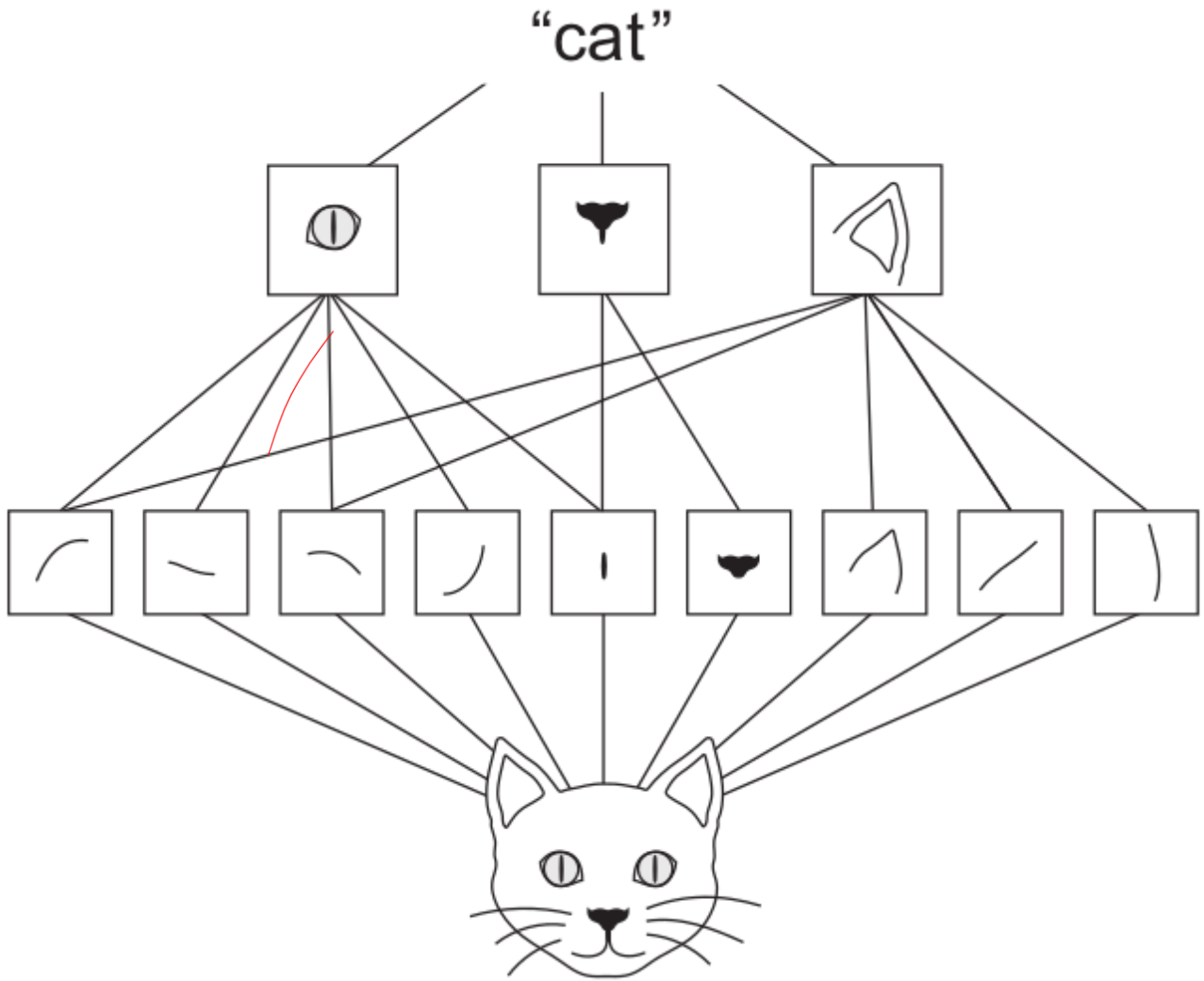
Inductive Biases in CNN

As we move to deeper layers:

- spatial resolution is reduced
- the number of maps increases

We search for **higher-level patterns**, and **don't care too much about their exact location**.

There are more high-level patterns than low-level details!

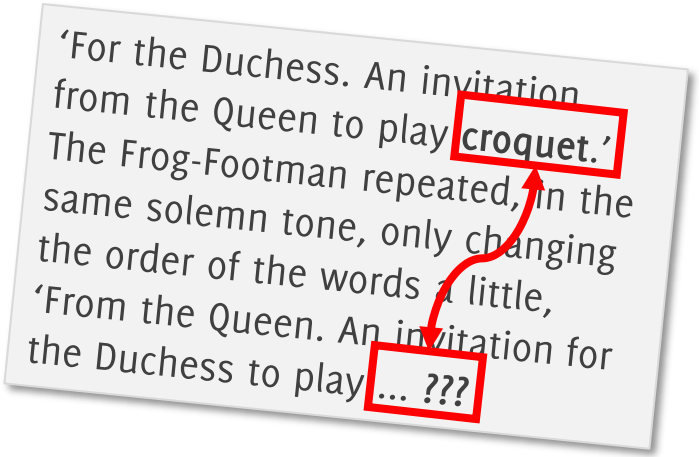


How can CNN exploit Long-Distance Dependencies?

Giacomo Boracchi

Long-Distance Dependencies

These are fundamental in NLP and for images as well



Img credits Mark Carman

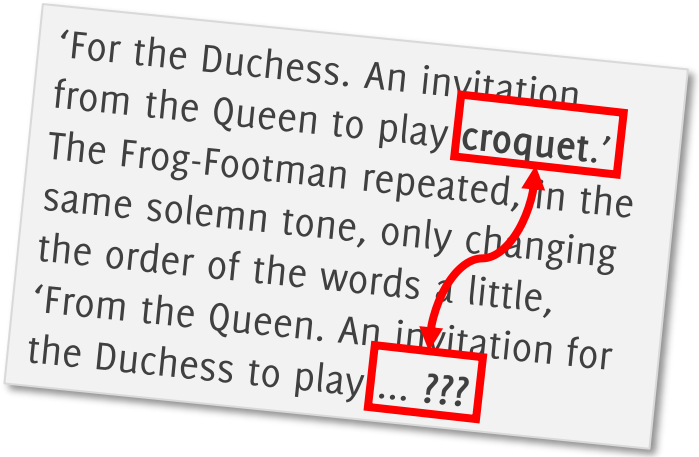
The **relative position** of **distinctive patterns** (eyes, lips) are useful for understanding the image content

Imagined by a GAN (generative adversarial network) StyleGAN2 (Dec 2019) - Karras et al. and Nvidia

Long-Distance Dependencies

These are fundamental in NLP and for images as well

The latent representation gathers **features extracted from regions of the image**. These regions need to be sufficiently large to gather entire patters (e.g. an eye).

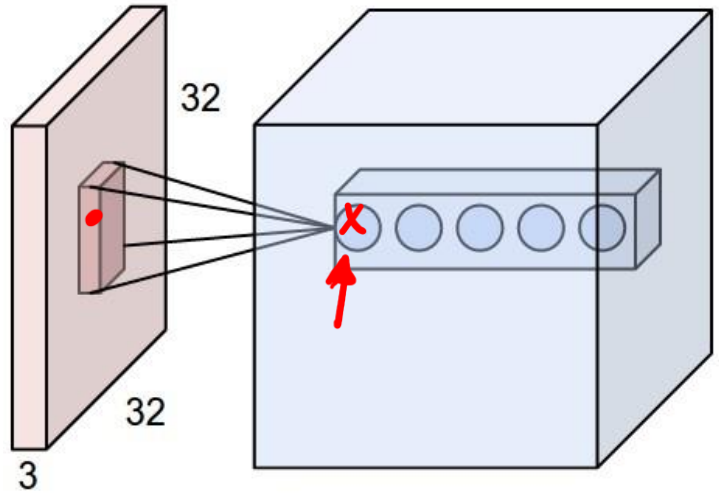


Img credits Mark Carman

Imagined by a GAN (generative adversarial network) StyleGAN2 (Dec 2019) - Karras et al. and Nvidia

The Receptive Field: a key concept of CNNs

- Due to **sparse connectivity** of CNN each output only **depends on a specific region** in the input: the **receptive field.**
- This is different from FC networks where the value of each output depends on the entire input.
- **The deeper you go, the wider the receptive field is**: pooling, convolutions and stride increase the receptive field
- **Usually, the receptive field refers** to the final **output unit** of the network in relation to the network input, but the same definition holds for intermediate volumes



By Aphex34 - Own work, CC BY-SA 4.0, https://commons.wikimedia.org/w/index.php?curid=45659236

More About Receptive Fields

Understanding the Effective Receptive Field in Deep Convolutional Neural Networks

Wenjie Luo* **Yujia Li*** **Raquel Urtasun** **Richard Zemel**
Department of Computer Science
University of Toronto
{wenjie, yujiali, urtasun, zemel}@cs.toronto.edu

The Effective Receptive Field

Not all pixels in a receptive field contribute equally *to an output unit's response. Intuitively it is easy to see that pixels at the center of a receptive field have a much larger impact on an output.*

In the forward pass, central pixels *can* ***propagate*** *information to the output through* ***many different paths****, while the pixels in* ***the outer area*** *of the receptive field have* ***very few paths*** *to propagate their impact.*

A similar mechanism holds for the backward pass, as the gradients for an output unit have **much larger magnitude** in central pixels.

In many cases, the impact in a receptive field distributes as a Gaussian: **the effective receptive field** only occupies a fraction of the **theoretical receptive field**.

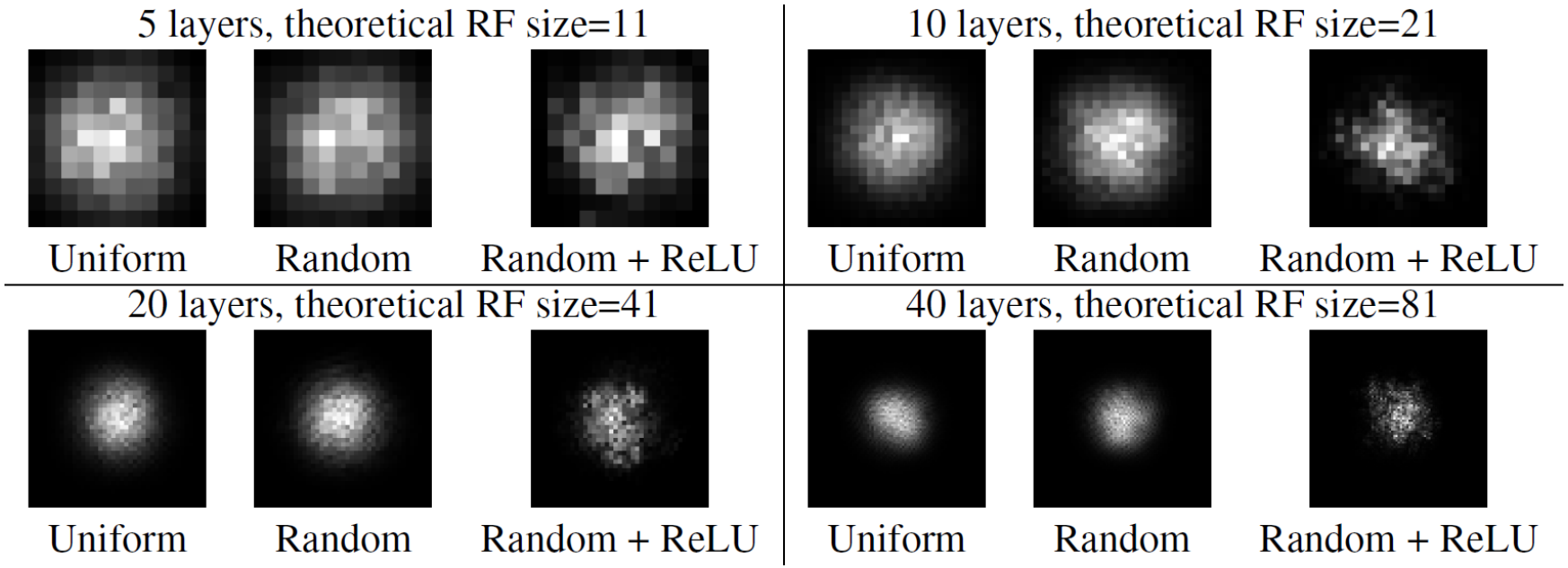
Luo, Wenjie, et al. "Understanding the effective receptive field in deep convolutional neural networks." NIPS 2016

The Effective Receptive Field

The effective receptive field is assessed as

$$\frac{\partial y_{0,0}}{\partial x_{i,j}^{0}}$$

Where $y_{0,0}$ is the center of the last conv layer $x_{i,j}^{0}$ is the i, j pixel in the input.

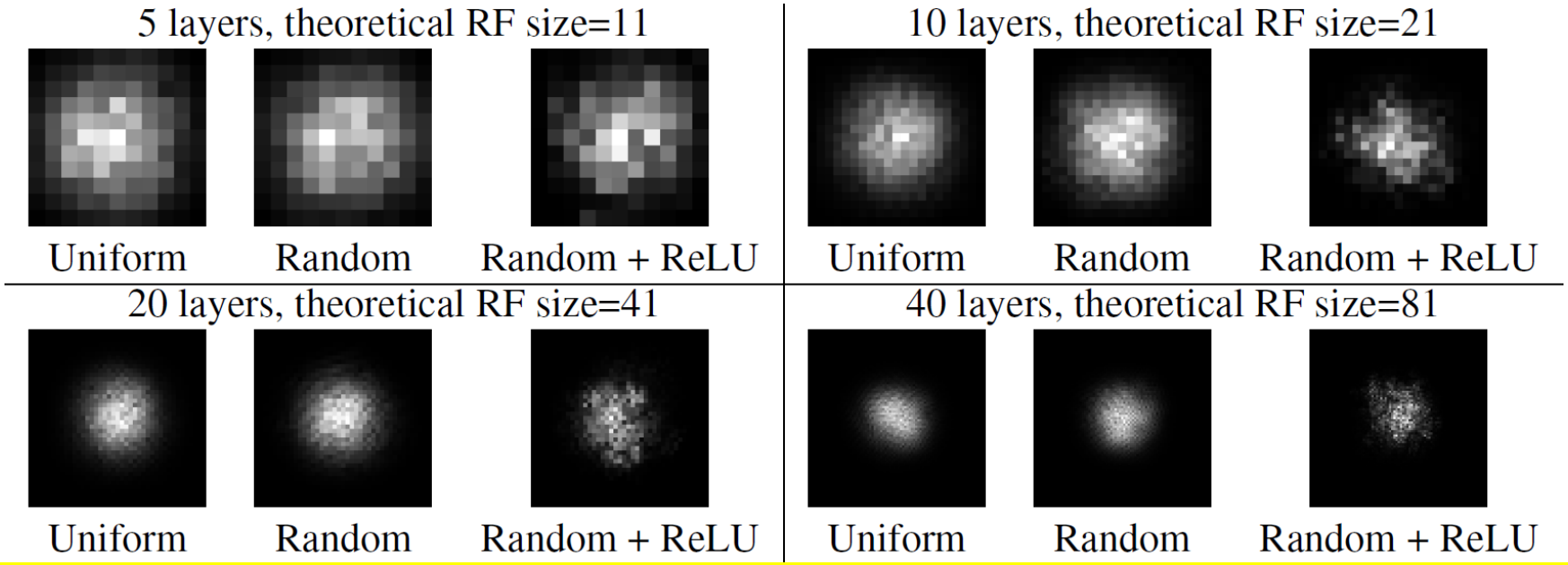


Luo, Wenjie, et al. "Understanding the effective receptive field in deep convolutional neural networks." NIPS 2016

The Effective Receptive Field

Computed by placing a gradient signal of 1 at the center of the output plane ($y_{0,0}$) and 0 everywhere else, and then **back-propagate this gradient** through the network to get input gradients ($x^0_{i,j}$).

Average the resulting gradients at the input over **multiple initialization** of the network

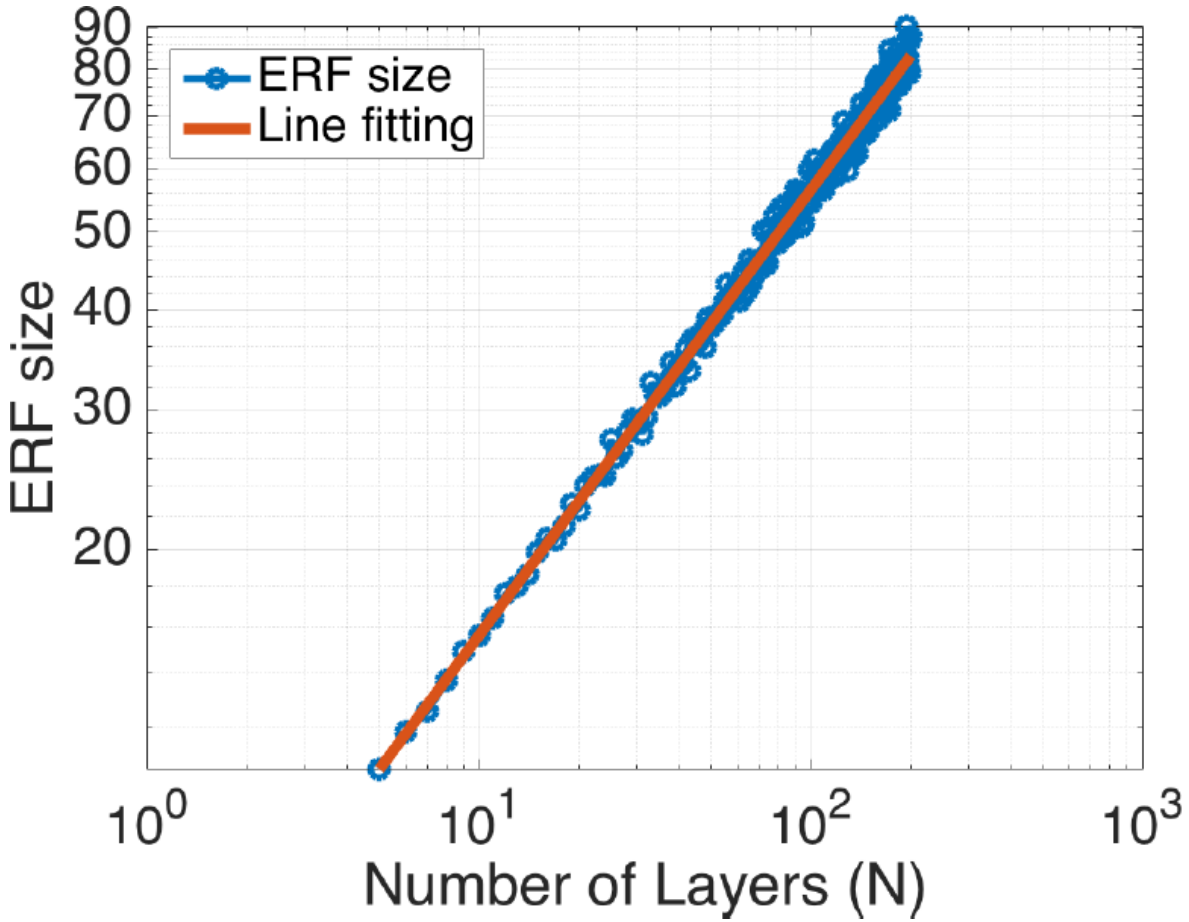


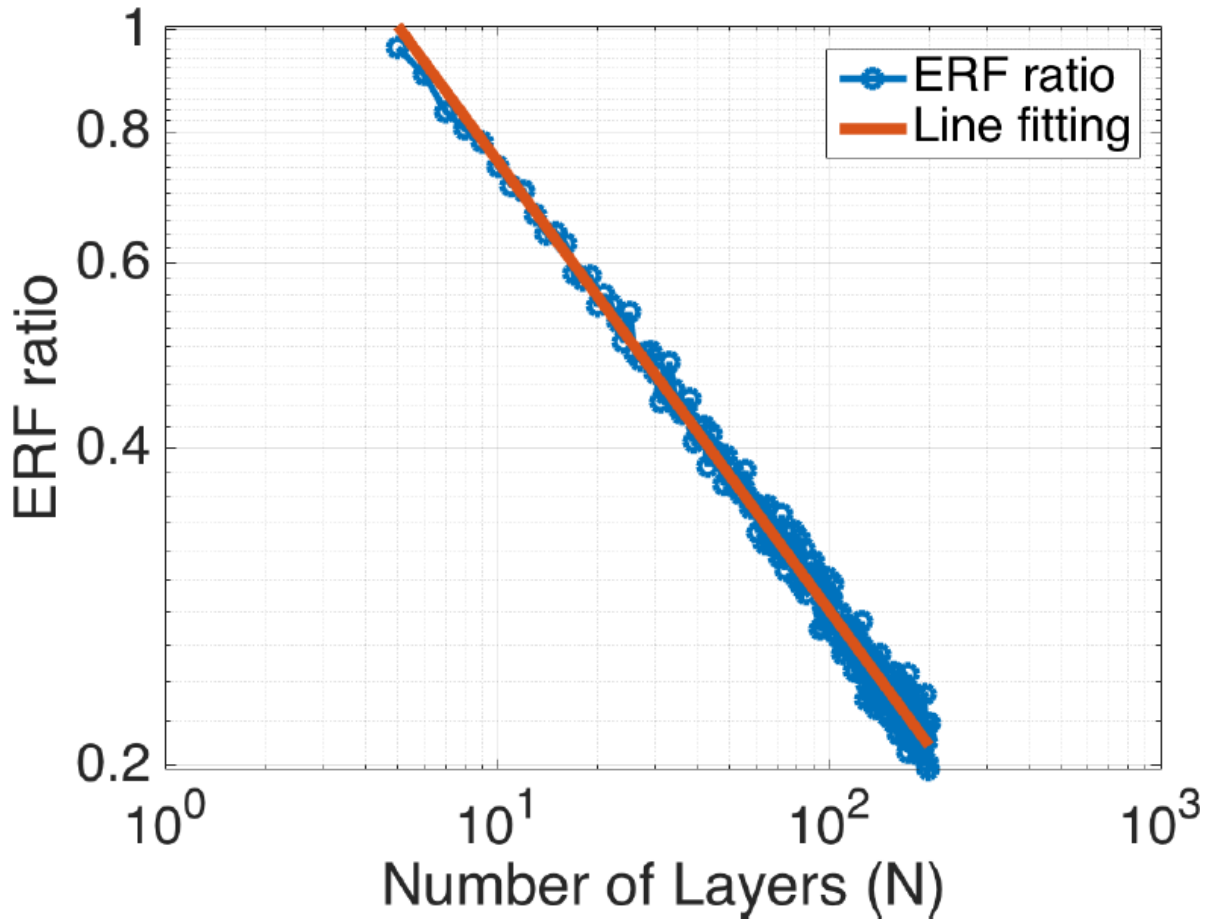
The deeper the network, the smaller the ERF w.r.t. the theoretical one!

Luo, Wenjie, et al. "Understanding the effective receptive field in deep convolutional neural networks." NIPS 2016

The Effective Receptive Field

How the network depth influences the ERF and the theoretical one



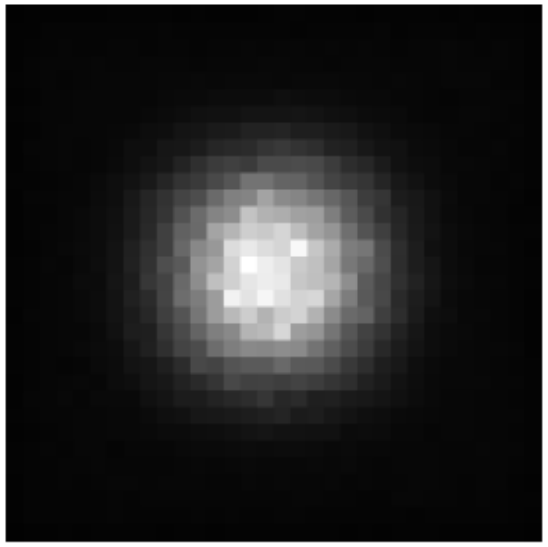


Luo, Wenjie, et al. "Understanding the effective receptive field in deep convolutional neural networks." NIPS 2016

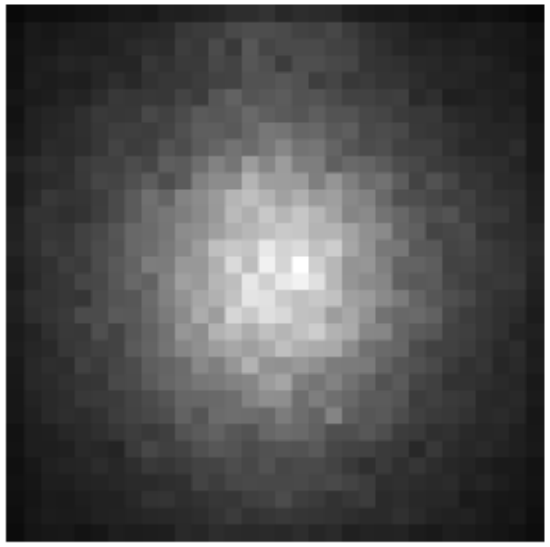
The Effective Receptive Field

Network training increases the ERF. Is random initialization bad?

CIFAR 10

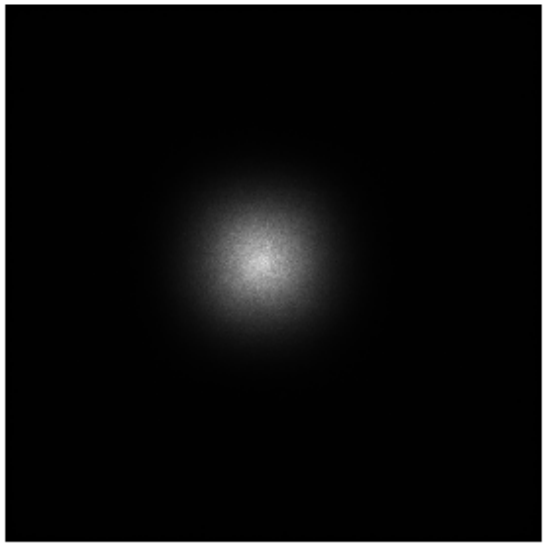


Before Training

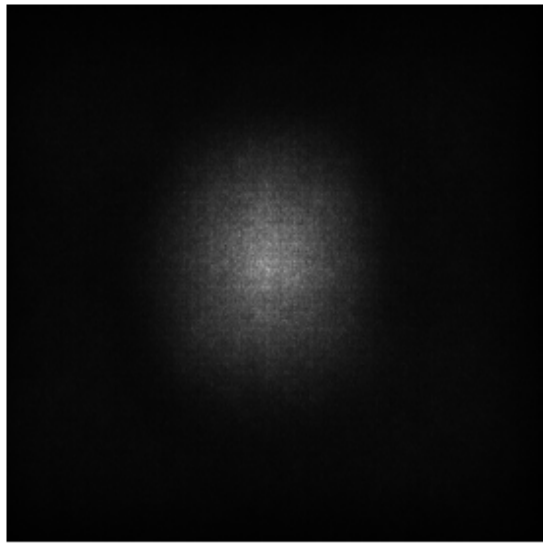


After Training

CamVid



Before Training



After Training

Luo, Wenjie, et al. "Understanding the effective receptive field in deep convolutional neural networks." NIPS 2016

Nonlocality and Self-similarity in Images

Let’s go back to hand-crafted algorithms… there is still much to learn!

Giacomo Boracchi

Self-Similarity in Natural Images

Patches in natural images are self-similar.

For each reference patch (in red) it is possible to find many other patches exhibiting a similar content.

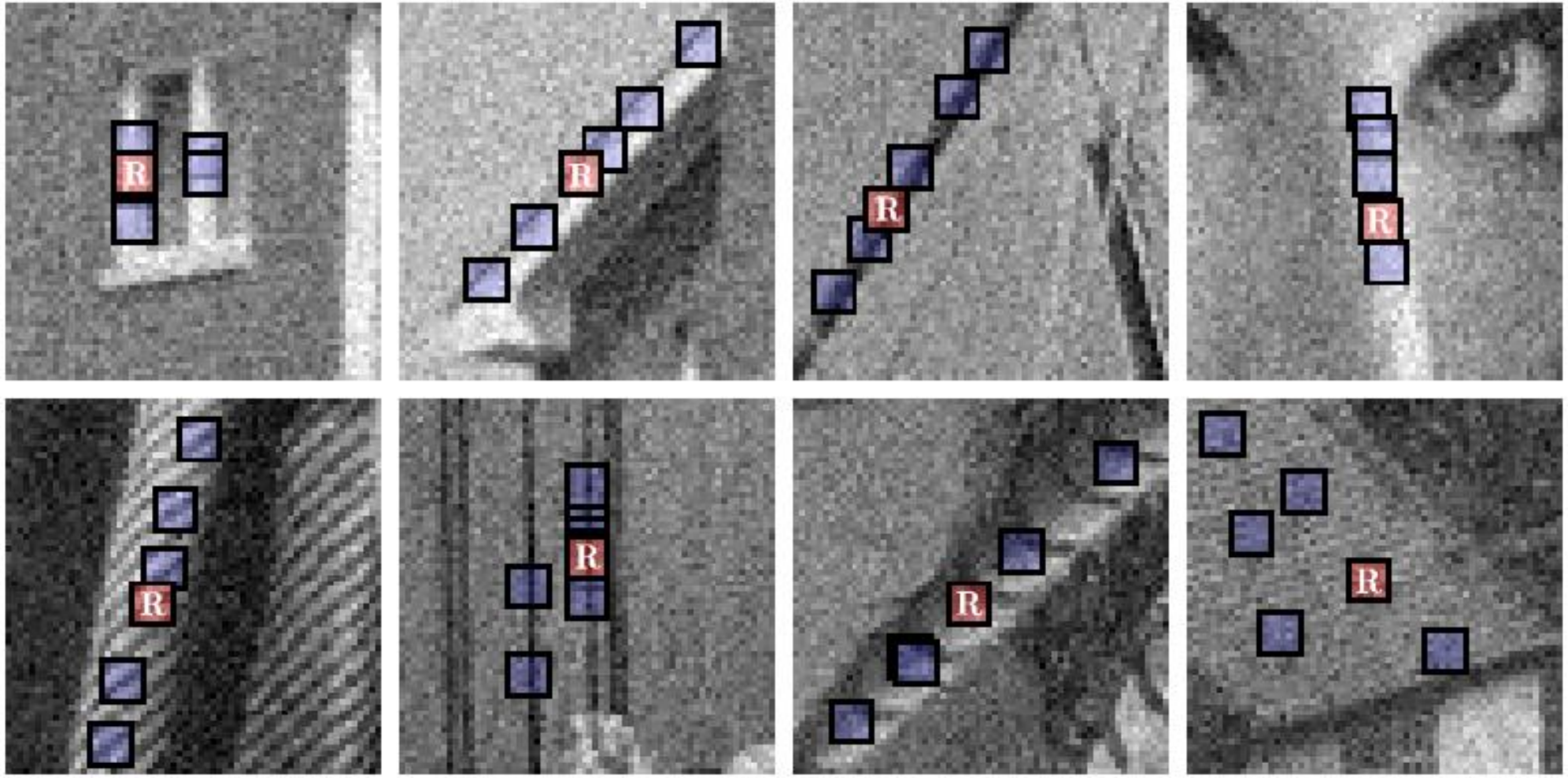
This has been used as a powerful prior to design **image restoration algorithms over the past few years.**

We will see a **key algorithm** to exploit long-distance dependencies in images

Image Credit: https://webpages.tuni.fi/foi/GCF-BM3D/

Dabov, Kostadin, et al. "Image denoising by sparse 3-D transform-domain collaborative filtering." IEEE TIP 2007

Self-Similarity in Natural Images

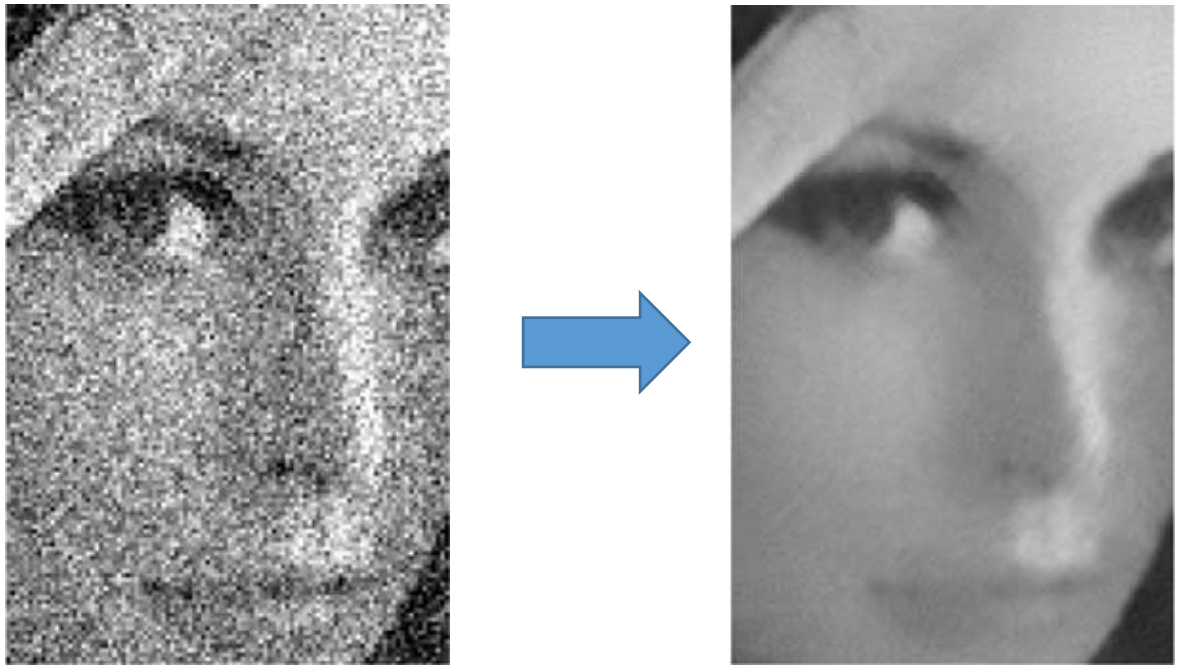


Dabov, Kostadin, et al. "Image denoising by sparse 3-D transform-domain collaborative filtering." IEEE TIP 2007

Self-similarity is a Powerful Denoising Prior

The goal of **image denoising** is to compute $\hat{y}$ *realistic* estimate of the original image y, given the noisy observation z.

Denoising is an **ill posed problem** and requires some form of **regularization** to promote outputs that are close to natural images.



z $\hat{y}$

Denoising: Image Formation Model

Observation model is

$$z(x) = y(x) + \eta(x), \qquad x \in \mathcal{X}$$

z $=$ y $+$ η

Denoising: Image Formation Model

Observation model is

$$z(x) = y(x) + \eta(x), \qquad x \in \mathcal{X}$$

where:

- x denotes the pixel coordinates in the domain $\mathcal{X} \subset \mathbb{Z}^2$
- y is the original (noise-free and unknown) image, $y \in [0,1]$
- z is the noisy observation, $z \in [0,1]$ (clipping)
- η is the noise realization

For the sake of simplicity, we assume Additive White Gaussian Noise (AWGN):

$\eta \sim N(0, \sigma^2)$ and $\eta(x)$ are all independent realizations.

The noise standard deviation σ is also assumed as known.

Image Patches

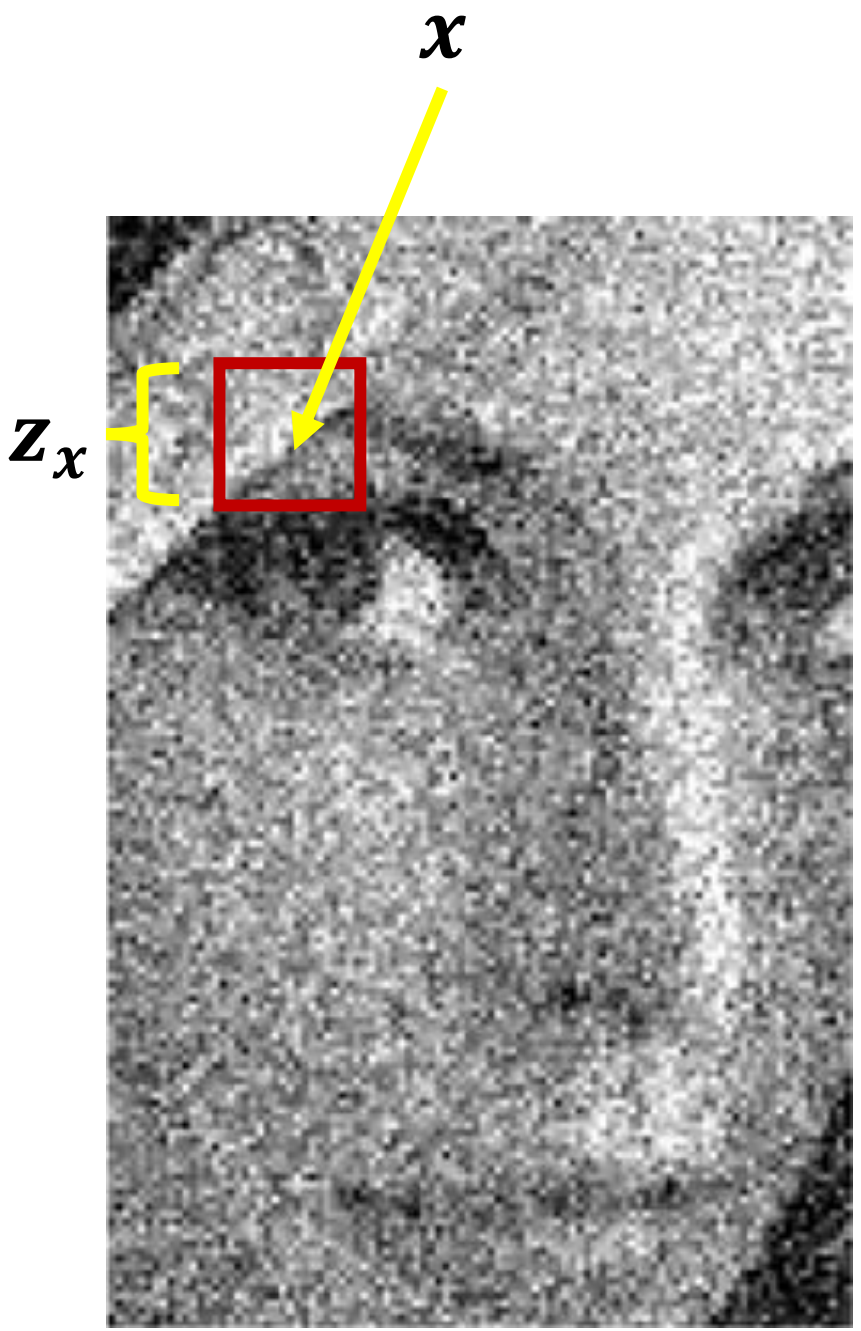
Let $U \subset \mathbb{Z}^2$ be a spatial neighborhood centered at the origin $(0,0) \in \mathbb{Z}^2$,

- we define a patch centered at a pixel $x \in X$ in the observation z

$$\boldsymbol{z_x} = \{z(x+u), \qquad u \in U\}$$

- a patch centered at a pixel $x \in X$ in the original image y

$$\boldsymbol{y_x} = \{y(x+u), \qquad u \in U\}$$



Self-similarity for Denoising

Prior: similar patches will correspond to similar central values.

Namely, the patch-wise distance **correlates well with the distance of the two central pixels** in the noisy-free patches.

The idea is to assess the similarity between pixels $y(x_1)$ and $y(x_2)$ (not available), via the similarity of the corresponding noisy patches $\boldsymbol{z}_{x_1}$ and $\boldsymbol{z}_{x_2}$.

Thus, weights based on photometric differences should be computed from $||\boldsymbol{z}_{x_1} - \boldsymbol{z}_{x_2}||_2$ rather than $z(x_1) - z(x_2)$.

Non Local Means Filter (NL-means)

The denoised image $\hat{y}$ is a weighted average of all image pixels

$$\hat{y}(x_1) = \sum_{x_2 \in X} w\left(\boldsymbol{z}_{\boldsymbol{x_1}}, \boldsymbol{z}_{\boldsymbol{x_2}}\right) z(x_2), \qquad \forall x_1 \in X$$

where weights $\{w(x_1, x_2)\}$ are adaptively defined depending on the similarity between two noisy patches $\boldsymbol{z}_{x_1}$ and $\boldsymbol{z}_{x_2}$

$$w\left(\boldsymbol{z}_{\boldsymbol{x_1}}, \boldsymbol{z}_{\boldsymbol{x_2}}\right) = \frac{e^{\left(-\frac{f\left(\boldsymbol{z}_{\boldsymbol{x_1}}, \boldsymbol{z}_{\boldsymbol{x_2}}\right)}{h^2}\right)}}{\sum_X e^{\left(-\frac{f\left(\boldsymbol{z}_{\boldsymbol{x_1}}, \boldsymbol{z}_{\boldsymbol{x_2}}\right)}{h^2}\right)}}$$

- $f\left(\boldsymbol{z}_{\boldsymbol{x_1}}, \boldsymbol{z}_{\boldsymbol{x_2}}\right)$: distance measure between patches in x_1 and x_2,
- $h > 0$ is a smoothing parameter ($h = \sigma$).
- $\boldsymbol{z}_{x_1}$ similar to $\boldsymbol{z}_{x_2} \Rightarrow f\left(\boldsymbol{z}_{\boldsymbol{x_1}}, \boldsymbol{z}_{\boldsymbol{x_2}}\right)$ is small $\Rightarrow$ $w(x_1, x_2)$ large
- NL-means operates pixel-wise

A. Buades, B. Coll, and J. M. Morel, A review of image denoising algorithms, with a new one, Multisc. Model. Simulat., vol. 4, no. 2, 2005

Windowed Patch Distance in NL-means

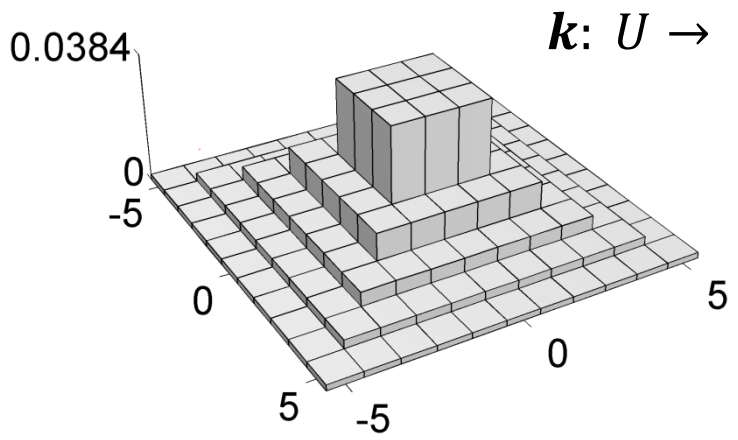
The distance operator is defined as a function $f(z_{x_1}, z_{x_2})$ of the *windowed quadratic distance* between patches

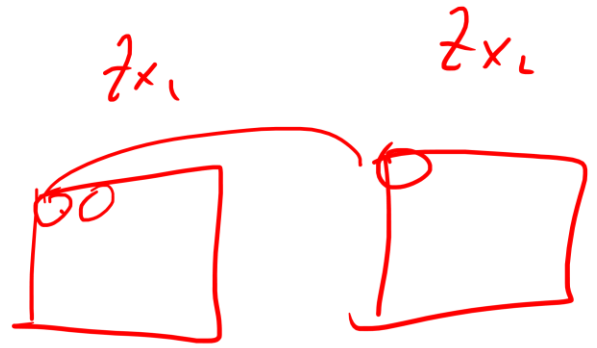
$$f(\boldsymbol{z_{x_1}}, \boldsymbol{z_{x_2}}) = \sum_{u \in U} \big(z(x_1 + u) - z(x_2 + u)\big)^2 \boldsymbol{k}(u)$$

Where $\boldsymbol{k}$: $U \rightarrow \mathbb{R}^+$ is a windowing kernel (we use $\boldsymbol{k}$ since w is being used for the aggregation weights)

Windowed Patch Distance in NL-means

The windowing kernel $\boldsymbol{k}$: $U \rightarrow \mathbb{R}^+$ adjusts the contribution of each difference term depending on the position of u with respect to the patch center.





$$\| z_{x_1} - z_{x_2} \|_2^2$$

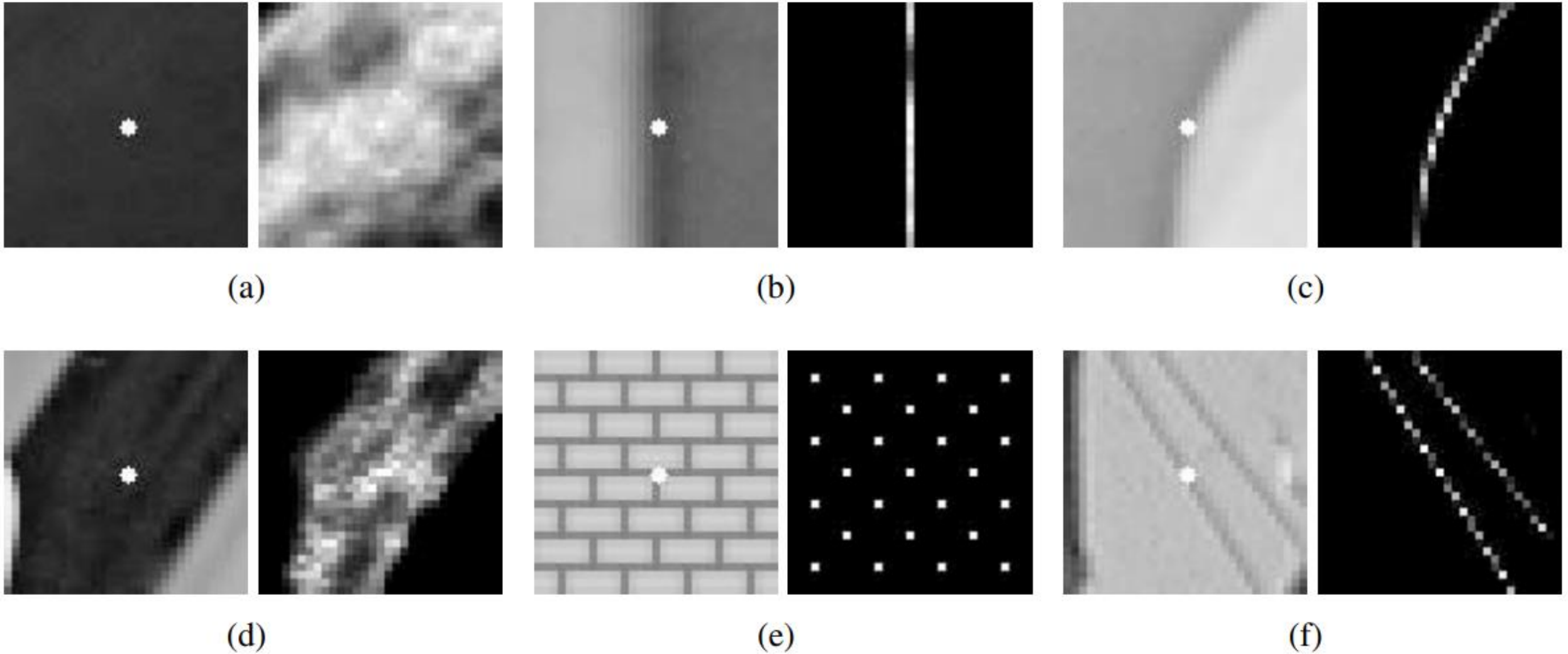
$$\| (z_{x_1} - z_{x_2}) \sqrt{K} \|_2^2$$

d performs a **pixel-wise** comparison of the patches

the decay of $\boldsymbol{k}$ reflects how much similarity between $y(x_1)$ and $y(x_2)$ may be implied from the similarity between $y(x_1 + u)$ and $y(x_2 + u)$ when $u \neq 0$.

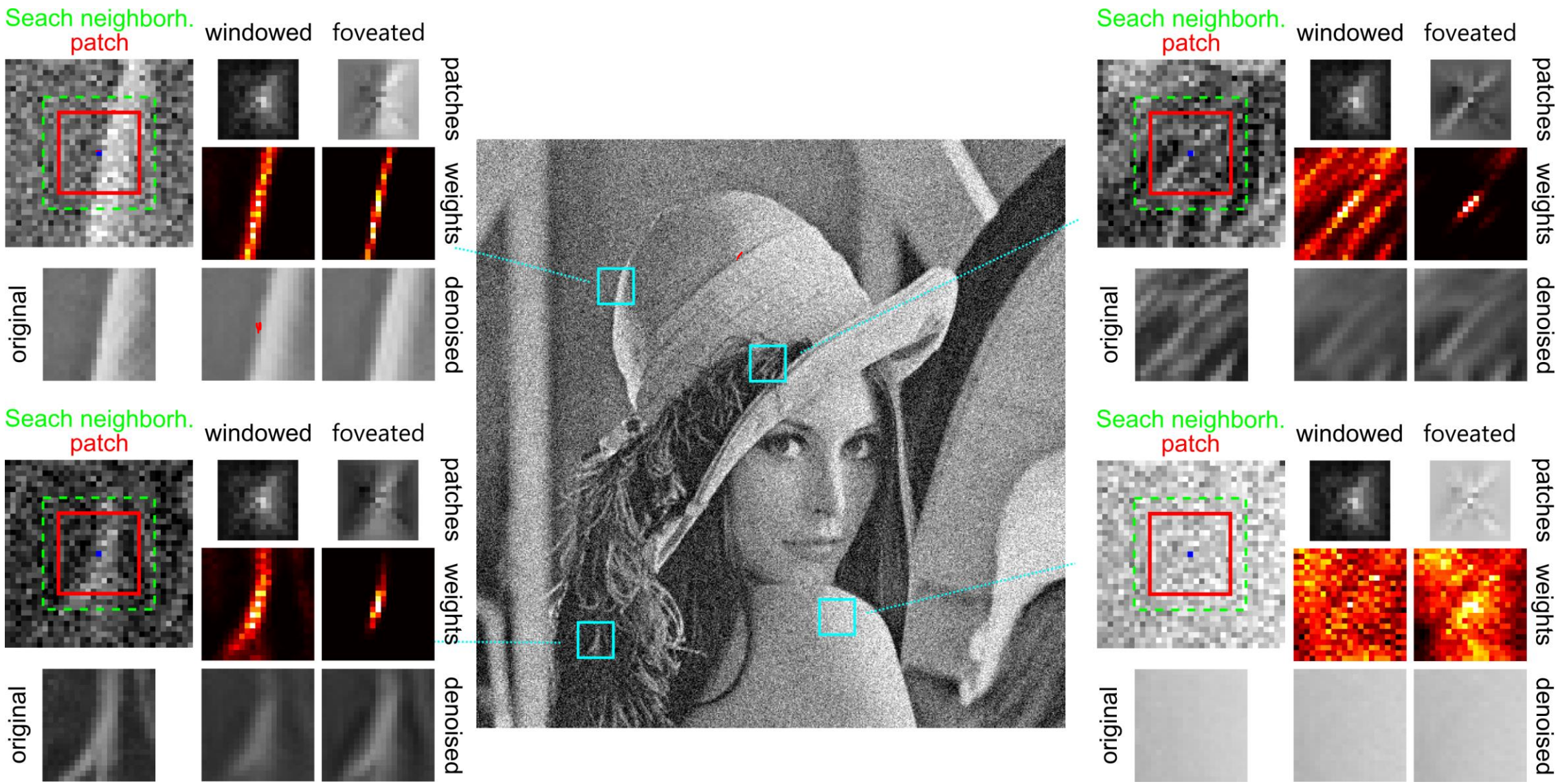
Weights in Nonlocal Means

Selected pixel Weights from neigh.



Buades, Antoni, Bartomeu Coll, and J-M. Morel. *A non-local algorithm for image denoising,* CVPR 2005

Weights from Patch-distances



Foi, A., & Boracchi, G. Foveated nonlocal self-similarity. *International Journal of Computer Vision*, 2016 *120*(1), 78-110.

Attention Mechanisms and Nonlocal Blocks

Giacomo Boracchi

Non-local Neural Networks

Non-local Neural Networks

Xiaolong Wang[1,2*] Ross Girshick[2] Abhinav Gupta[1] Kaiming He[2]

[1]Carnegie Mellon University [2]Facebook AI Research

Wang, X., Girshick, R., Gupta, A., & He, K. Non-local neural networks CVPR 2018

Non-local Neural Networks

In images, long-distance dependences are learned by stacking many convolutional layers.

Increasing the receptive field is not simple:

- Computationally inefficient
- Networks becomes difficult to train

New non-local operation in CNN that is inspired to NonLocal Means Denoising.

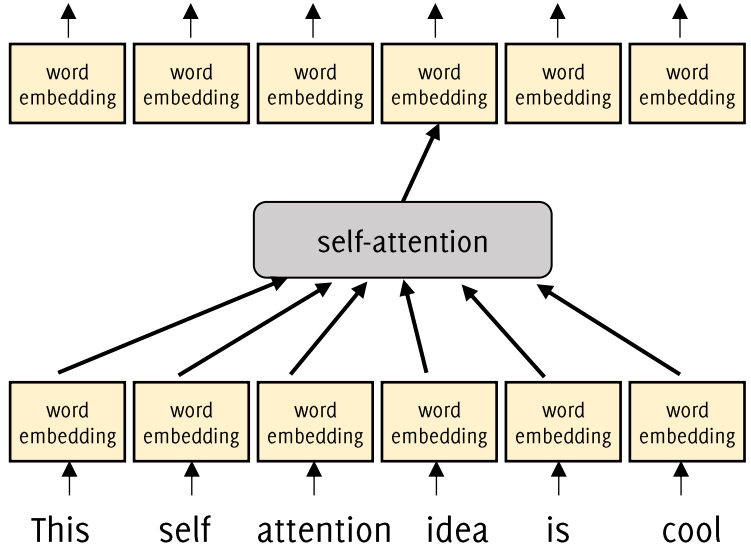
Compute a weighted sum of all the features in the input activation map

Wang, X., Girshick, R., Gupta, A., & He, K. Non-local neural networks CVPR 2018

Self Attention

In NLP, self attention is a mechanism for:

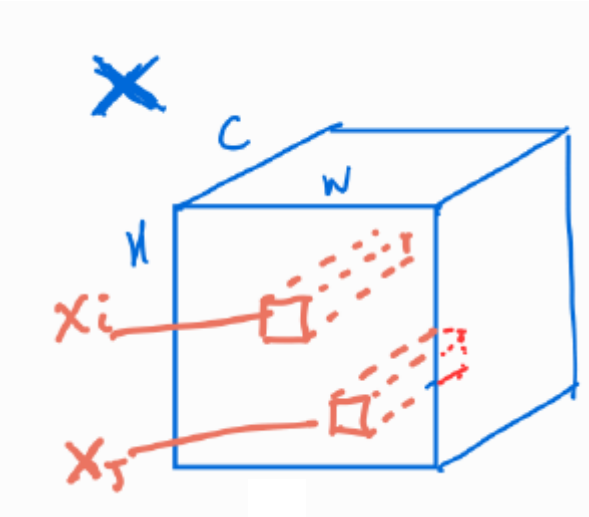
- Combining all the input word embeddings to produce new word embedding
- Each high-level embedding is **weighted average** of word embeddings provided as input to the layer



Non-local Blocks bring this mechanism to neural networks designed for processing images and videos.

Rather than word embedding, self-attention will involve features of the same activation map.

Non local operator



Generic non-local operator

$$y_i = \frac{1}{\mathcal{C}(x_i)} \sum_j f(x_i, x_j) g(x_j)$$

where (the notation is NOT consistent with NLM)

- x_i and x_j are feature vectors at different spatial locations of an activation map. x_i is the reference, x_j spans the entire activation
- $f(x_i, x_j)$ is a scalar distance between the two activations
- $g(x_j)$ is an embedding of the feature map x_j
- $\mathcal{C}(x_i)$ is a normalization factor

$$\mathcal{C}(x_i) = \sum_j f(x_i, x_j)$$

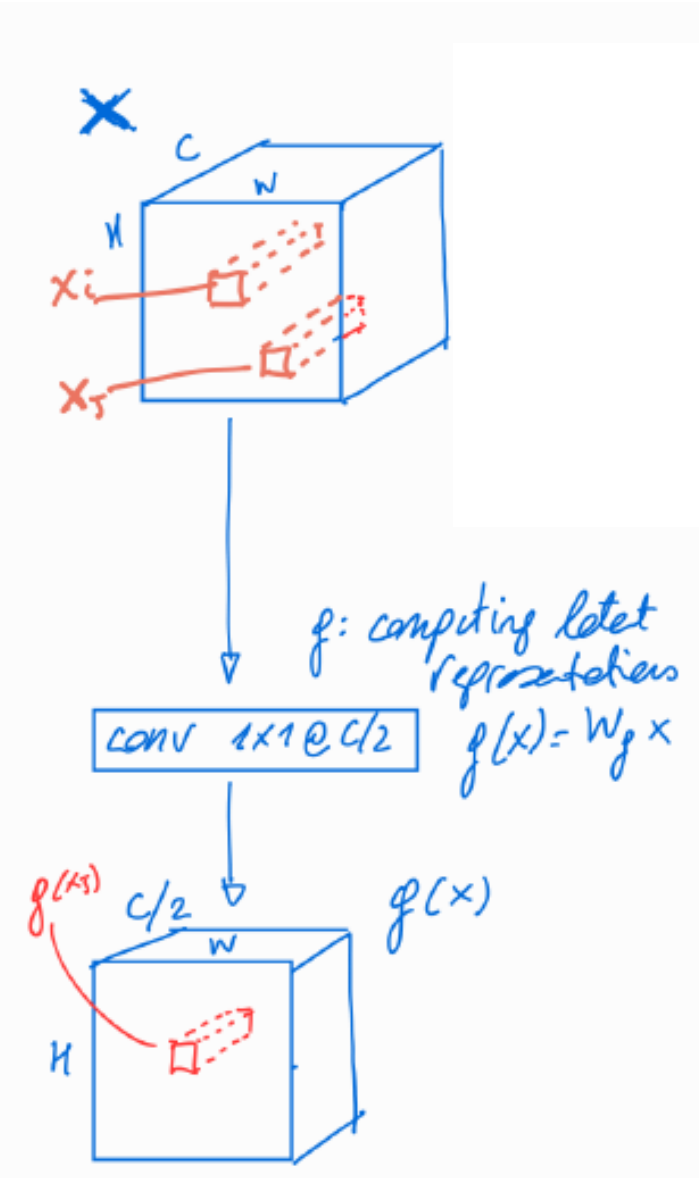
Wang, X., Girshick, R., Gupta, A., & He, K. Non-local neural networks CVPR 2018

The Embeddings $\frac{1}{\mathcal{C}(\boldsymbol{x_i})}\sum_j f(\boldsymbol{x_i}, \boldsymbol{x_j})\boldsymbol{g(x_j)}$

All the embeddings are typically computed by linear transformations

$$g(\boldsymbol{x_j}) = W_g \boldsymbol{x_j}$$

For the sake of efficiency, **embedding is computed by are 1x1 conv layers with C/2 filters**, to halve the number of channels ($W_\theta \in \mathbb{R}^{C/2 \times C}$)



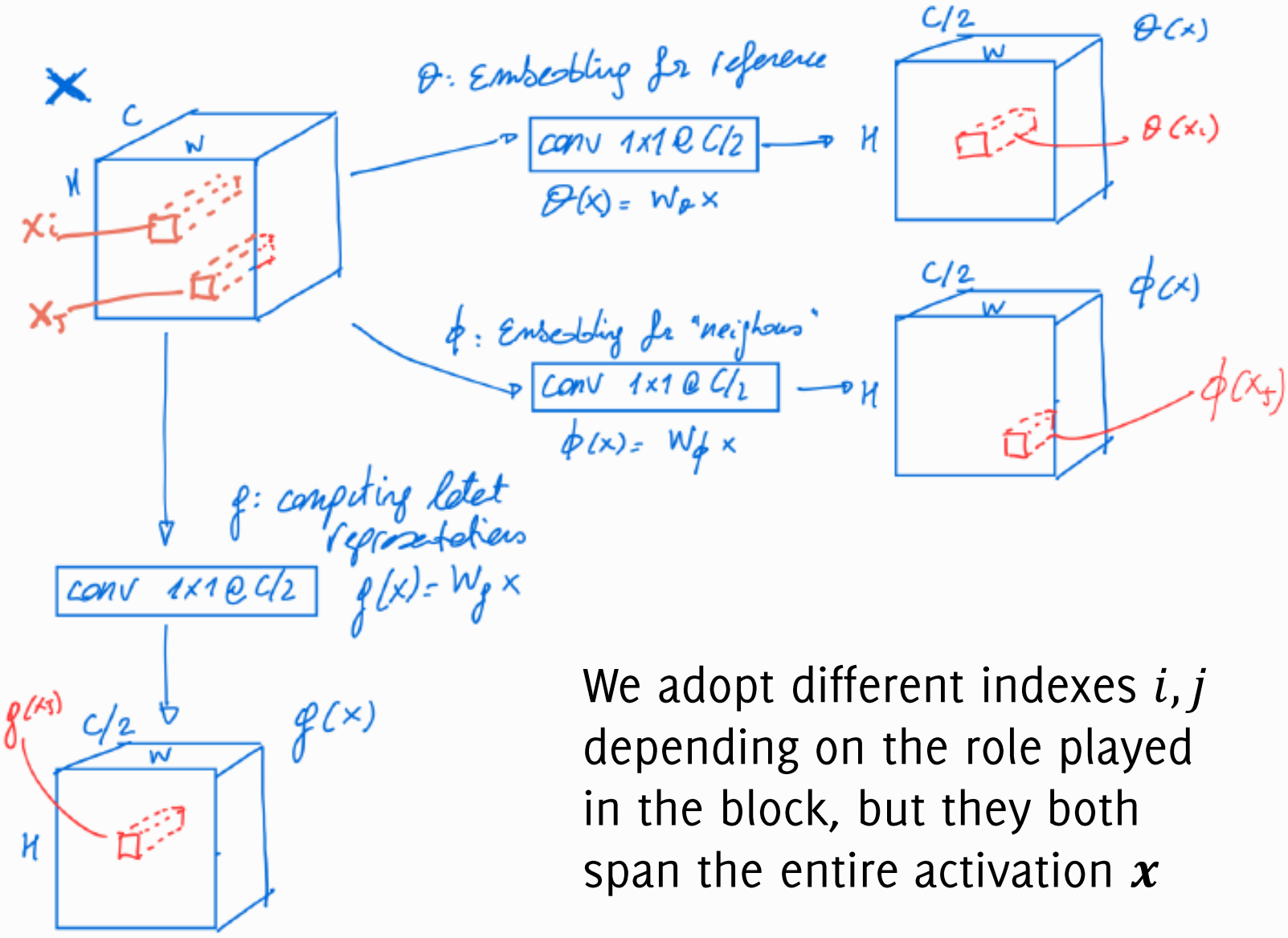
We adopt different indexes i, j depending on the role played in the block, but they both span the entire activation $\boldsymbol{x}$

The Embeddings $\frac{1}{\mathcal{C}(\boldsymbol{x_i})}\sum_j f(\boldsymbol{x_i}, \boldsymbol{x_j})g(\boldsymbol{x_j})$

All the embeddings are typically computed by linear transformations

$$\theta(\boldsymbol{x_i}) = W_\theta \boldsymbol{x_i}$$
$$\phi(\boldsymbol{x_j}) = W_\phi \boldsymbol{x_j}$$
$$g(\boldsymbol{x_j}) = W_g \boldsymbol{x_j}$$

For the sake of efficiency, these are 1x1 conv layers with C/2 filters, to halve the number of channels ($W_\theta \in \mathbb{R}^{C/2 \times C}$)



We adopt different indexes i, j depending on the role played in the block, but they both span the entire activation $\boldsymbol{x}$

The distance $\frac{1}{\mathcal{C}(x_i)}\sum_j f(x_i, x_j)g(x_j)$

$$y_i = \frac{1}{\mathcal{C}(x_i)}\sum_j f(x_i, x_j)g(x_j)$$

This has to be a scalar function, here are a few examples

Dot Product:

$$f(x_i, x_j) = x_i^T x_j$$

Gaussian:

$$f(x_i, x_j) = e^{x_i^T x_j}$$

Embedded Gaussian:

$$f(x_i, x_j) = e^{x_i^T W_\theta^T W_\phi x_j}$$

Concatenation/Dense:

$$f(x_i, x_j) = ReLU(w_f^T \cdot [x_i W_\theta, W_\phi x_j])$$

Wang, X., Girshick, R., Gupta, A., & He, K. Non-local neural networks CVPR 2018

The distance $\frac{1}{\mathcal{C}(x_i)}\sum_j f(x_i, x_j)g(x_j)$

$$y_i = \frac{1}{\mathcal{C}(x_i)}\sum_j f(x_i, x_j)g(x_j)$$

This has to be a scalar function, here are a few examples

Dot Product:

$$f(x_i, x_j) = x_i^T x_j$$

Gaussian:

$$f(x_i, x_j) = e^{x_i^T x_j}$$

Embedded Gaussian:

$$f(x_i, x_j) = e^{x_i^T W_\theta^T W_\phi x_j}$$

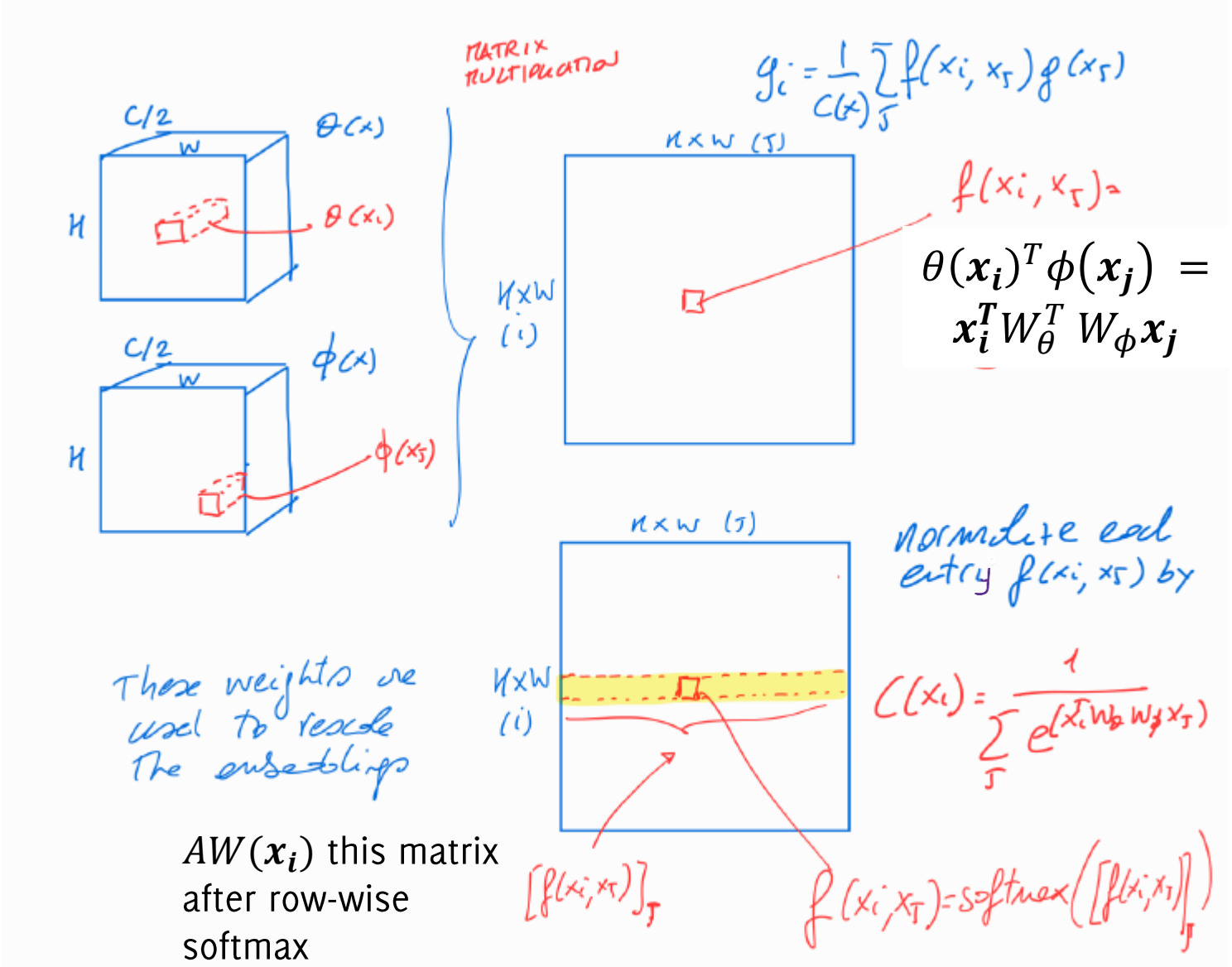
Concatenation/Dense:

$$f(x_i, x_j) = ReLU(w_f^T \cdot [x_i W_\theta, W_\phi x_j])$$

Wang, X., Girshick, R., Gupta, A., & He, K. Non-local neural networks CVPR 2018

The Self-Attention Weights: values of $f(\boldsymbol{x_i}, \boldsymbol{x_j})$

In case of embedded Gaussians, the weights given by $f(\boldsymbol{x_i}, \boldsymbol{x_j})$ are computed by matrix multiplication.
Exponential + Normalization can be performed by a softmax layer over each rows of the weight matrix.



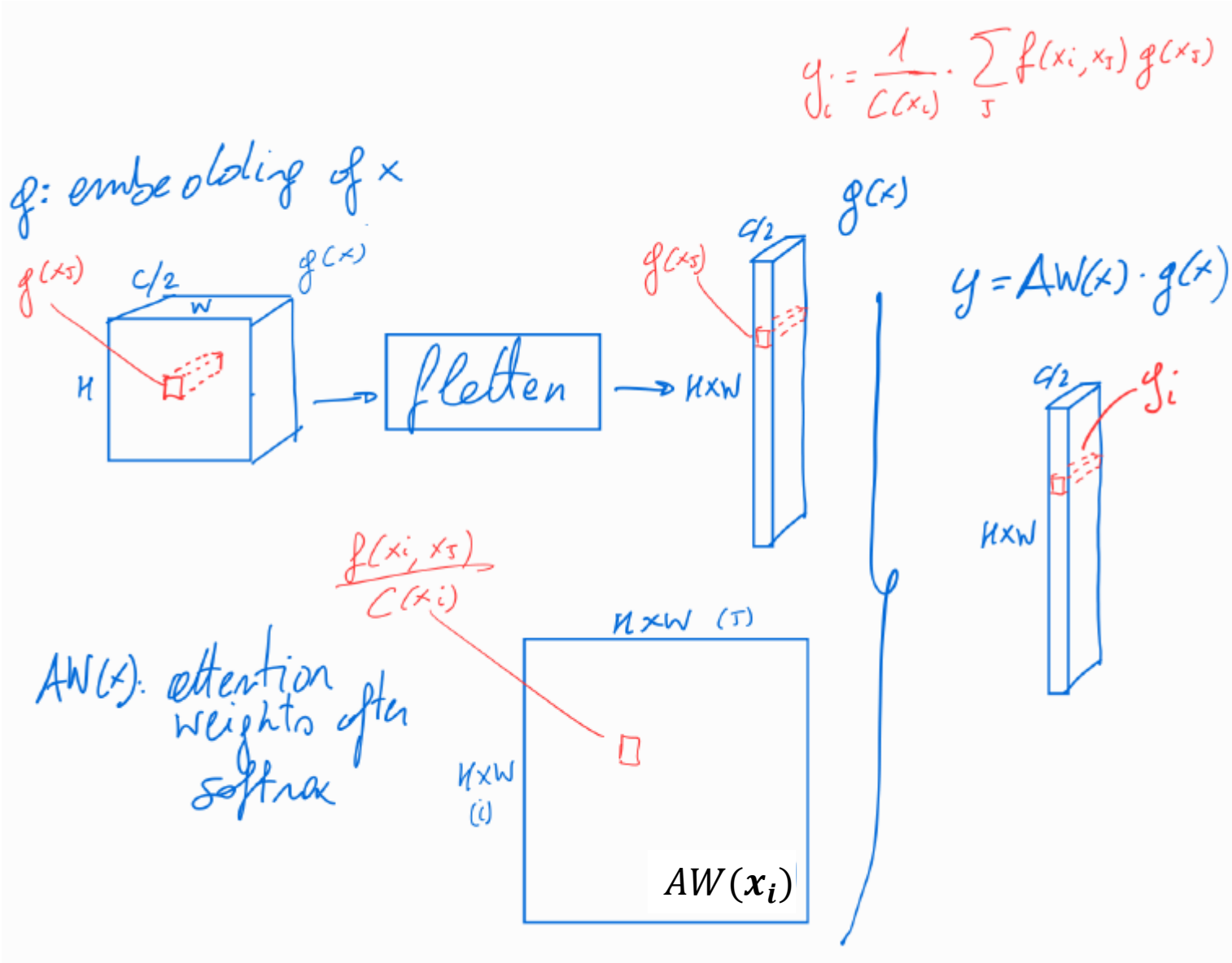
$AW(\boldsymbol{x_i})$ this matrix after row-wise softmax

Applying the self-attention

The self-attention is obtained by multiplying (tensor multiplication) the embeddings $g(\boldsymbol{x_j})$ against the obtained weights.

The output vector $\boldsymbol{y_i}$ corresponds to a weighted sum of all the embeddings $g(\boldsymbol{x_j})$ where the weights depend on $g(\boldsymbol{x_i})$.

By tensor multiplication we compute the output y_i in the entire activation map



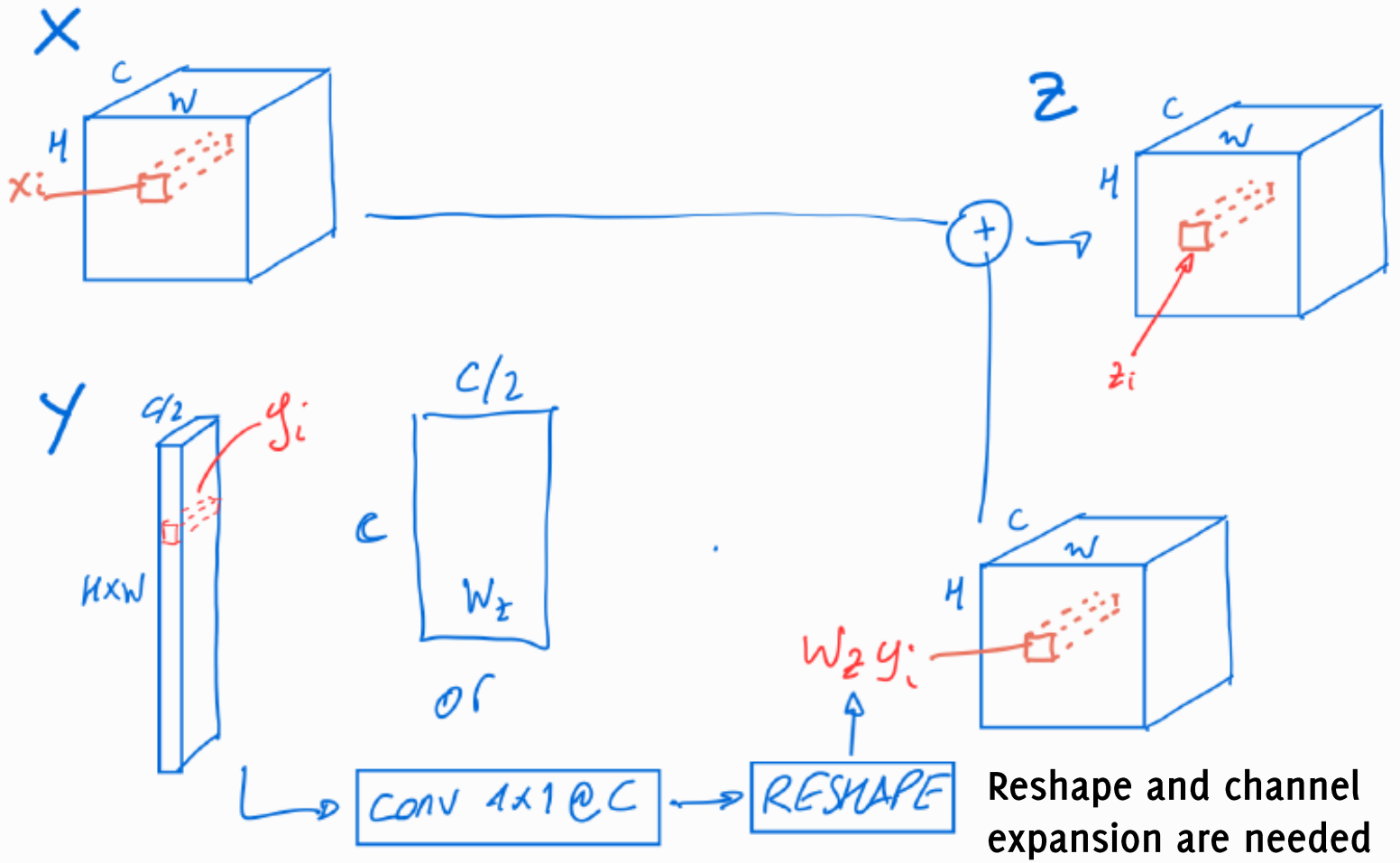
The Non-local Block

The output of the non-local block is not $\boldsymbol{y}$ itself, but this is **used with skip connections**

$$\boldsymbol{z_i} = W_z \boldsymbol{y_i} + \boldsymbol{x_i}$$

W_z is also implemented as a convolutional layer with 1x1 convolutions.

This makes **simpler to introduce the block** in a **pretrained model** (initializing W_z to zeros)



Reshape and channel expansion are needed for the residual connection

Efficiency notes

The **number of cannels C** in the embedding is **halved** to reduce computations.

Similarly, it is possible to **perform spatial pooling** on $\phi(x_j)$ and $g(x_j)$, to reduce the terms in the sum

$$\boldsymbol{y_i} = \frac{1}{\mathcal{C}(\boldsymbol{x_i})} \sum_{\hat{j}} f\left(\boldsymbol{x_i}, \boldsymbol{x_{\hat{j}}}\right) g(\boldsymbol{x_{\hat{j}}})$$

This give rise to **rectangular (vertical) weight matrix** $AW(\boldsymbol{x})$.

Still, $\boldsymbol{y_i}$ is defined in every location of input activation $i = 1, \ldots, H \times W$ but the weights depend on a subset of spatial locations (e.g., $j = [1 : n : \ H \times W]$).

Wang, X., Girshick, R., Gupta, A., & He, K. Non-local neural networks CVPR 2018

Video Classification Models

The Non-local block is seamlessly inserted in 3D CNN to perform video classification (as illustrated here), leveraging nonlocality in time.

«Nevertheless, our method surpasses all the existing RGB or RGB + flow based methods by a good margin. Without using optical flow and without any bells and whistles, our method is on par with the heavily engineered results of the 2017 competition winner»

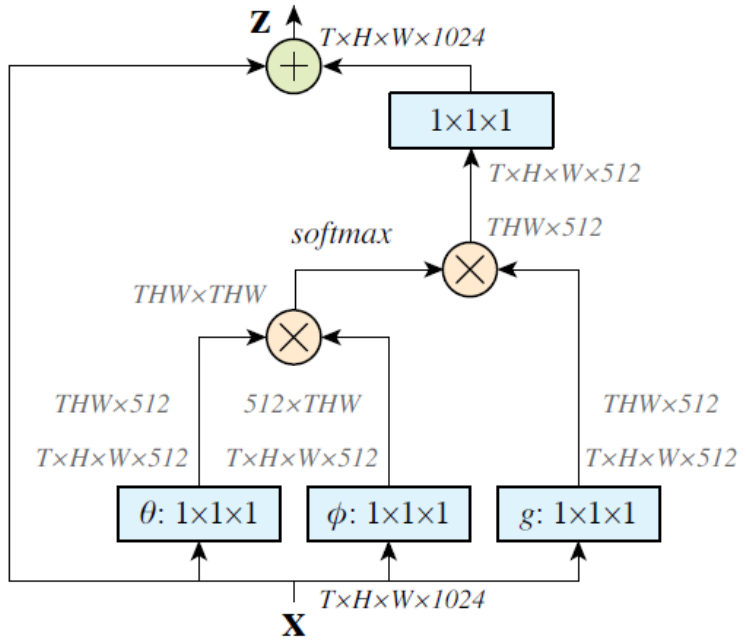


Figure 2. A spacetime **non-local block**. The feature maps are shown as the shape of their tensors, *e.g.*, $T \times H \times W \times 1024$ for 1024 channels (proper reshaping is performed when noted). "$\otimes$" denotes matrix multiplication, and "$\oplus$" denotes element-wise sum. The softmax operation is performed on each row. The blue boxes denote $1 \times 1 \times 1$ convolutions. Here we show the embedded Gaussian version, with a bottleneck of 512 channels. The vanilla Gaussian version can be done by removing θ and ϕ, and the dot-product version can be done by replacing softmax with scaling by $1/N$.

Wang, X., Girshick, R., Gupta, A., & He, K. Non-local neural networks CVPR 2018

Vision Transformers

Giacomo Boracchi

Vision Transformers (ViT)

Published as a conference paper at ICLR 2021

AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

Alexey Dosovitskiy[*,†], **Lucas Beyer**[*], **Alexander Kolesnikov**[*], **Dirk Weissenborn**[*], **Xiaohua Zhai**[*], **Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby**[*,†]

[*]equal technical contribution, [†]equal advising

Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale*. ICLR 2021.

Background

Self-attention-based architectures, in particular **Transformers**, have become the **model of choice in NLP**.

Dominant approach:

1. **train** huge models on a **large corpus**,
2. **fine tune** on a specific task

Transformers achieve SoA

In **Visual Recognition**, **CNNs remain dominant** with SoA achieved by **ResNet-like architecture.**

Vision Transformer (ViT)

- The first *pure transformer* architecture applied on images without relying on convolutional blocks.
- Non-overlapping patches are processed as tokens in text.
- The entire image is fed to a Transformer as a sequence of patch embeddings.

Vision Transformers (proudly) **lack the inductive biases (priors) inherent to CNNs**, namely the translation equivariance and locality!

Trained on **"mid-size" dataset** (e.g. Alexnet 1K), **ViT does not pair ResNet** performance.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale*. ICLR 2021.

Vision Transformer (ViT) and Large Scale Training

When trained or pre-trained on **very large datasets** (ImageNet 21K, JFT-300M)

- Outperformed CNN-based architectures.
- **Large scale training enable learning 2D patterns** of images even **without inductive biases**, namely a model that does not promote such a 2D structure.

The same paradigm as for text:

- training on a huge dataset,
- fine tuning for a specific task with fewer data

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale*. ICLR 2021.

Large Datasets (in 2021)

“Mid-size” datasets:

- Imagenet 1k (1K classes, 1M images). The one used for “Alexnet”

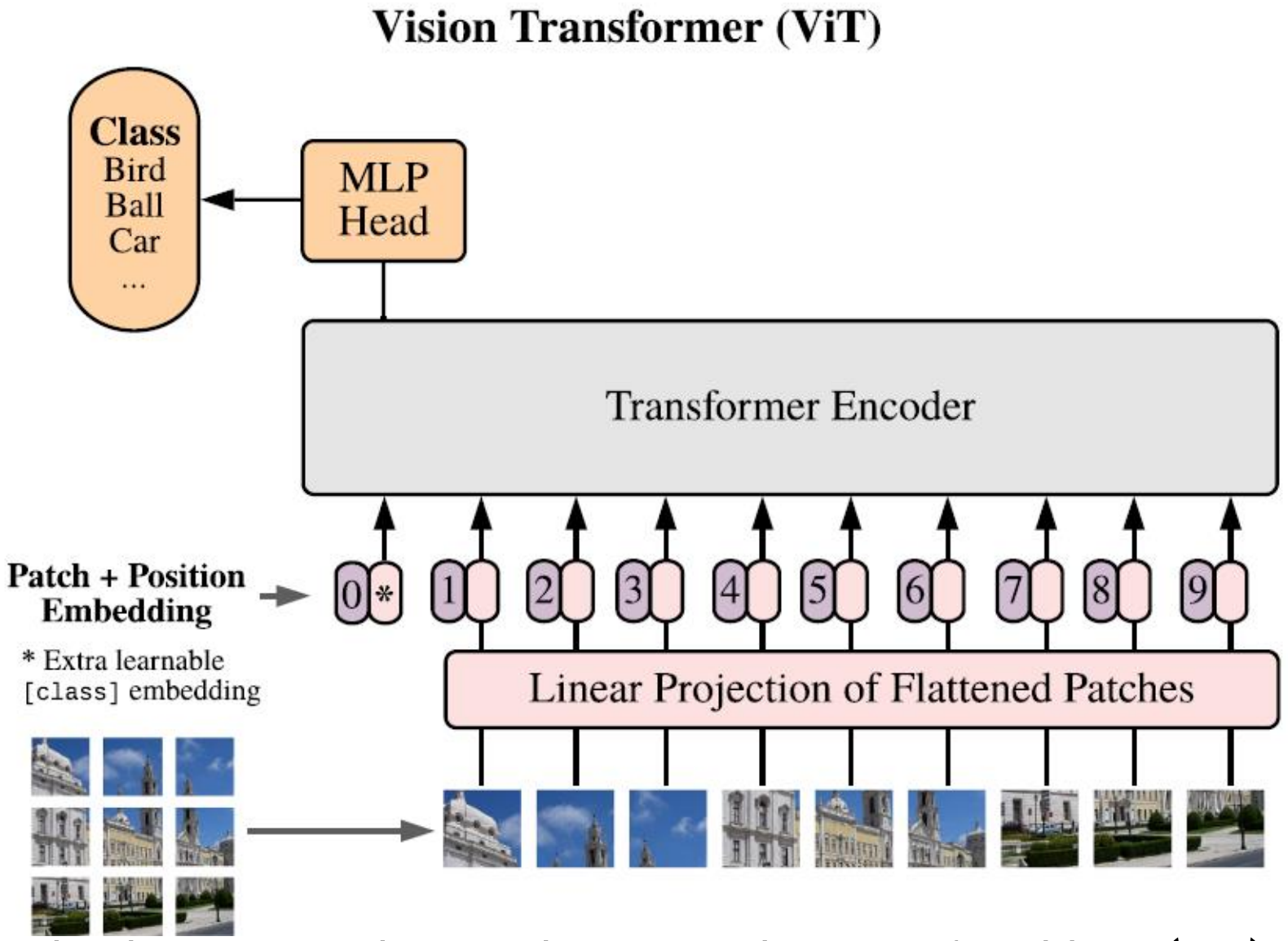
“Large” Datasets:

- Imagenet 21K (21K classes, 14M images a superset of Imagenet 1K)
- JFT-300M (18K classes, 303M images) is an **internal Google dataset** used for training image classification models. Over one billion labels for the 300M images (a single image can have multiple labels) out of which approximately 375M are selected via an algorithm.
- Typically images here are larger (384x384?)

Watch out: vectors are considered as rows in this paper

… and notation is not consistent at all with the rest of the slides

Vit Overview



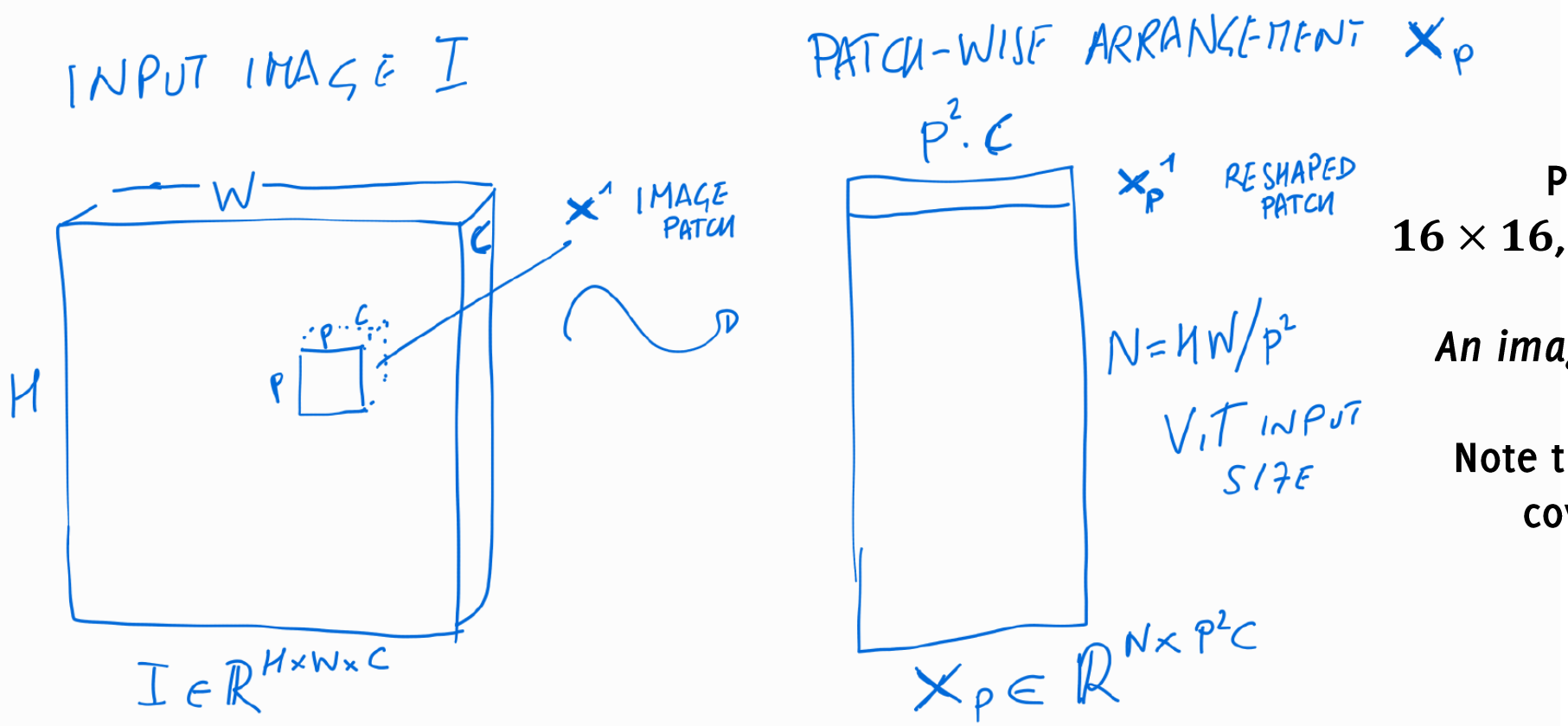
Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale.* ICLR 2021.

Patch-wise image arrangement

Images $I \in \mathbb{R}^{H \times W \times C}$ are divided in non-overlapping $P \times P$ patches

$$\boldsymbol{x^i} \in \mathbb{R}^{P \times P \times C}$$

Each patch is arranged in a row vector of a matrix $\boldsymbol{x_p} \in \mathbb{R}^{N \times P^2 C}$, being $N = HW/P^2$ the number of extracted patches



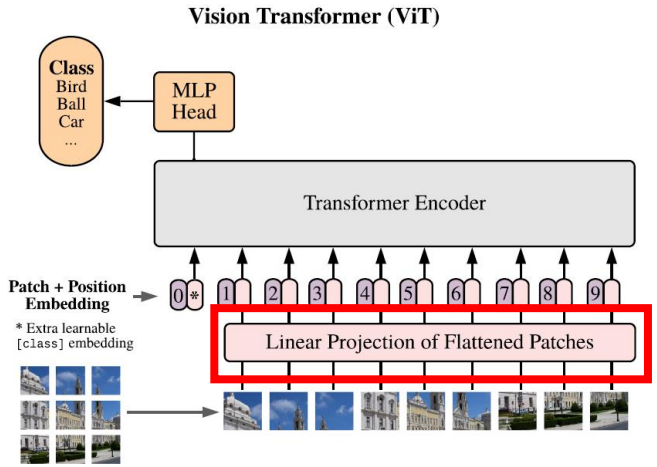
Patches are typically
16×16, images are $\approx 256 \times 256$
hence the title
An image is worth 16x16 words..

Note that each unfolded patch
covers all the channels

Patches are typically
16×16, images are $\approx 256 \times 256$
hence the title
An image is worth 16x16 words..

Note that each unfolded patch
covers all the channels

Linear Projection of Flattened Patches



The i-th patch is embedded in a D-dimensional vector by a learnable linear mapping (remember these are row vectors)

$$\boldsymbol{x_p^i \rightarrow x_p^i E}$$

where

$$E \in \mathbb{R}^{3P^2 \times D}$$

$x_p^1 \longmapsto z^1 = x_p^1 E \in \mathbb{R}^D$ $\quad E \in \mathbb{R}^{P^2 C \times D}$

This embedding is repeated for each and every patch

Embedding the entire image

The image is embedded into a matrix

$$\boldsymbol{x_p} \in \mathbb{R}^{N \times P^2 C}$$

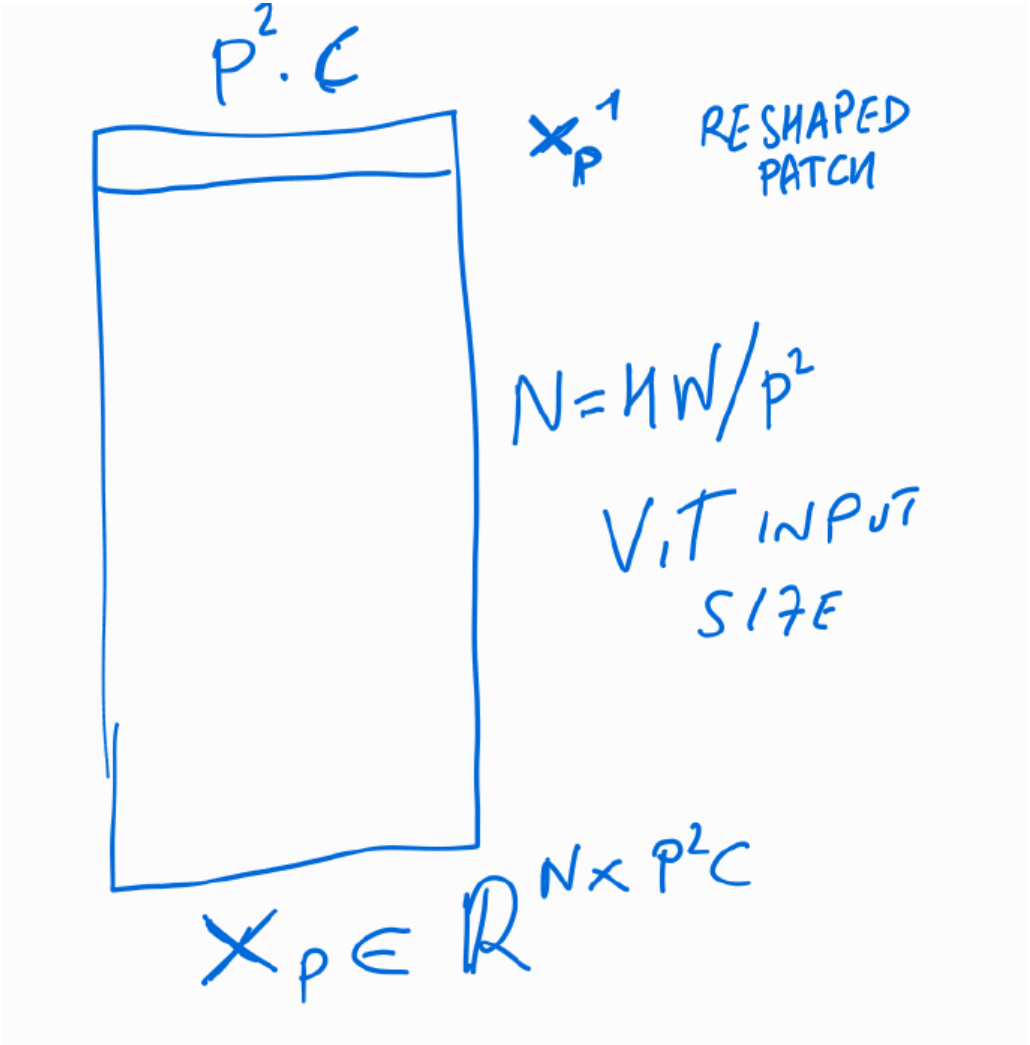
Where $N = HW/P^2$ the number of patches.

N is equivalent to the **size of a sentence** fed to the transformer and can in principle vary.

N is set to a large **value to be considered as the maximum length**, and in case of shorter sentences this is padded with zero.

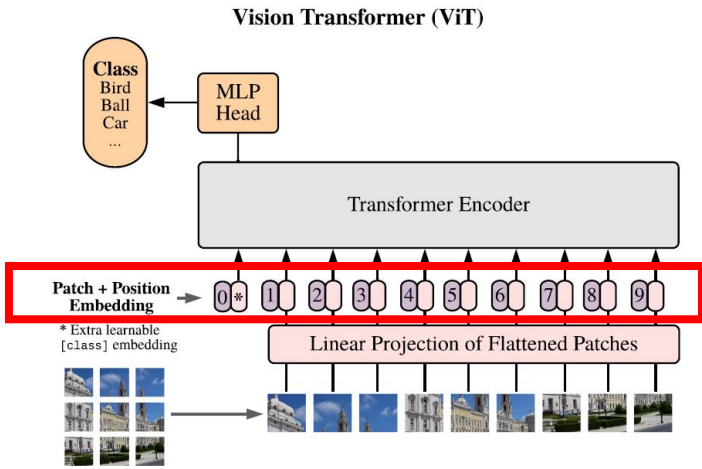
Rmks:

- the coefficients of the transformers are not influenced by the input lenght N
- no problem in processing larger images, it's like a larger text



Positional Encoding

The positional encoding is *added* to their respective embedded patches



$$I \longmapsto z_0 = \begin{bmatrix} x_{class} \\ z_p^1 \\ \\ z_p^N \end{bmatrix}_{N+1} \times D + \begin{bmatrix} \\ E_{pos} \\ \\ \end{bmatrix} \times D$$

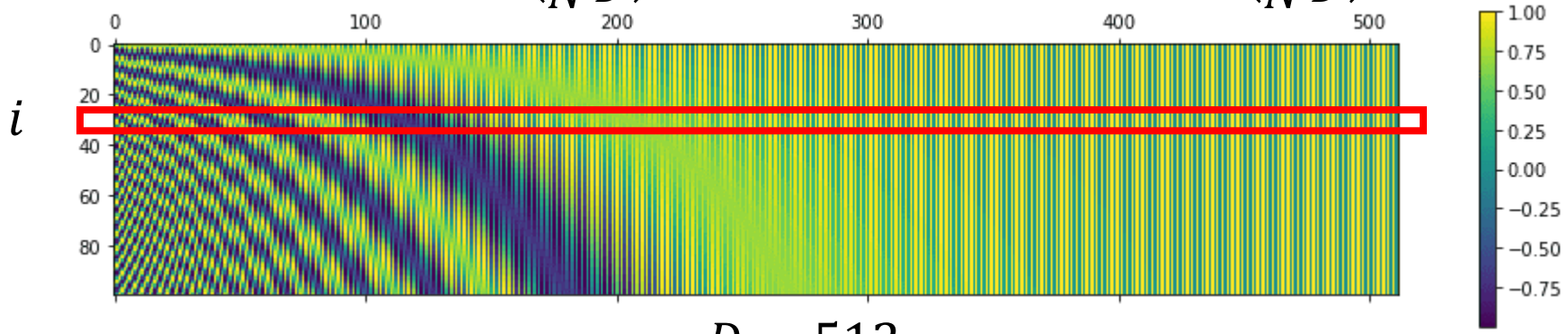
Positional Encoding

Since each patch in an image simultaneously flows through the Transformer, **the model itself doesn't have any sense of position/order for the patch in the image**.

We can enumerate the position of patches (1 dimensional ordering seems ok) or we can define a mapping from the i-th patch to a D-dimensional vector

$$[1:N] \to [-1,1]^D$$

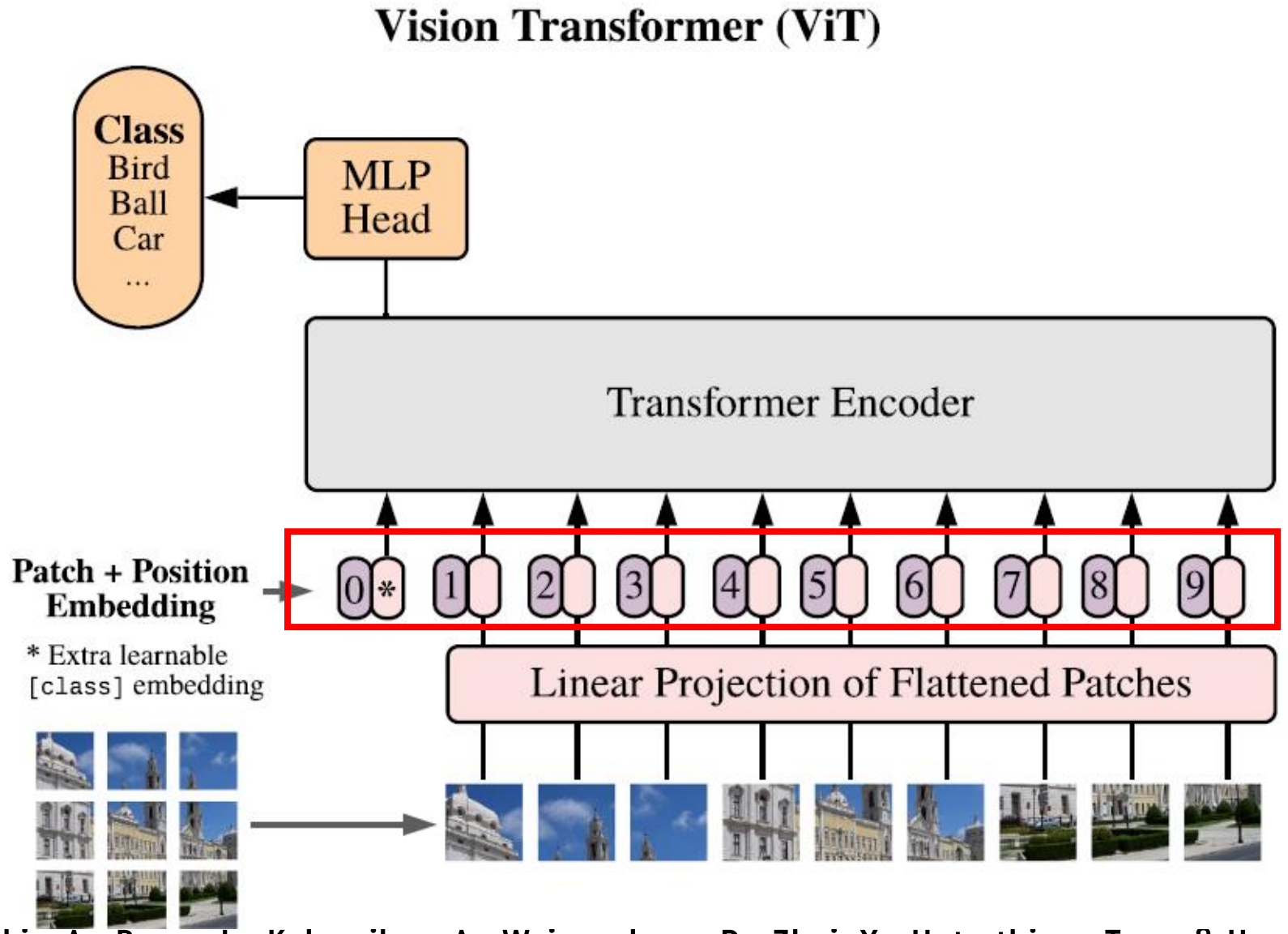
$$PE(i,2j) = \sin\left(\frac{i}{N^{\frac{2j}{D}}}\right), \qquad PE(i,2j+1) = \cos\left(\frac{i}{N^{\frac{2j}{D}}}\right)$$



$$D = 512$$

Or more in general the positional embedding can be **learned as a matrix product** (after such preliminary encoding in a D-dimensional vector)

ViT Overview



Here is where we are so far

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale.* ICLR 2021.

ViT Output

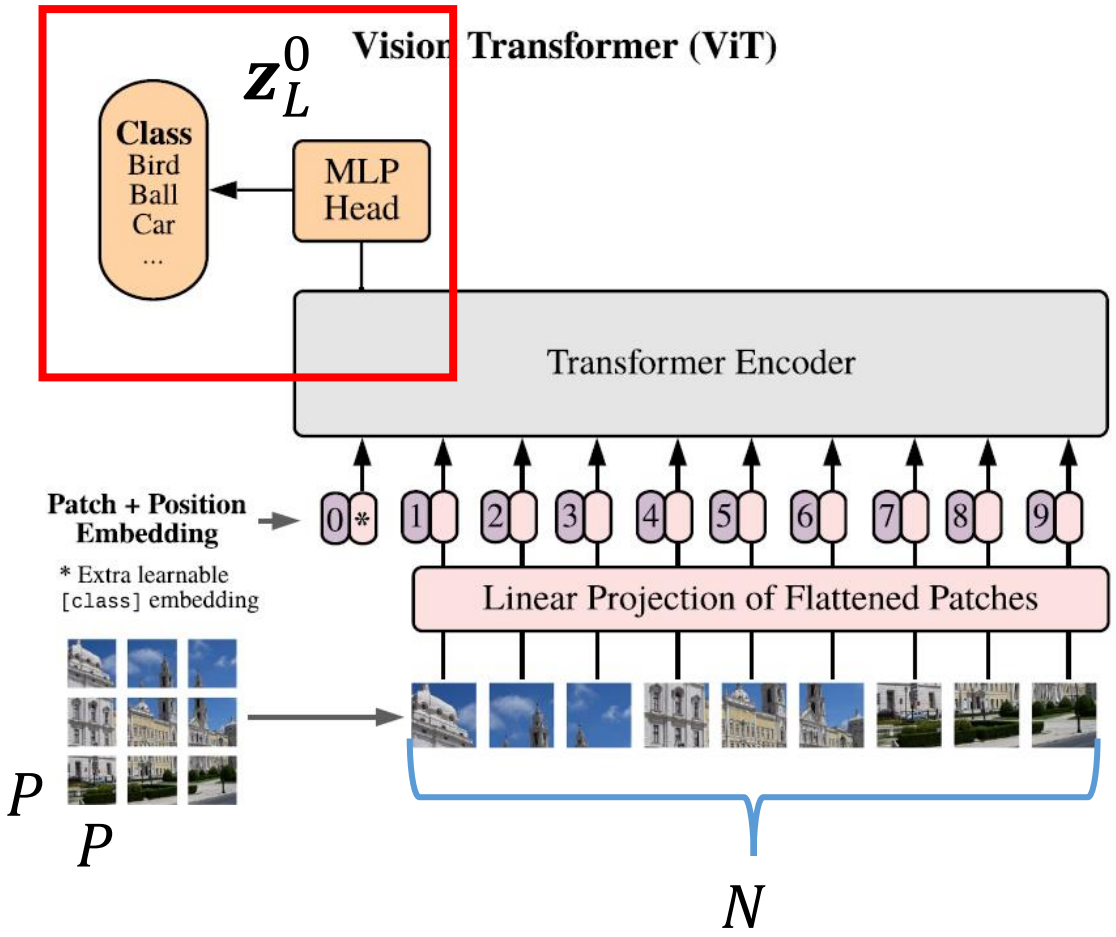
The entire image is embedded into

$$z_0 = [label; \dots; \dots] + E_{pos}$$

Where *label* is the label to be predicted.

Since ViT will be trained like BERT, the [class] embedding will be trained to **infer the missing word, corresponding to the image label.**

A classification head is attached to $\boldsymbol{z}_L^0$

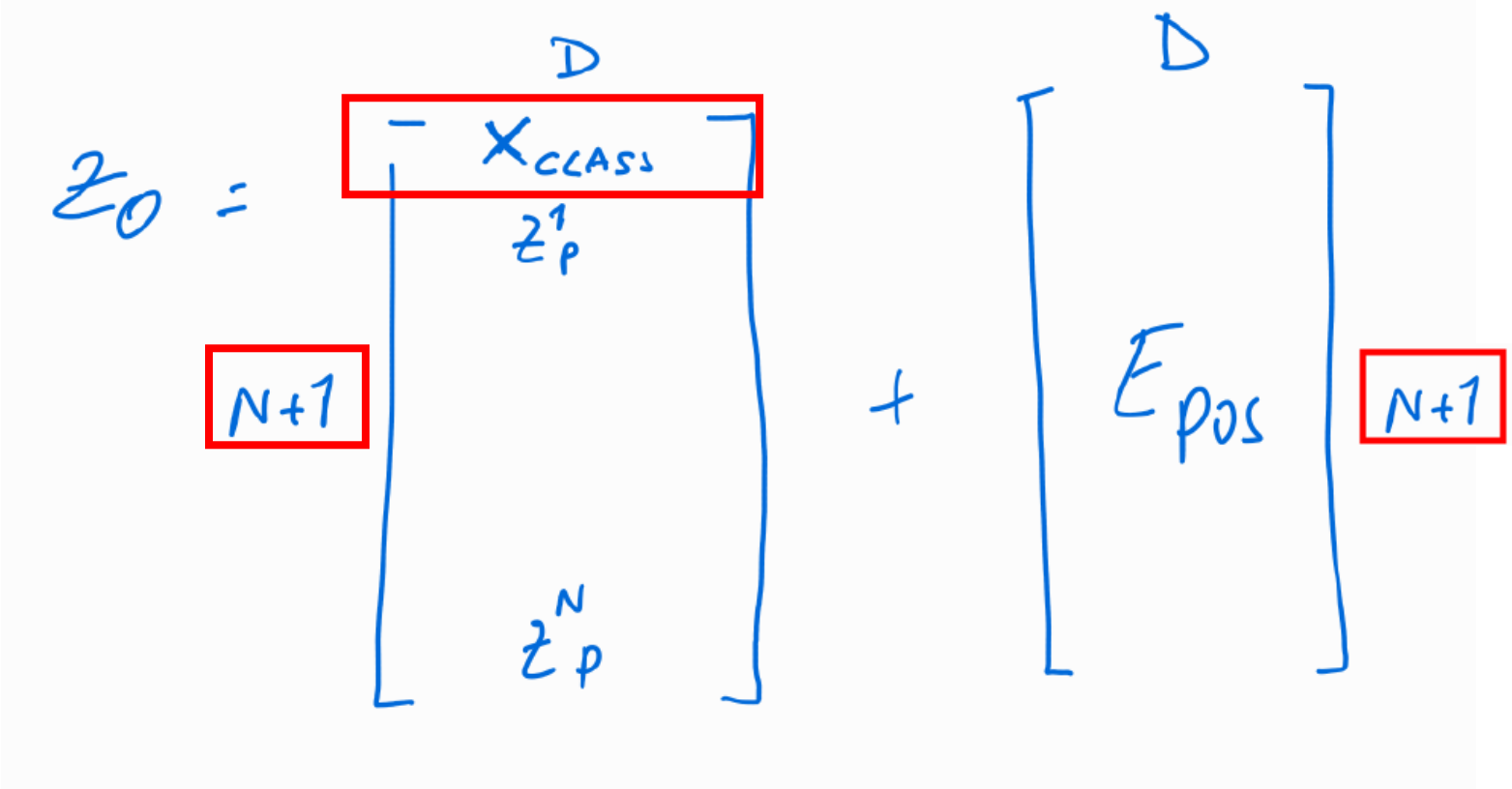


Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale*. ICLR 2021.

ViT Output

This is added as an additional learnable embedding in position 0

At the end of the Transformer's stack, this vector is fed to a MLP (possibly with an hidden layer) to predict image class

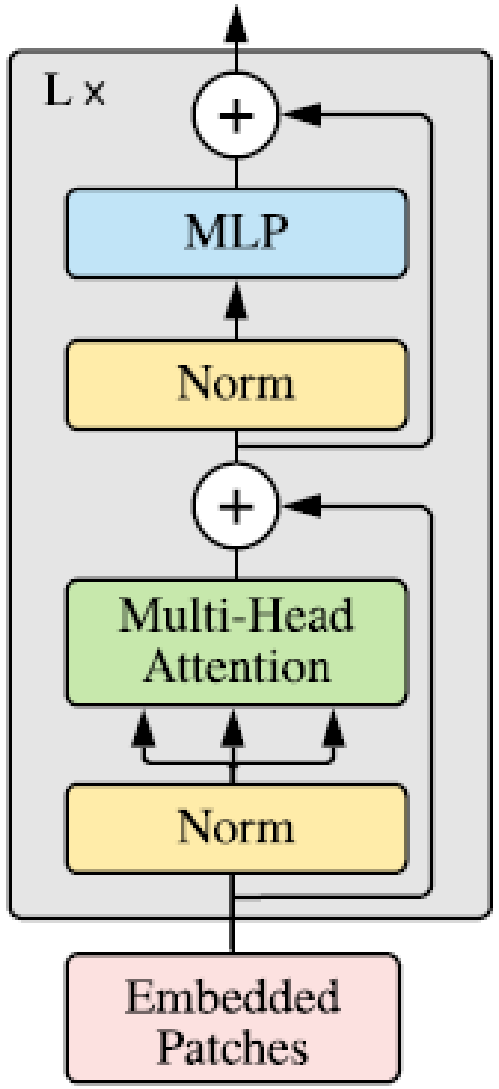


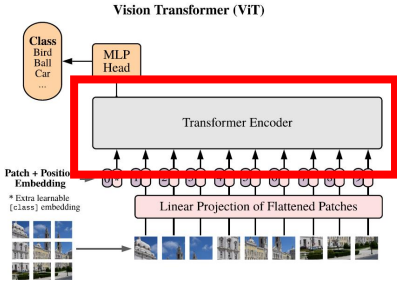
Inside the Transformer Block

It alternates between

- A Multiheaded Self Attention (MSA) layer
- A Multi-layer Perceptrons (MLP) layer
- Including a LayerNorm in betwewn

This a **plain vanilla transformer** as for text (only the input changes in ViT)





Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale*. ICLR 2021.

LayerNorm

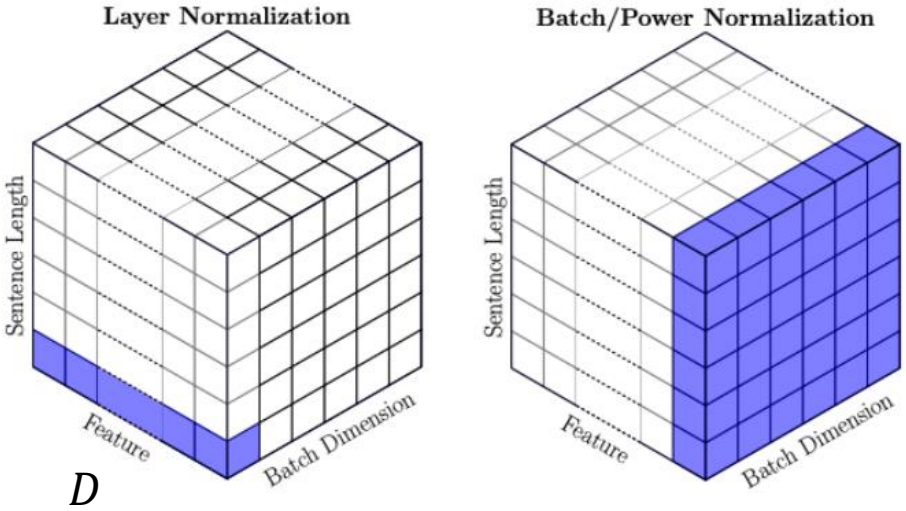
Standardize (like z-score subtracting the mean and dividing by the variance) **all the entries of each representation** (ignoring the batch-dimension).

Involves all the features $(i = 1, \dots, D)$ in the $l - th$ element for a single training sequence:

$$\mu^l = \frac{1}{D}\sum_{i=1}^{D} a_i^l \qquad \sigma^l = \sqrt{\frac{1}{D}\sum_{i=1}^{D}\left(a_i^l - \mu^l\right)^2}$$

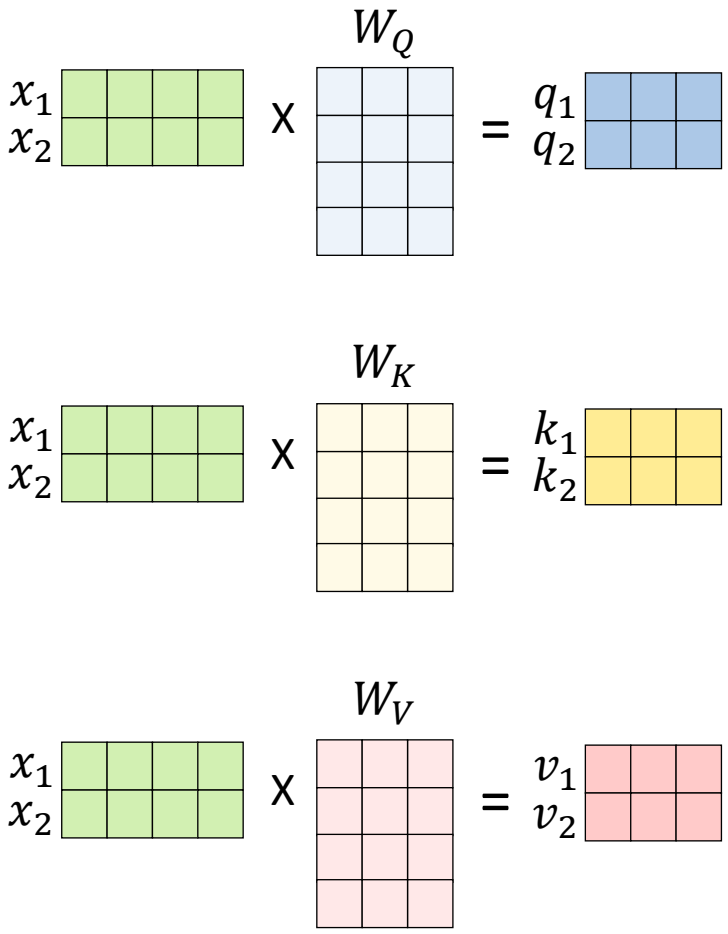
Like batch normalization, it is possible to learn bias and offset after that.

This is meant to **speed up training in R-NN and Transformer** architecture

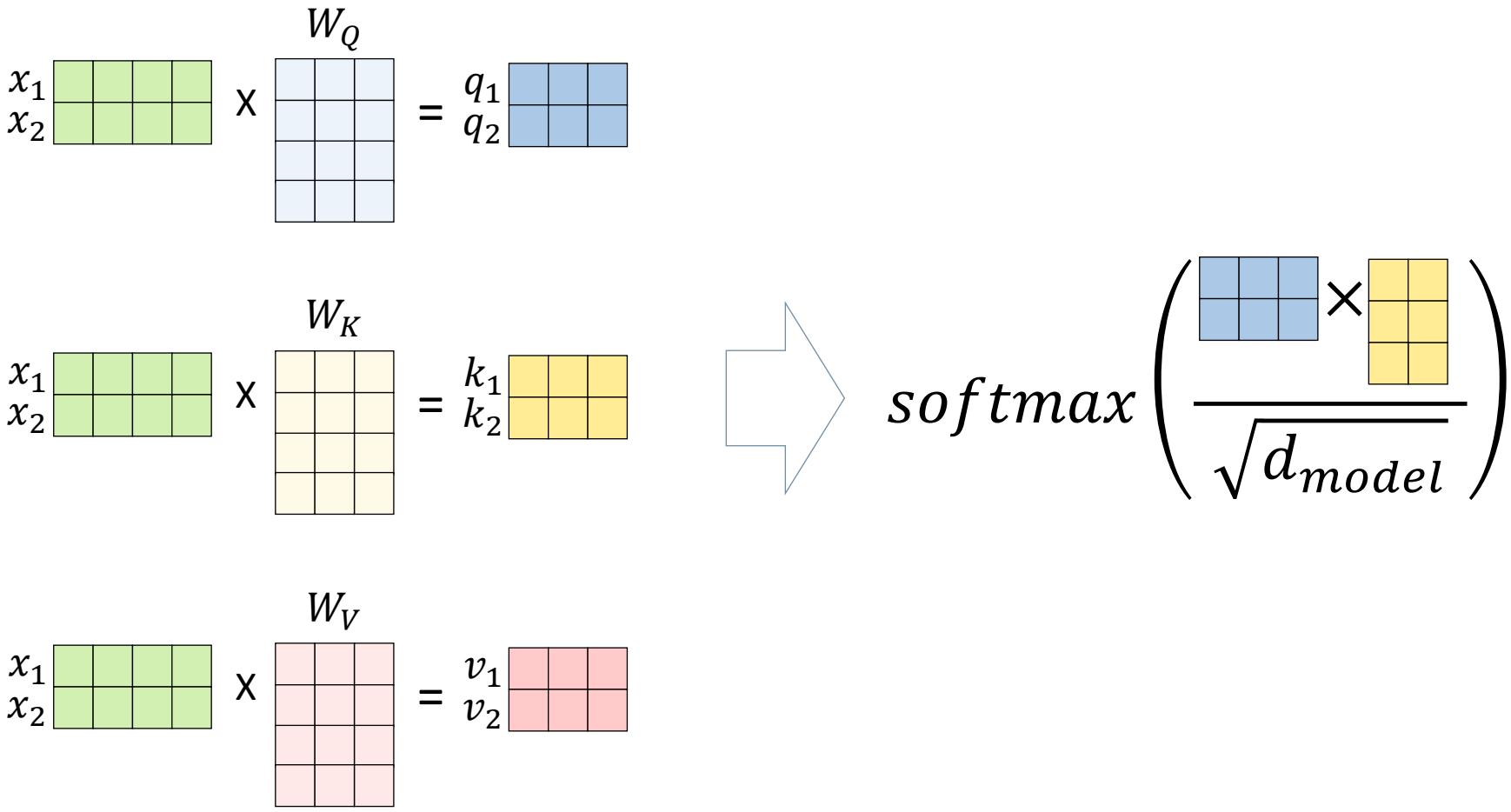


Ba, J. L., Kiros, J. R., & Hinton, G. E. (2016). Layer normalization. arXiv preprint arXiv:1607.06450.

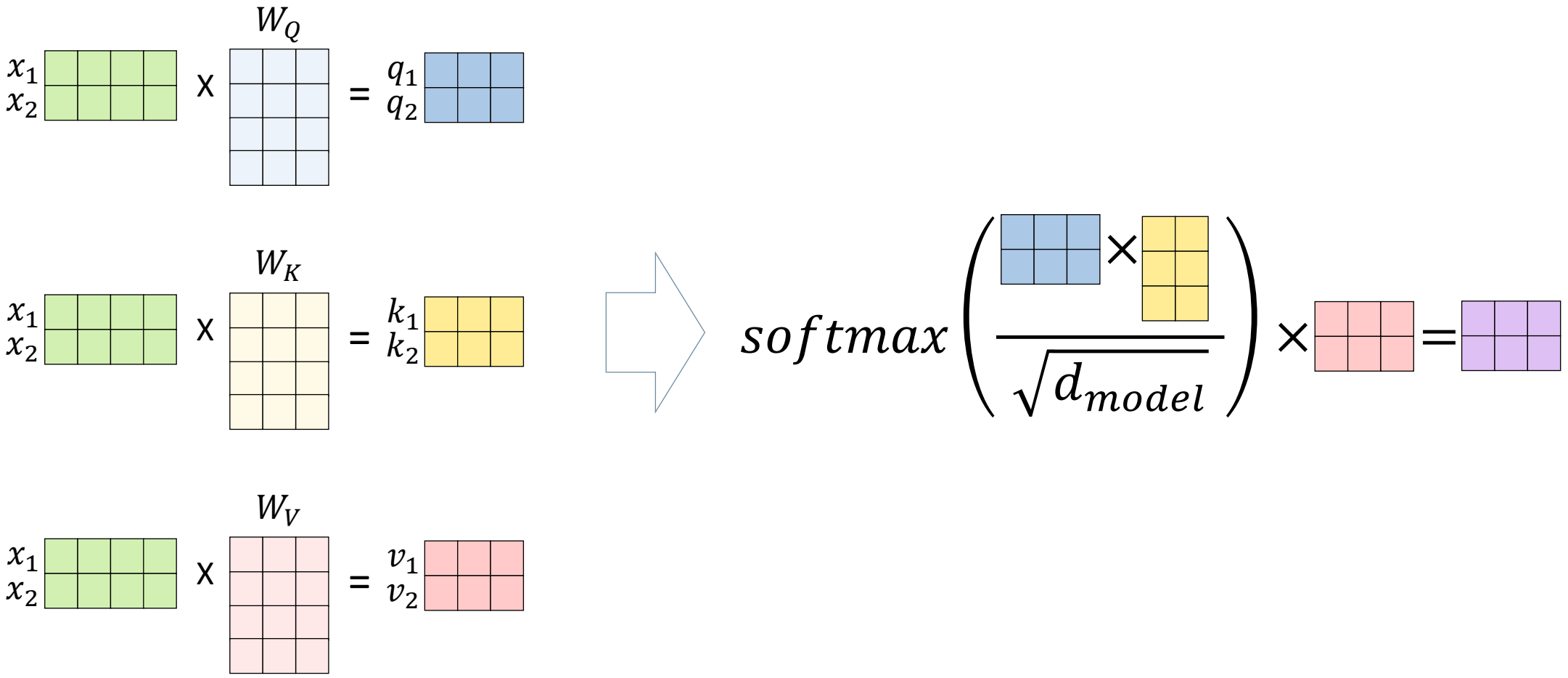
Self-Attention: Learnable Embeddings



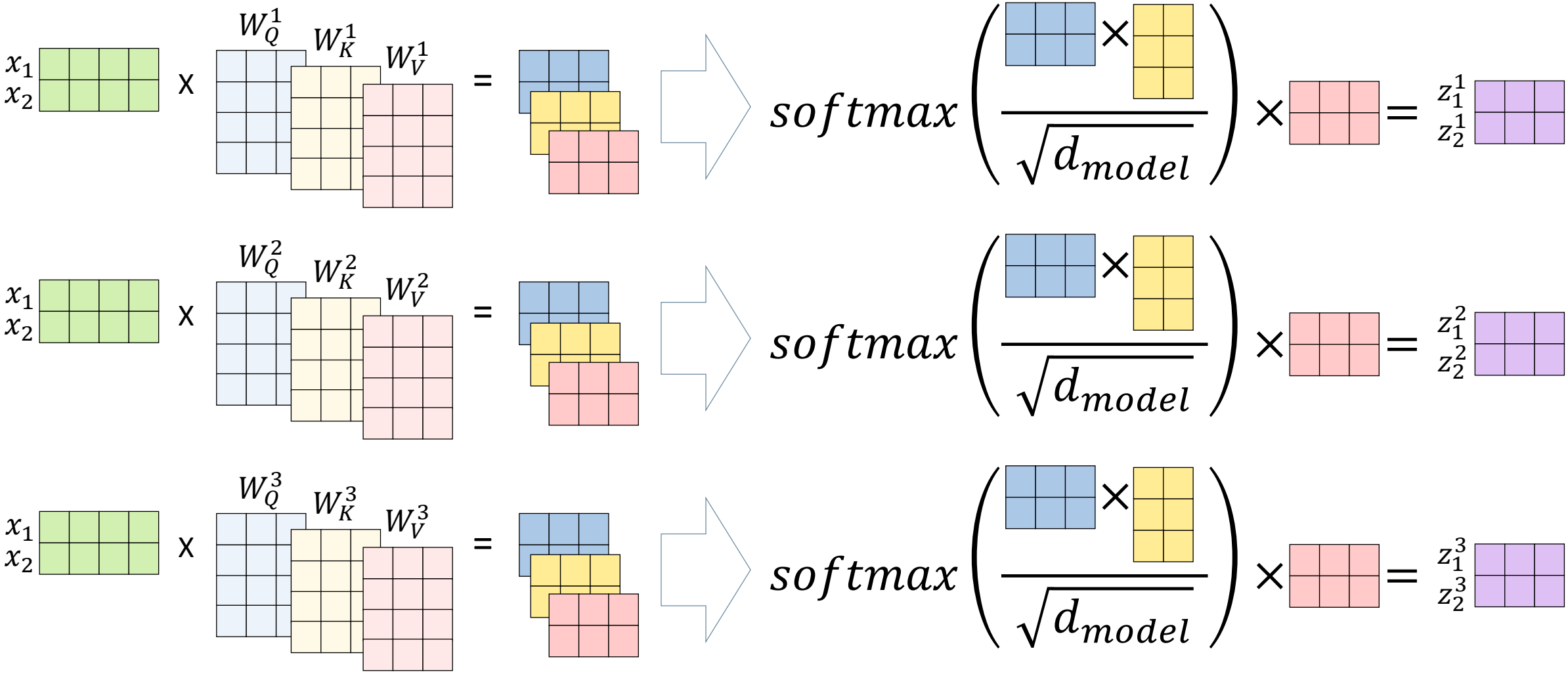
Self-Attention: The Self-Attention Weights



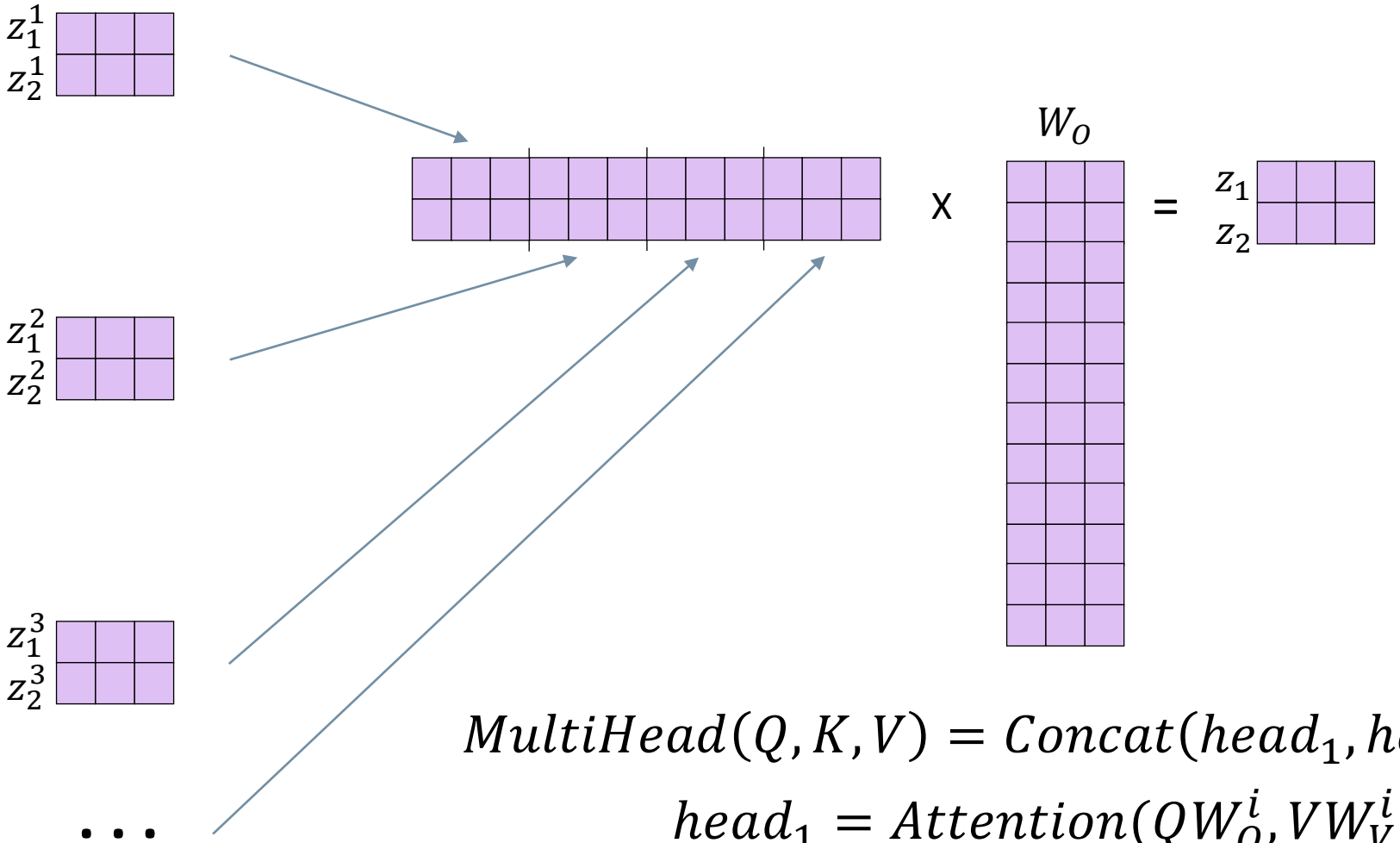
Self-Attention: Applying the Self-Attention



Multiheaded Self-Attention (MSA)



Multiheaded Self-Attention (MSA)



$$MultiHead(Q,K,V) = Concat(head_1, head_2, \dots, head_n)W_o$$

$$head_1 = Attention(QW_Q^i, VW_V^i, VW_V^i)$$

Summarizing The Transformer Block

These operation are executed in a sequence of L blocks.

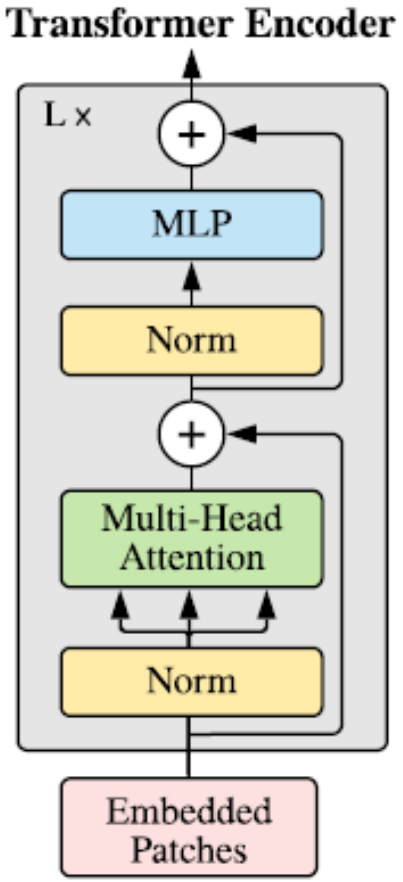
Each block contains residual connections

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

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

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

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

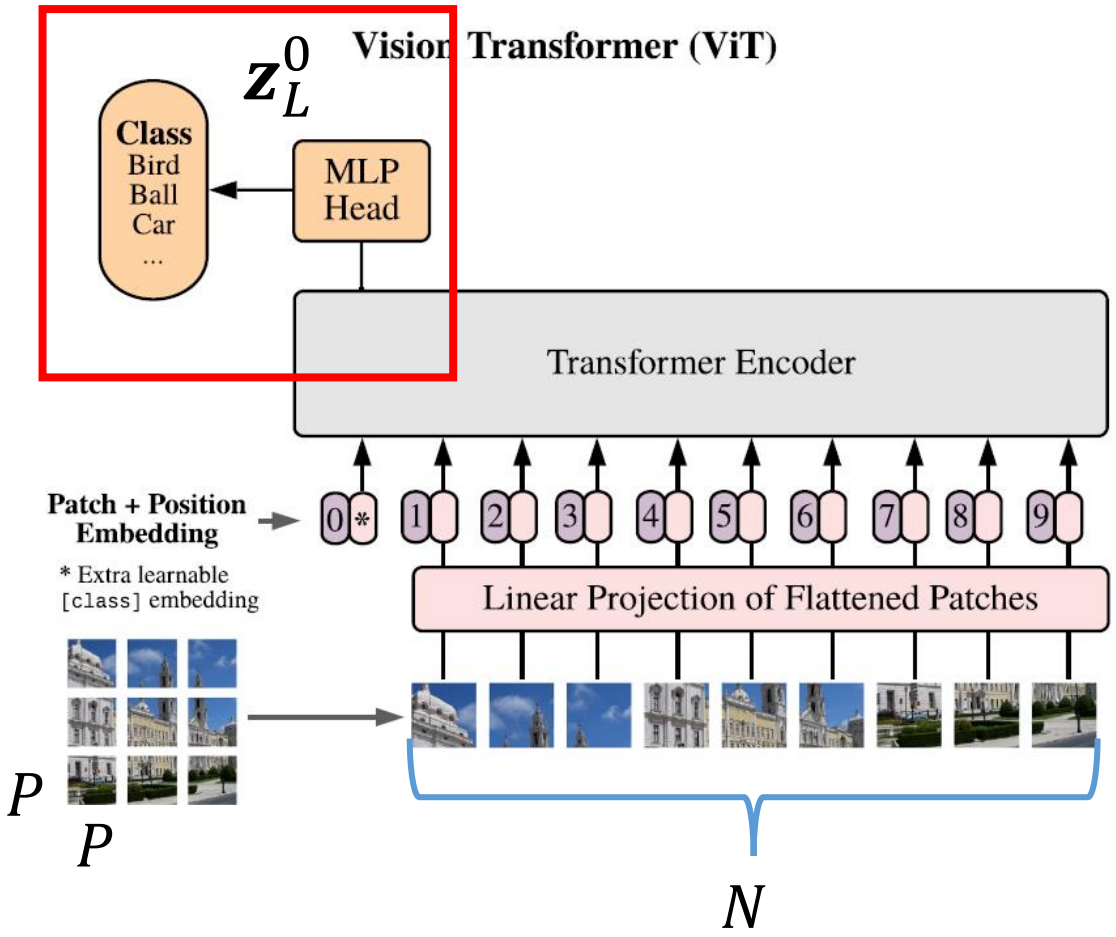


Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale*. ICLR 2021.

ViT Output

The classification head can be a simple linear layer (here MLP head) attached to the z_L^0 **output token**

As in modern CNNs, the feature classifier is relatively simple…



Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale*. ICLR 2021.

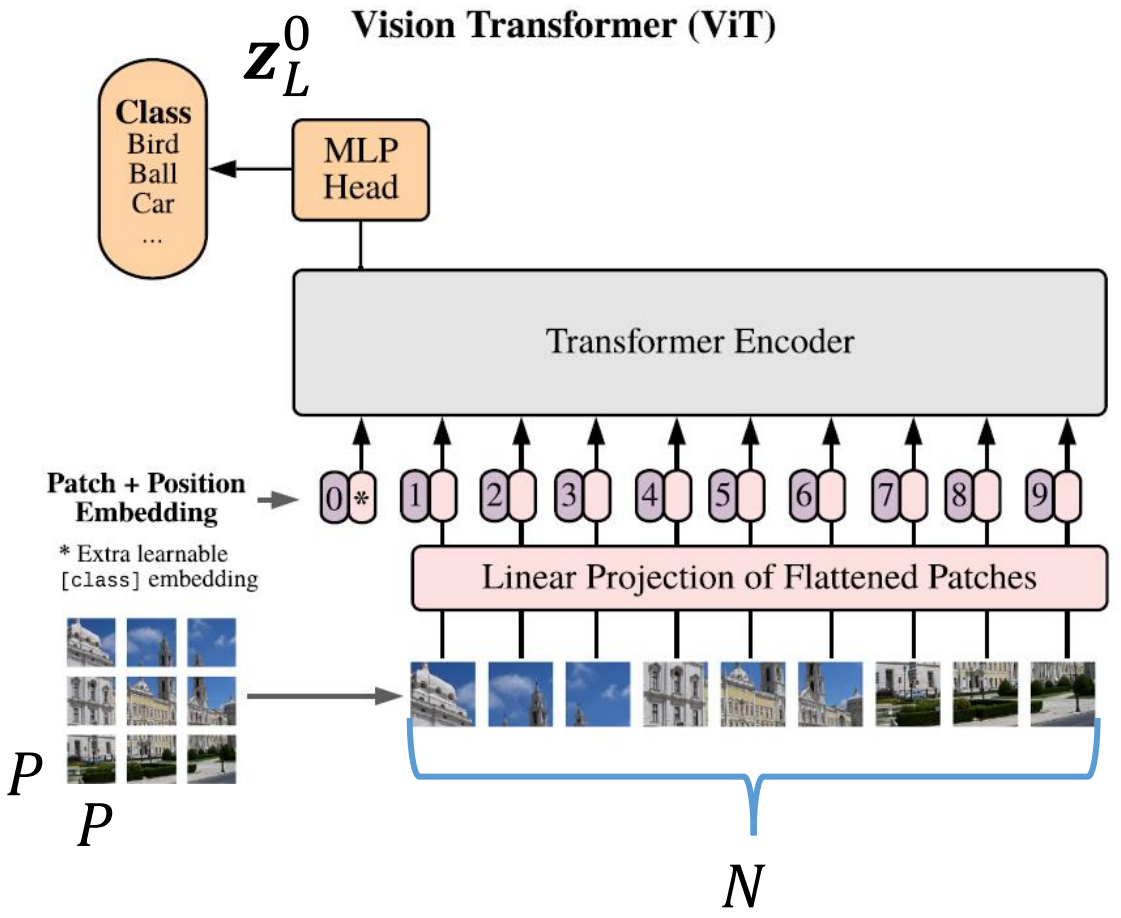
ViT Training

Pre-training: train ViT as a **classifier on huge datasets** with the MLP on the top made of a **single hidden layer**

Fine-Tuning: the **MLP is discarded and replaced by a linear layer** mapping to the K output classes of the specific classification problem

When **operating on higher resolution images, the patch-size P cannot be changed.**

The same model is used on a longer sequence, but the positional **embeddings are lineary interpolated**



Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale.* ICLR 2021.

Inductive bias in ViT

CNNs have **locality, two-dimensional** structure, and **translation invariance** properties that are granted **by design**.

ViT include much **fewer image-specific inductive bias than a CNN**

- locality, image is always used batch-wise (patchify layer)
- Interpolation of 2D positional embeddings (to adjust location in images)
- Self-attention layers are global

However, there are no substantial differences between 1D or 2D positional embeddings…

All the **spatial relations** among patches have to be **learned in ViT**.

Trained Models

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

Table 1: Details of Vision Transformer model variants.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale.* ICLR 2021.

Classification Performance

Legend: ViT-[MODEL_TYPE] / [PATCH_SIZE]

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

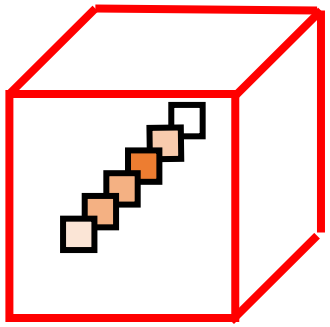
Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., ... & Houlsby, N. (2020). *An image is worth 16x16 words: Transformers for image recognition at scale.* ICLR 2021.

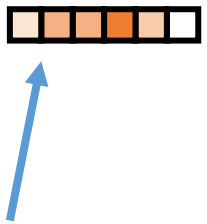
Alternative to Patch Extraction

It is also possible to perform the linear embedding on the latent representation of a CNN backbone.

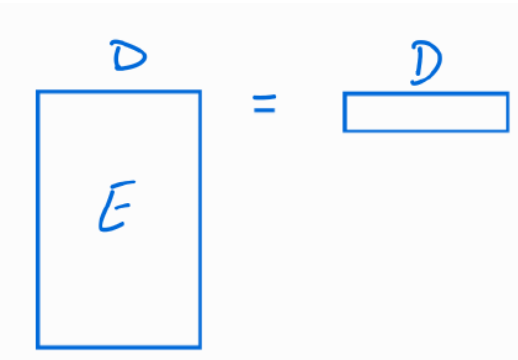
This is the idea behind Compact Convolutional Transformers (CCT)

Latent features



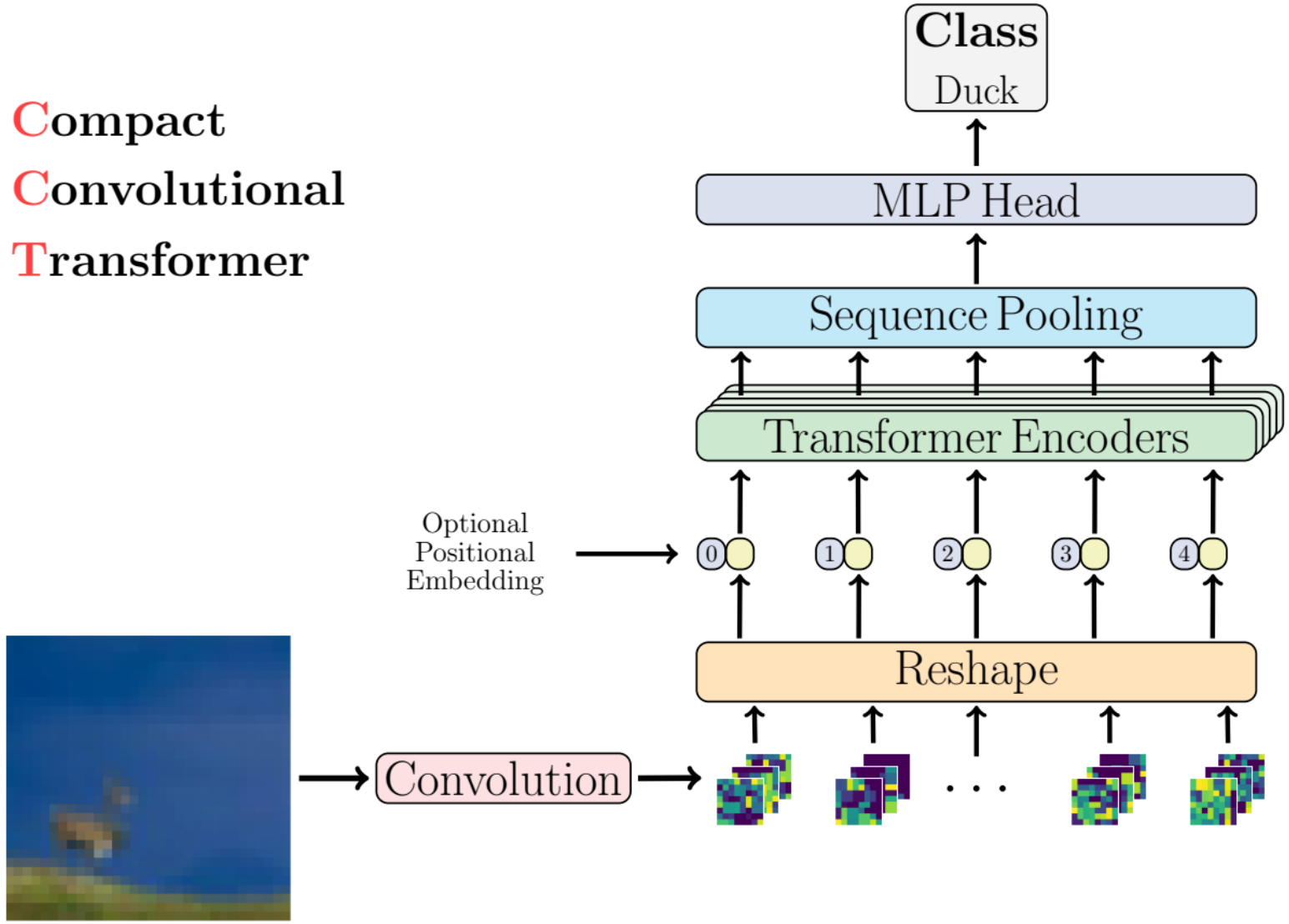


It is possible to pair a positional encoder here



Compact Convolutional Transformers (CCT)

Compact
Convolutional
Transformer



Hassani, A., Walton, S., Shah, N., Abuduweili, A., Li, J., & Shi, H. (2021). Escaping the big data paradigm with compact transformers. *arXiv preprint arXiv:2104.05704.*

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

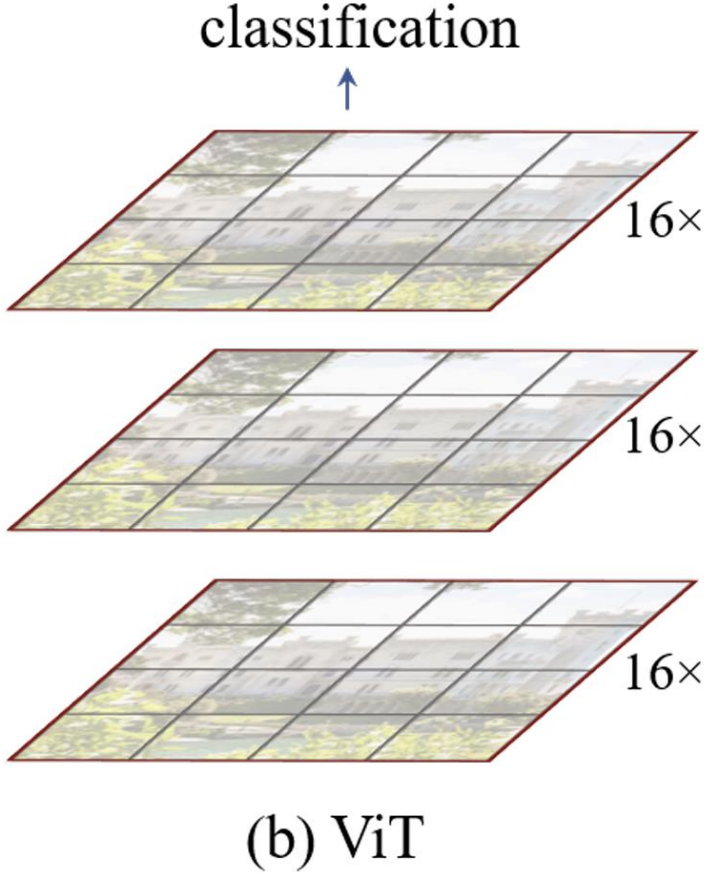
Ze Liu[1,2†*] Yutong Lin[1,3†*] Yue Cao[1*] Han Hu[1*‡] Yixuan Wei[1,4†]
Zheng Zhang[1] Stephen Lin[1] Baining Guo[1]
[1]Microsoft Research Asia [2]University of Science and Technology of China
[3]Xian Jiaotong University [4]Tsinghua University
{v-zeliu1,v-yutlin,yuecao,hanhu,v-yixwe,zhez,stevelin,bainguo}@microsoft.com

Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. "Swin transformer: Hierarchical vision transformer using shifted windows." CVPR 2021.

Vanilla ViT models

Challenges in adapting Transformer from language to vision arise from differences between the two domains, such as large variations in the scale of visual entities and the high resolution of pixels in images compared to words in text.

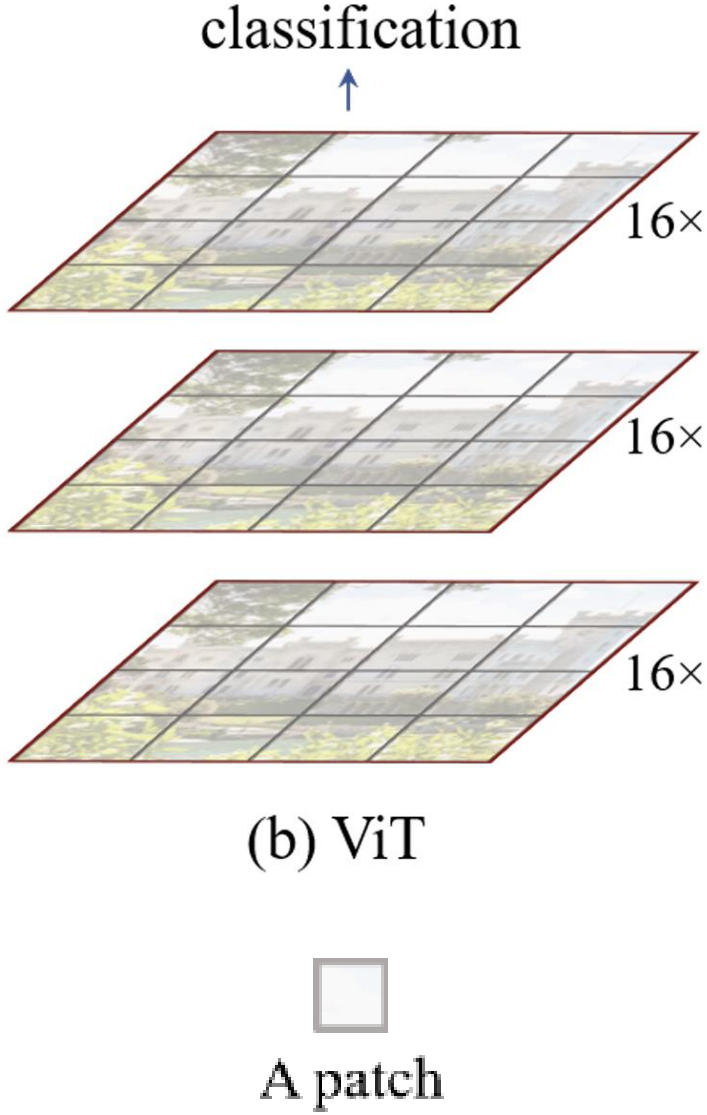
.. we seek to expand the applicability of Transformer such that it can serve as a general-purpose backbone for computer vision, as it does for NLP and as CNNs do in vision.



(b) ViT

Vanilla ViT models

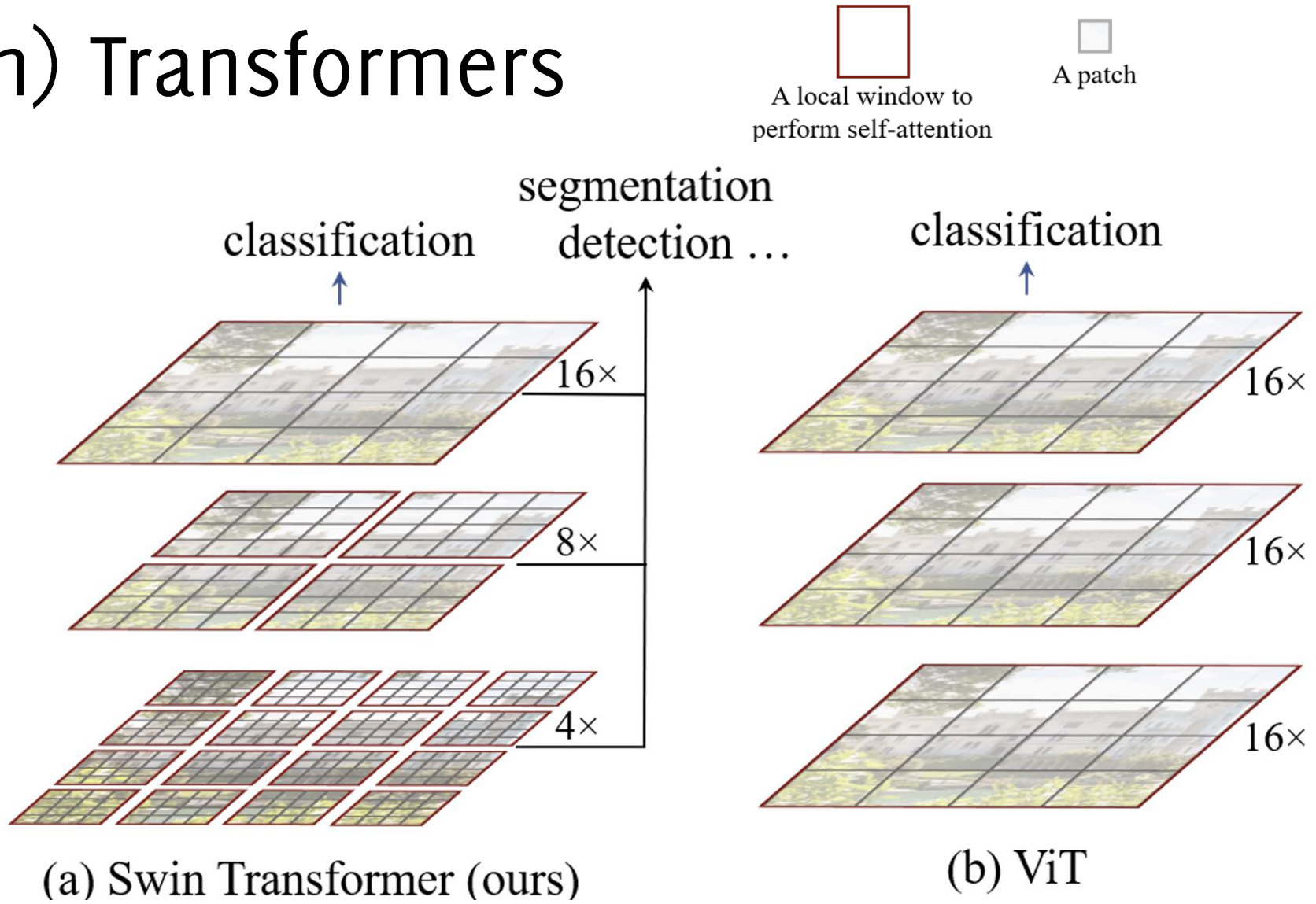
- Achieve state-of-the-art performance in Image Classification when trained on very large datasets
- The only **image-specific inductive bias** consists in the «patchify» layer, which crops and embed image patches.
- As a matter of fact, **ViT backbones do not excell** when solving higher level vision tasks (e.g. object detection) and in providing dense predictions (e.g. segmentation).
- **Intractable for high-resolution images**, due to the **quadratic complexity** of **global attention mechanism.**
- Difficulties in handling objects at different scales, as the patch-size is fixed (tokens have a fixed size)



Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. "Swin transformer: Hierarchical vision transformer using shifted windows." CVPR 2021.

Hierarchical (Swin) Transformers

- Constructs a **hierarchical feature map, gradually merging patches in deeper layers**
- Provides **accurate dense predictions** also on high-resolution images
- Attention is computed within local windows, which makes the computation time linear w.r.t. image dimensions
- Achieves strong performance on image classification, object detection and semantic segmentation

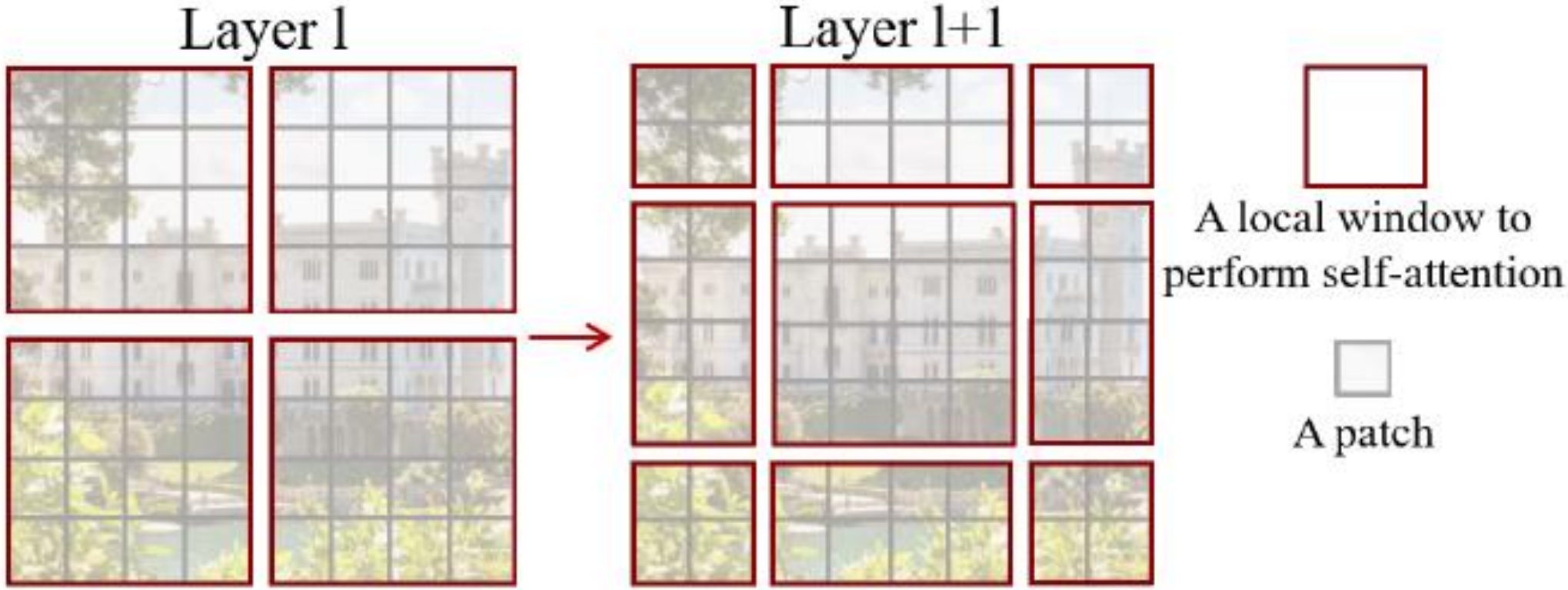


Patches are in gray, the window where attention is computed is in red

Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. "Swin transformer: Hierarchical vision transformer using shifted windows." CVPR 2021.

Hierarchical (Swin) Transformers

Introduces **attention within local windows**: performns **sliding window / local processing like CNNs**. This result in connections among neighboring regions where attention is computed.



Liu, Ze, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. "Swin transformer: Hierarchical vision transformer using shifted windows." CVPR 2021.

Combining Text and Images

Giacomo Boracchi

Advanced Classification Settings

Closed Set Classification Settings (the standard)

You are given a training set

$$TR = \{(\boldsymbol{x}(t), y(t)), \boldsymbol{x} \in \mathbb{R}^d, y \in \Lambda\}$$

You train a classifier

$$\mathcal{K}: \mathbb{R}^d \rightarrow \Lambda$$

Which has to classify instances having label Λ.

Open Set Recognition Settings

The classifier $\mathcal{K}$ at test time need to recognize possible input samples that do not belong to any class in Λ

$$\mathcal{K}: \mathbb{R}^d \rightarrow \{\Lambda, \text{"unknown"}\}$$

These are more general settings and require changing $\mathcal{K}$ after training or adopting specific training procedures

Advanced Classification Settings

Open set recognition requires specific model or training procedures

- Even though one of the simplest solution seems to be monitoring the maximum of logit!

Still, this **does not provide any hint more than "unknown"** for images that do not belong to Λ.

… a much more appealing alternative would be to train a computer vision model having "natural language" as a target.

Vaze, Sagar, Kai Han, Andrea Vedaldi, and Andrew Zisserman. "Open-set recognition: A good closed-set classifier is all you need." arXiv preprint arXiv:2110.06207 (2021).

Advanced Classification Settings

Zero Shot Learning at test time, the classifier observes samples from classes do not belonging to Λ, and **needs to predict the class that they belong to!**

This looks impossible… indeed Zero-shot learning leverage additional information to exploit regularities in the data

- e.g. classes are accompanied by pre-defined structured description (e.g. “red head” description to unseen bird classes)
- Free textual description (“zebra is a striped horse”)
- Class similarity in the embedding

Few Shot Learning: a very small amount of annotated training samples (e.g. a dozen) is provided for fine tuning the model

Vaze, Sagar, Kai Han, Andrea Vedaldi, and Andrew Zisserman. "Open-set recognition: A good closed-set classifier is all you need." arXiv preprint arXiv:2110.06207 (2021).

Learning Transferable Visual Models From Natural Language Supervision

Alec Radford [*1] Jong Wook Kim [*1] Chris Hallacy [1] Aditya Ramesh [1] Gabriel Goh [1] Sandhini Agarwal [1]
Girish Sastry [1] Amanda Askell [1] Pamela Mishkin [1] Jack Clark [1] Gretchen Krueger [1] Ilya Sutskever [1]

[1]openAI

CLIP: Contrastive Language Image Pre-Training

Idea: ***Learning perception*** *from the* ***supervision*** *contained in* ***natural language*** *paired with images.*

What kind of «natural language paired with images»?

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

CLIP: Contrastive Language Image Pre-Training

Idea: ***Learning perception*** *from the* ***supervision*** *contained in* ***natural language*** *paired with images.*

What kind of «natural language paired with images»?

Image Captions, of course!

A cat sitting on a chair

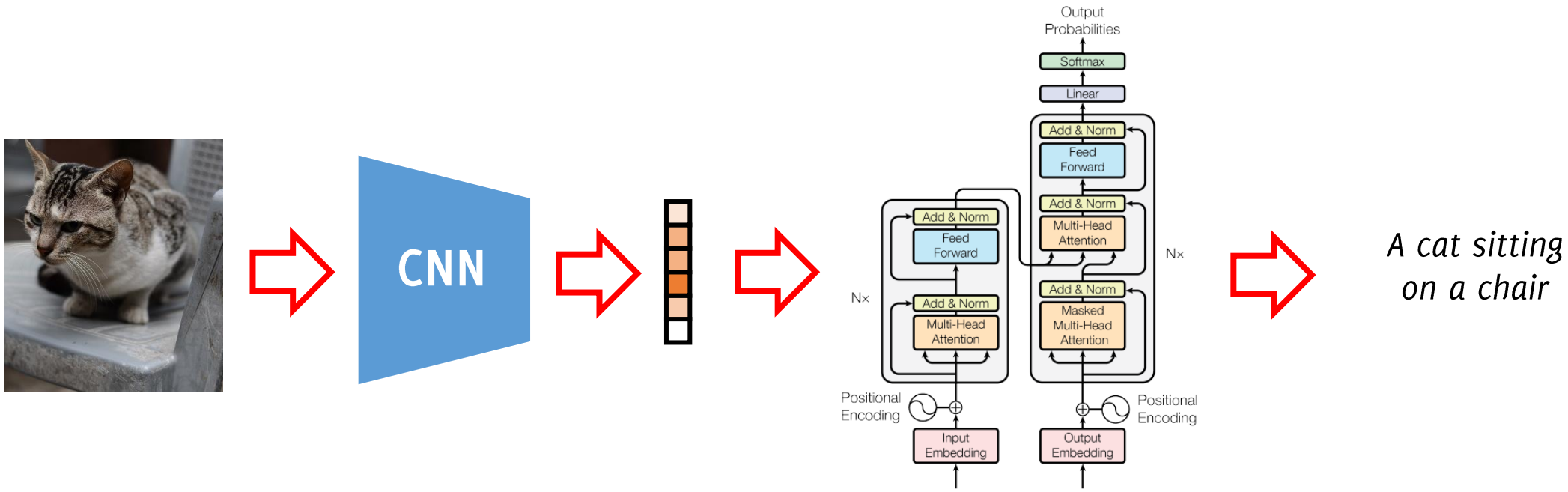
Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

Captioning

Jointly train two networks:

- a CNN for images / a ViT
- a transformer for text

predict the exact words of each caption given the input image



WiT Dataset

WIT: WebImageText, a new dataset is constructed containing **400 million (image, text) pairs.**

Images are collected form publicly available sources on the Internet, selected to have the **text answering among 500K queries**. Approximatively 20K images per query.

A cat sitting on a chair

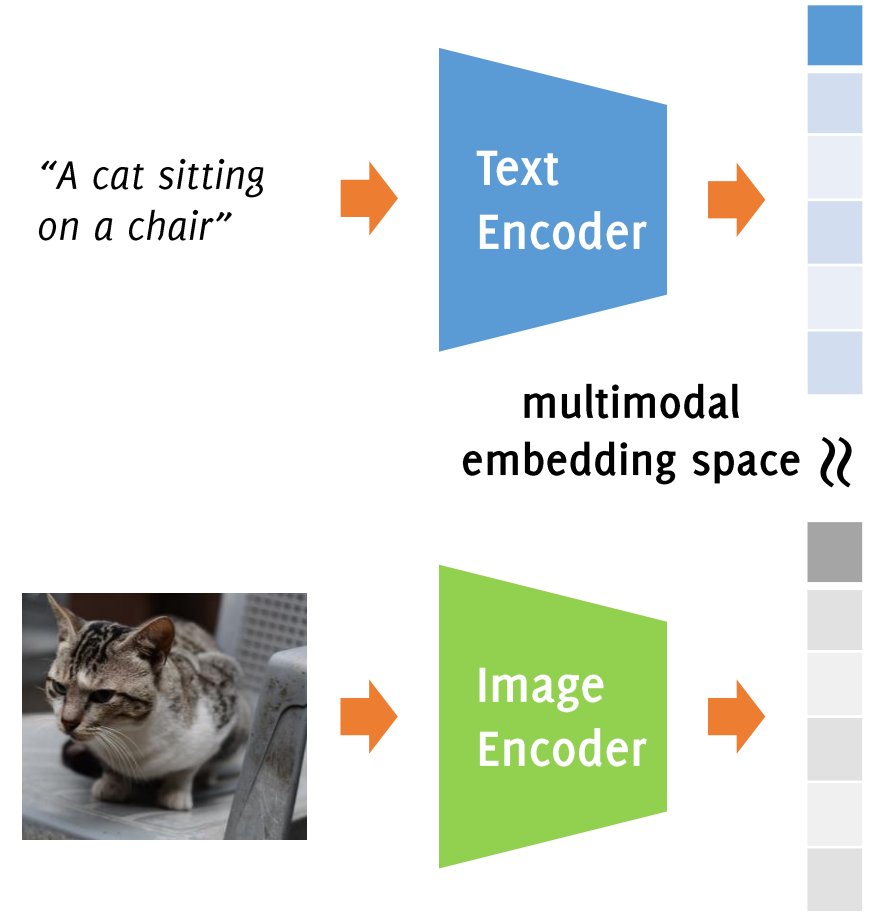
CLIP: Contrastive Language Image Pre-Training

In order to learn visual representation by text supervision there are better training options.

Different formulation of the problem: train models to predict **which text "as a whole" is paired with which image** (given a batch).

Models trained by minimizing **the contrastive loss** were shown to **outperform models trained to predict the equivalent objective.**

The key idea is to learn a multimodal embedding space by jointly training an image encoder and a text encoder



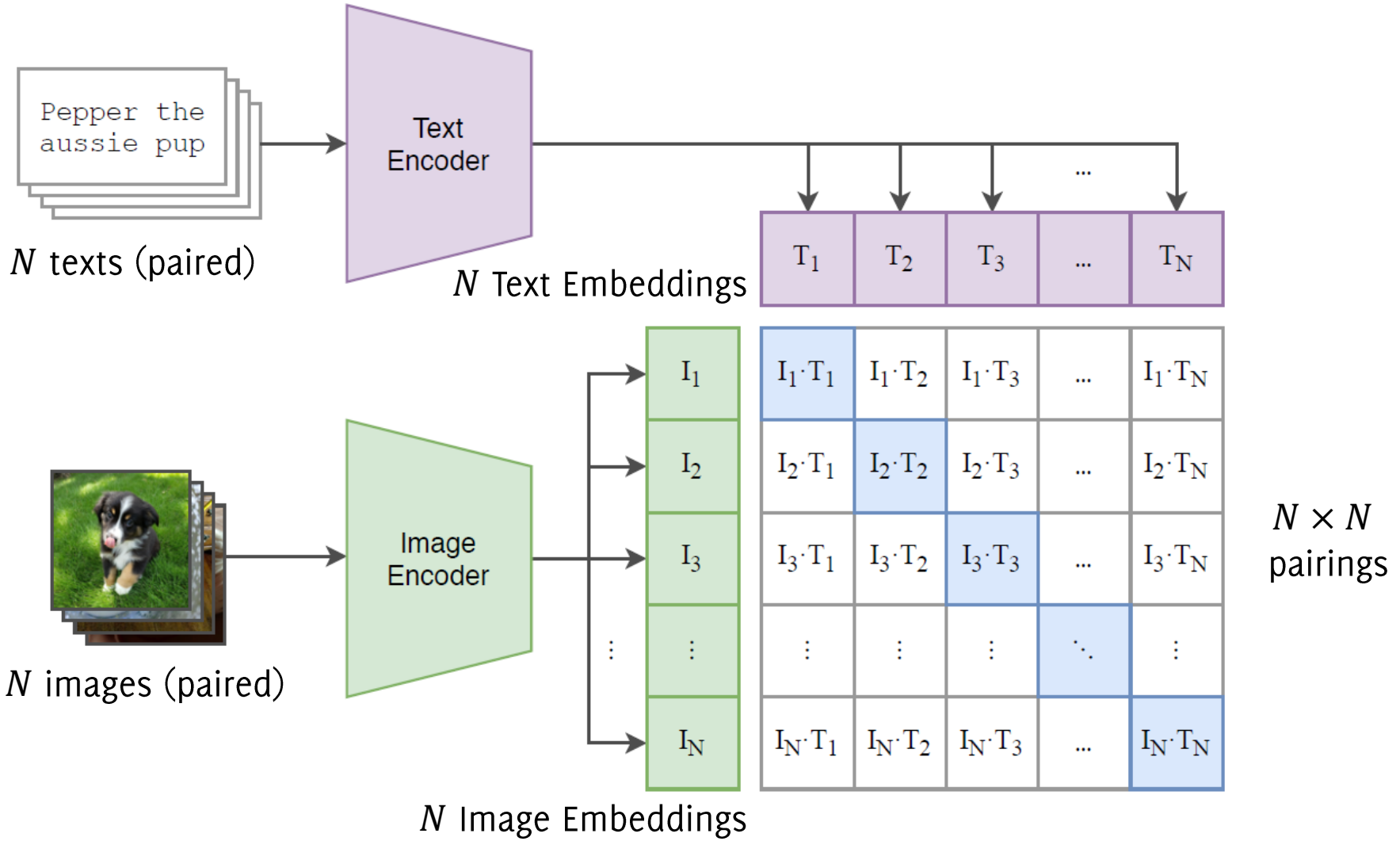
Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

CLIP: Contrastive Language Image Pre-Training

CLIP jointly trains:

- an image encoder
- a text encoder

to predict the correct pairings of (image, text) examples

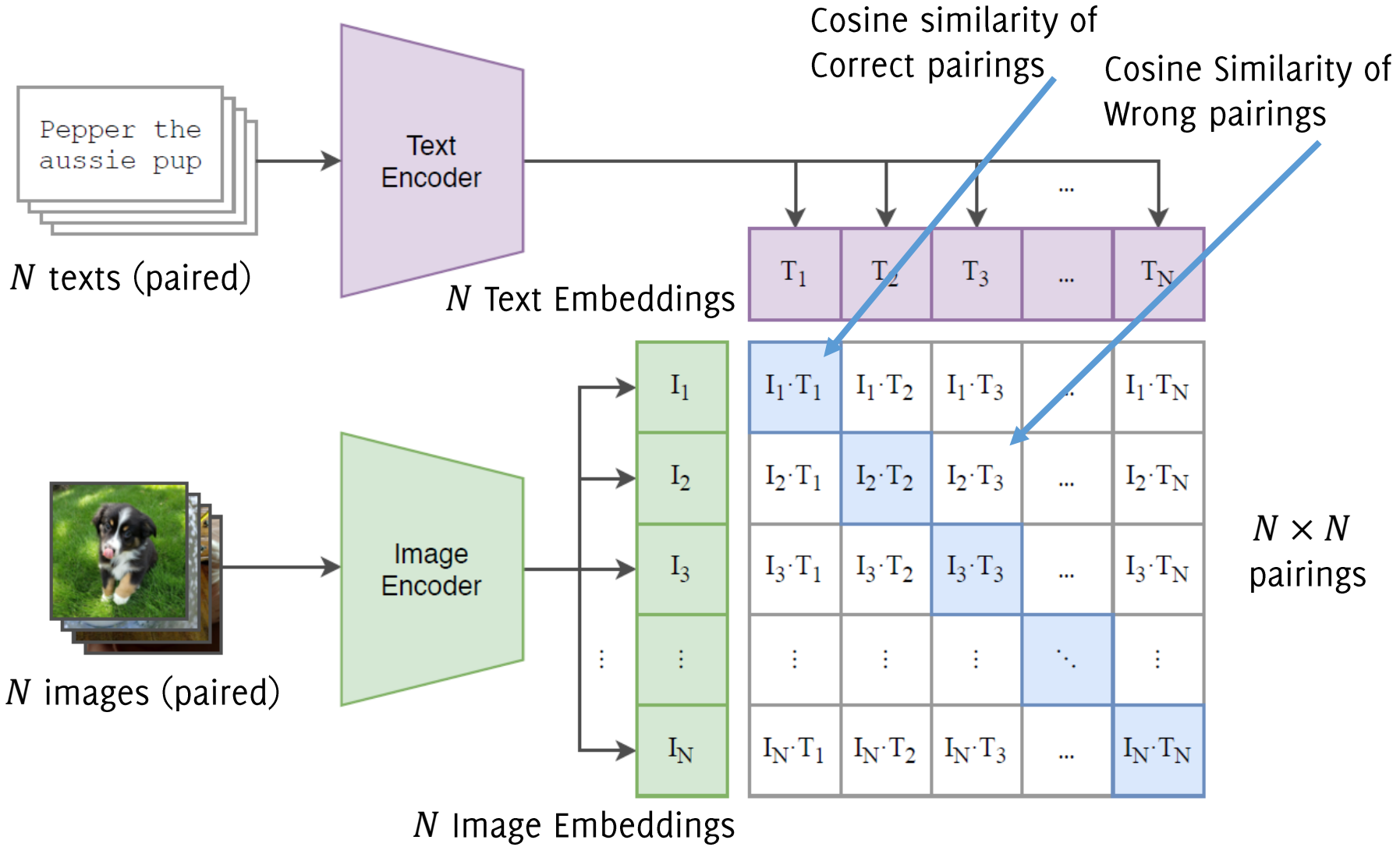


Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

CLIP: Contrastive Language Image Pre-Training

Maximize the cosine similarity among the N correct pairings (i.e. the diagonal entries of the matrix)

Minimize the cosine similarity at the $N^2 - N$ wrong pairings (i.e. the off-diagonal entries of the matrix)



Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

CLIP: Contrastive Language Image Pre-Training

- no pre-training needed,
- the only form of augmentation is image cropping

Two learnable feature extractors.
Extract features from the entire batch.

Two learnable linear projections to compute the embeddings in the **same space**

Compute cosine similarities + temperature scaling (embeddings are normalized)

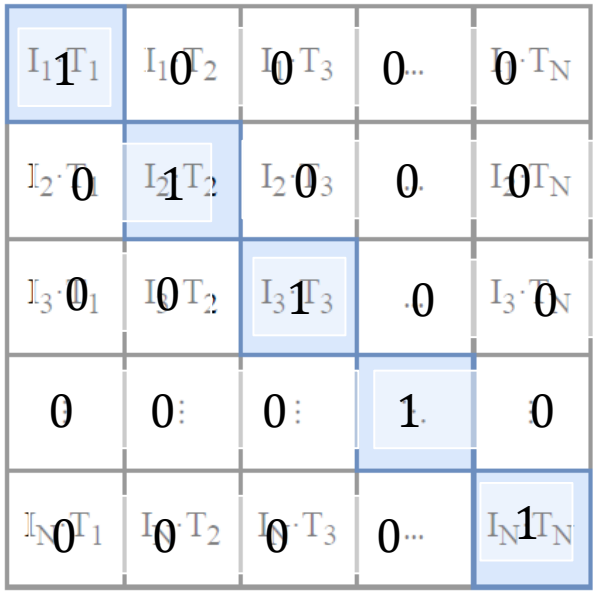
Sum the off-diagonal and diagonal loss using different labels

```
# 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
```

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

CLIP: Contrastive Language Image Pre-Training

Ideally, we **would like to have a matrix like this.**



Entries are always in [0,1]:

- 0 when embeddings are orthogonal,
- 1 when they are parallel

Entries do not sum to 1

- Both columns and rows of the perfect association can be seen as 1-hot encoding
- Departure from this ideal solution can be assessed by categorical cross-entropy.
- This is how you implement in Py

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

Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

Zero-Shot Classifier

CLIP is pre-trained to predict if an image and a text are paired together in WIT… that's not very useful by itself!

However, it enables **Zero-Shot / Few-Shot learning,** say having a model to predict classes of unknown events with zero/very little annotated training examples.

Testing zero-shot learning is a way to study the generalization capabilities of a model to unseen datasets. That's a quantitative assessment of task-learning capabilities.

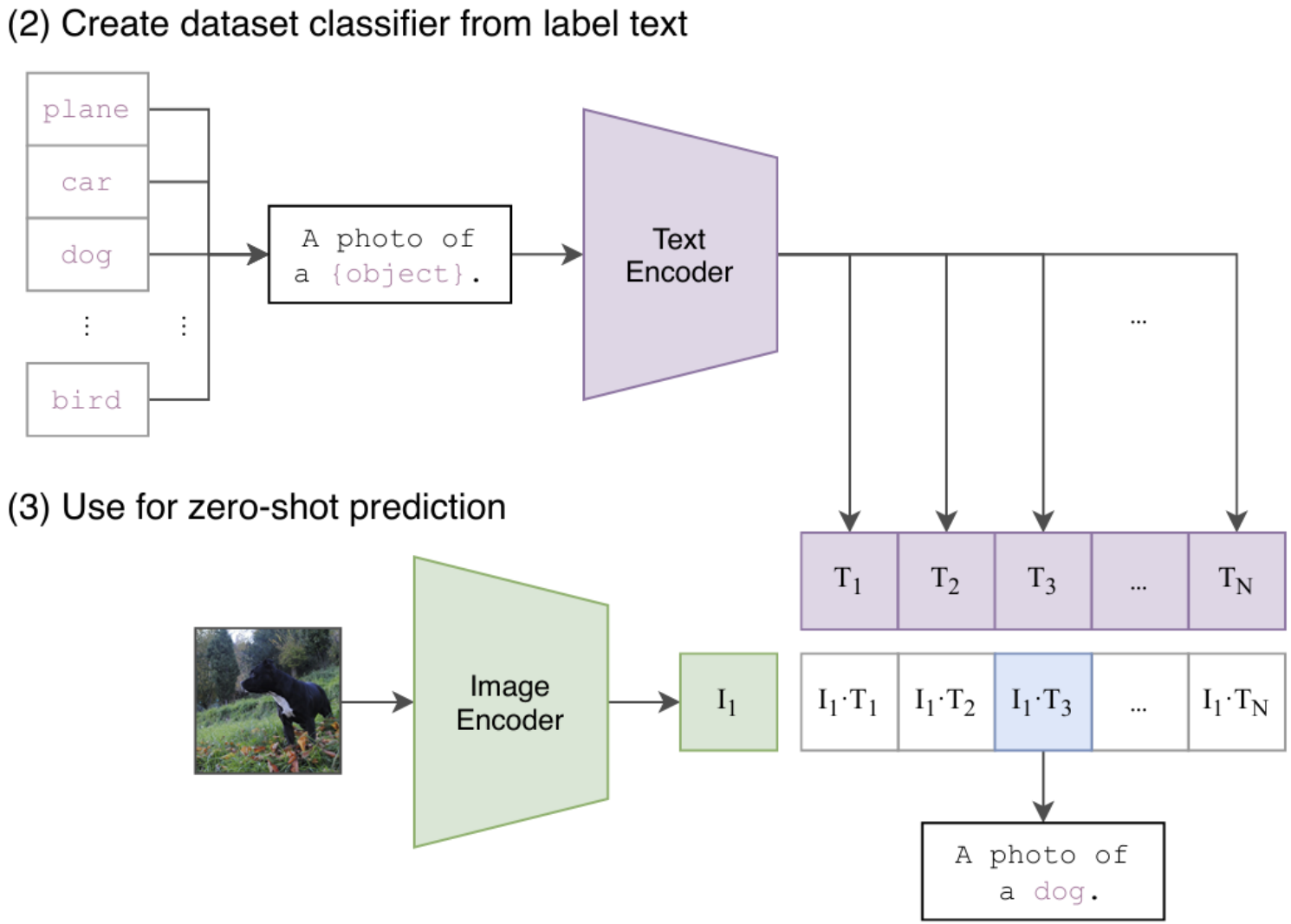
Zero-Shot CLIP

- Take pre-trained CLIP
- Consider the class names as text $\{t_i\}$
- Classify an image I by associating it to the text $\{t_i\}$ that is closest to I according to CLIP

Zero-Shot Classifier by CLIP

Zero-Shot CLIP

- Take pre-trained CLIP
- Consider the class names we are interested in as text $\{t_i\}$, and form a **prompt**
- Classify an image I by associating it to the **prompt** $\{t_i\}$ that is closest to I according to CLIP in the **multimodal embedding space**

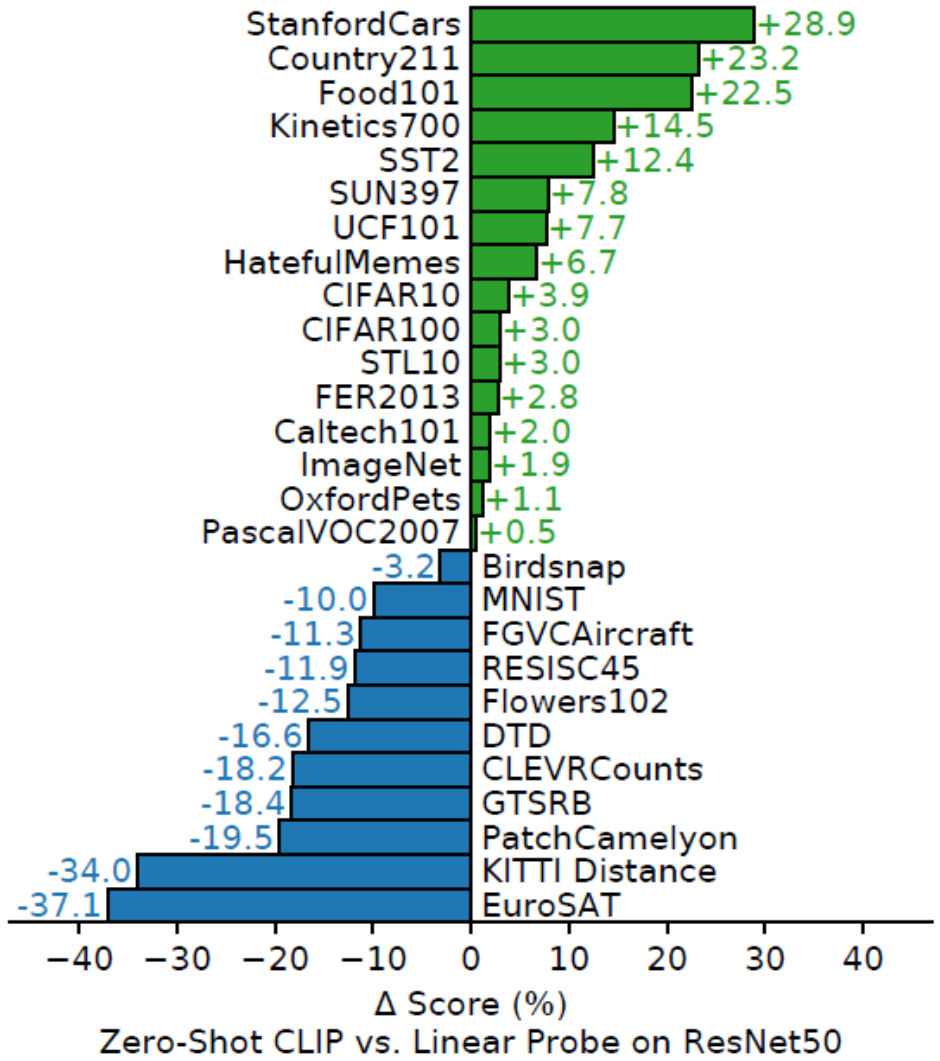


Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

Zero-Shot Classification Performance

Baseline: fully supervised classifier and regularized (logistic regression) fitted on the ResNet50 pre-trained features. This is trained in a supervised manner.

Zero-shot CLIP classifier outperforms a fully supervised classifier on 16 datasets, including ImageNet.



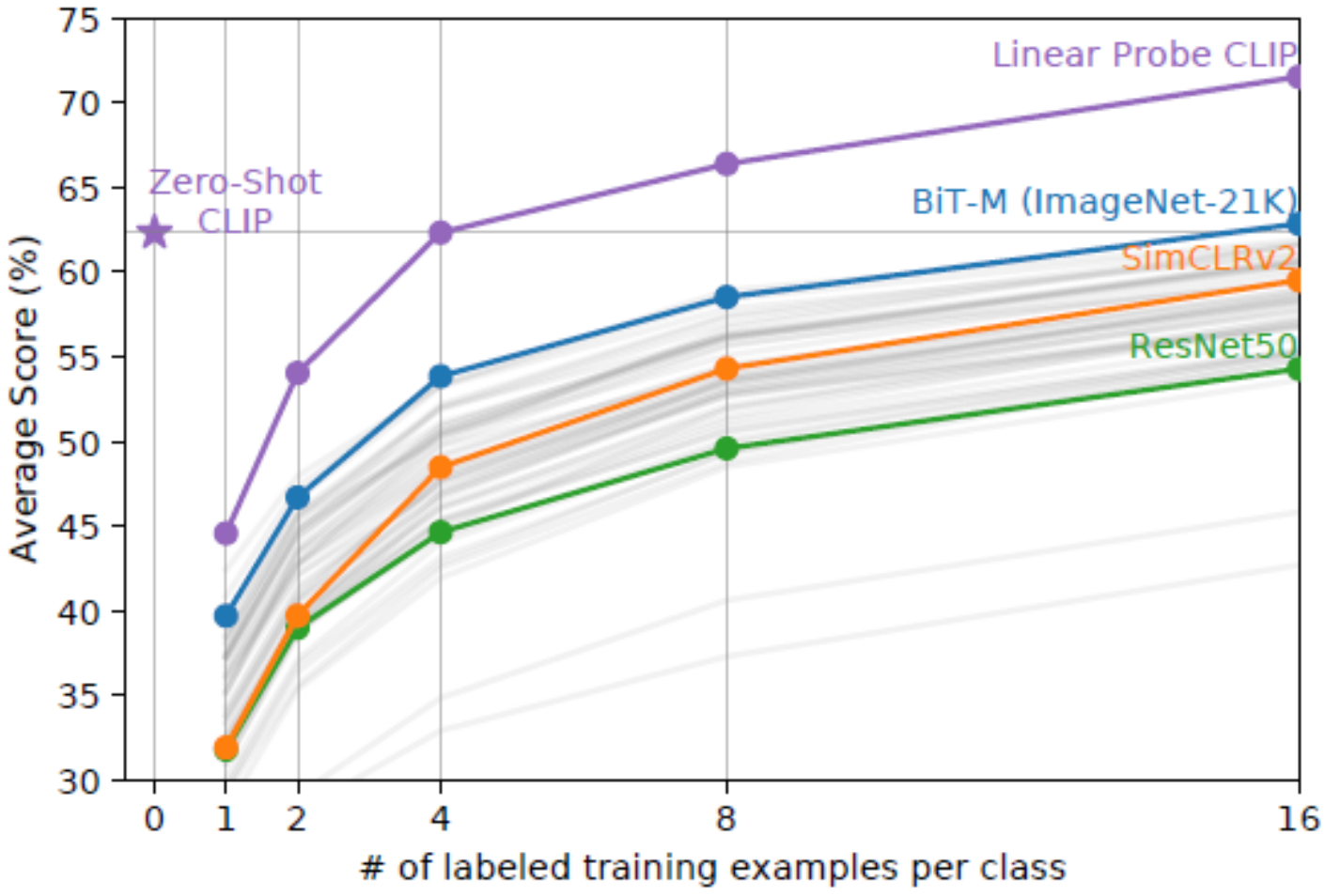
Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

Few-Shot Classification Performance

Comparison **against few shot learning solutions**, where a logistic regression is trained on features from other models.

Zero-shot CLIP matches the performance of 4-shot logistic regression on the same feature space

- Zero-shot CLIP classifier is generated via natural language, and allows to directly "communicate" visual concepts
- Supervised learning must infer concepts indirectly from training examples



Radford, Alec, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry et al. *"Learning transferable visual models from natural language supervision."* ICML, 2021.

Clip Limitations

- Around **1000x increase in compute is required** for zero-shot CLIP to reach overall SOTA performance in all the dataset.

- Many complex visual recognition tasks are difficult to specify just through text

- There is a counter-intuitive drop in performance when transitioning from a zero-shot to a few-shot setting

More on Zero-Shot Learning by CLIP

Published as a conference paper at ICLR 2022

LANGUAGE-DRIVEN SEMANTIC SEGMENTATION

Boyi Li
Cornell University, Cornell Tech

Kilian Q. Weinberger
Cornell University

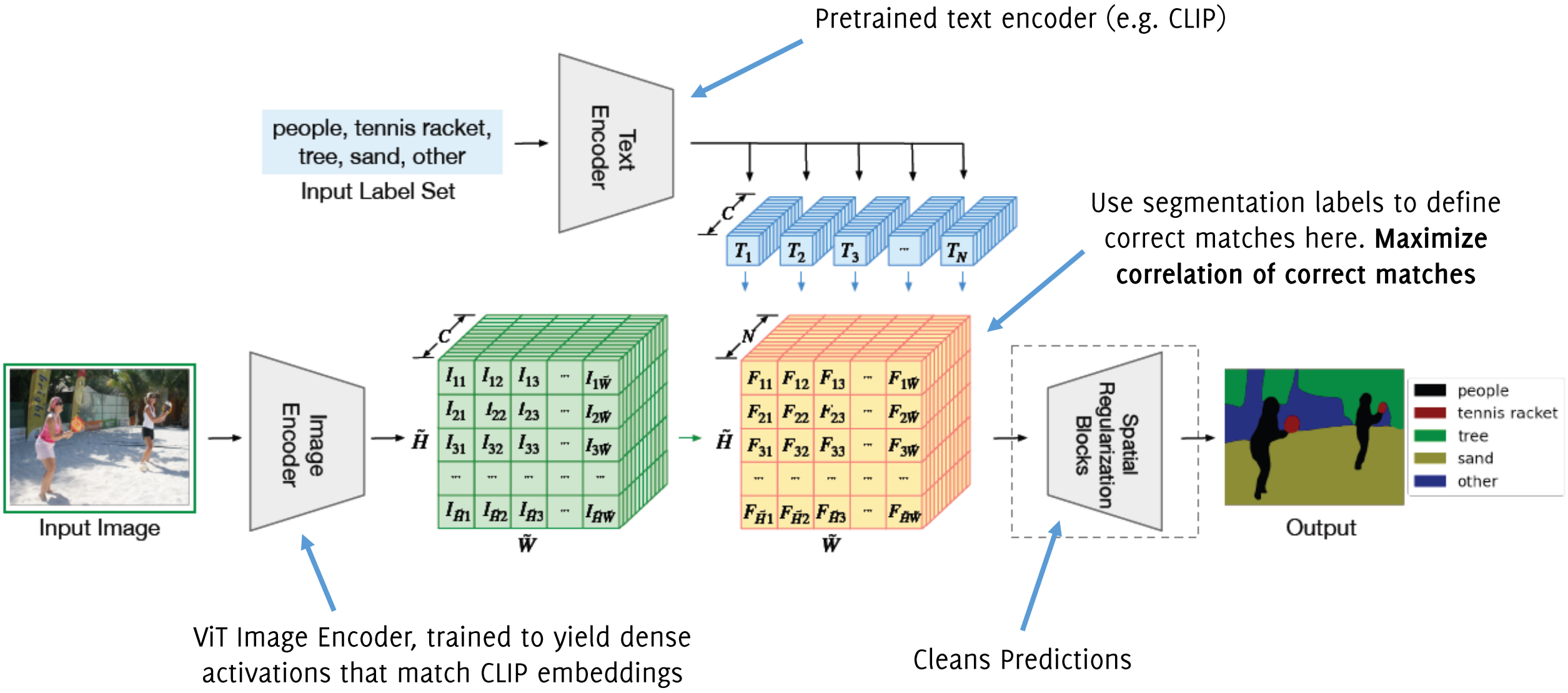
Serge Belongie
University of Copenhagen

Vladlen Koltun
Apple

René Ranftl
Intel Labs

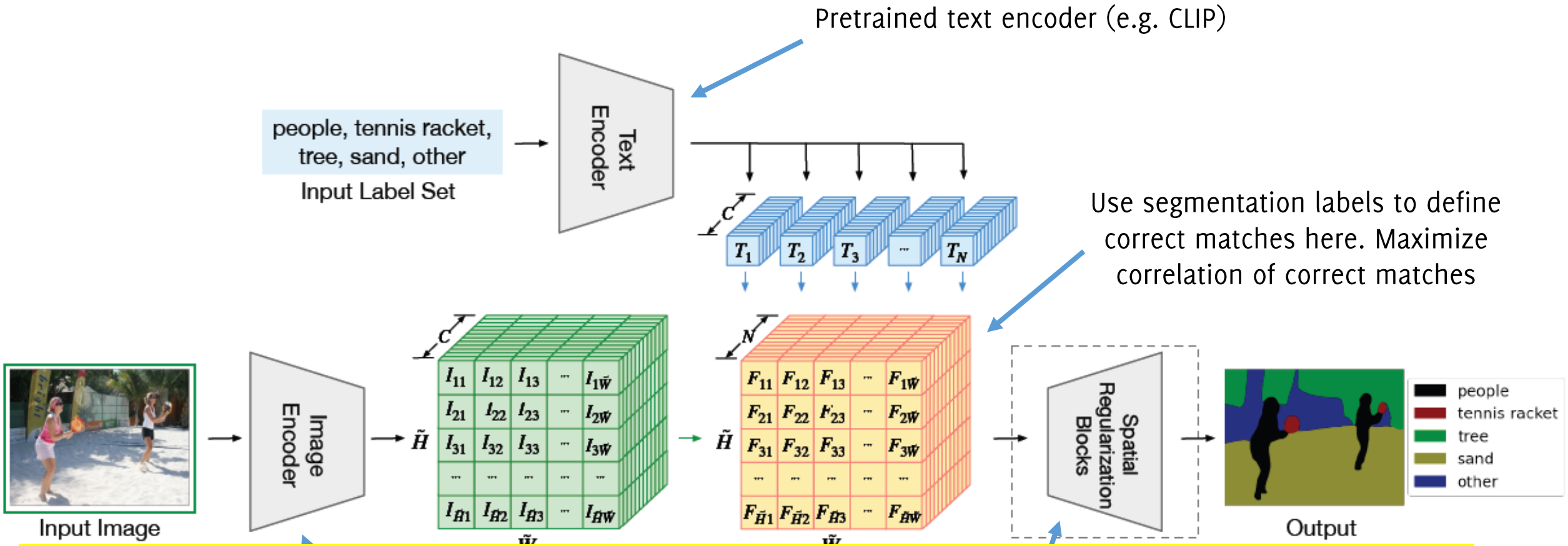
Li, Boyi, Kilian Q. Weinberger, Serge Belongie, Vladlen Koltun, René Ranftl. *"Language-driven semantic segmentation."* ICLR 2022

LSeg: Language Driven Semantic Segmentation



Li, Boyi, Kilian Q. Weinberger, Serge Belongie, Vladlen Koltun, René Ranftl. "Language-driven semantic segmentation." ICLR 2022

LSeg: Language Driven Semantic Segmentation



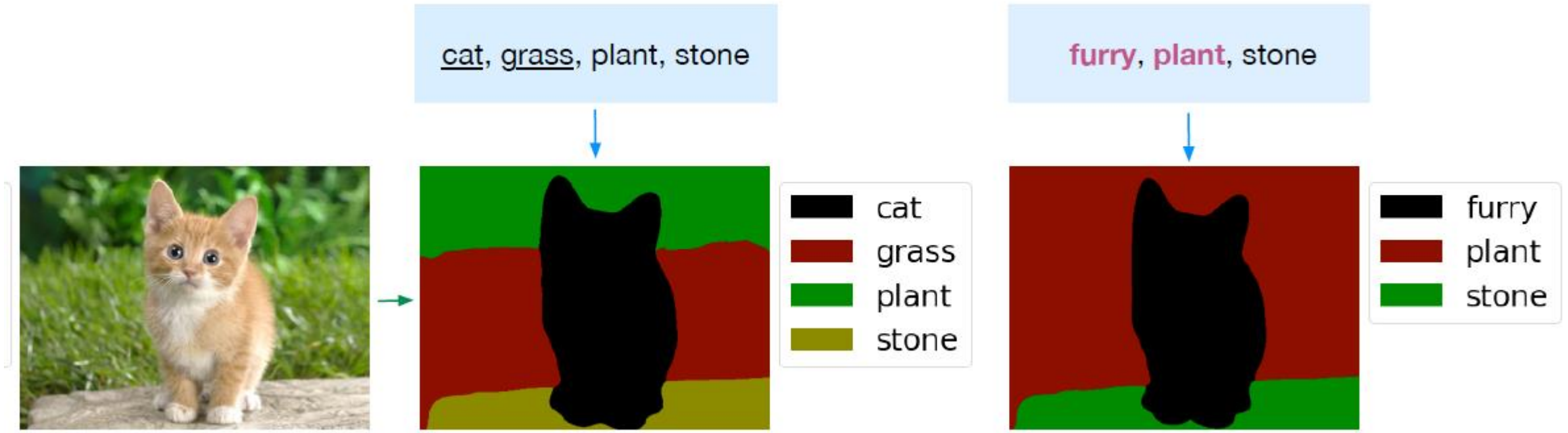
This model is trained on few labels for semantic segmentation.
However, since **CLIP text embedding brings words referring to the same visual concepts nearby**, we can perform zero-shot learning in semantic segmentation

Li, Boyi, Kilian Q. Weinberger, Serge Belongie, Vladlen Koltun, René Ranftl. "Language-driven semantic segmentation." ICLR 2022

Zero-Shot Semantic Segmentation

Red classes are added at test time (never annotated)

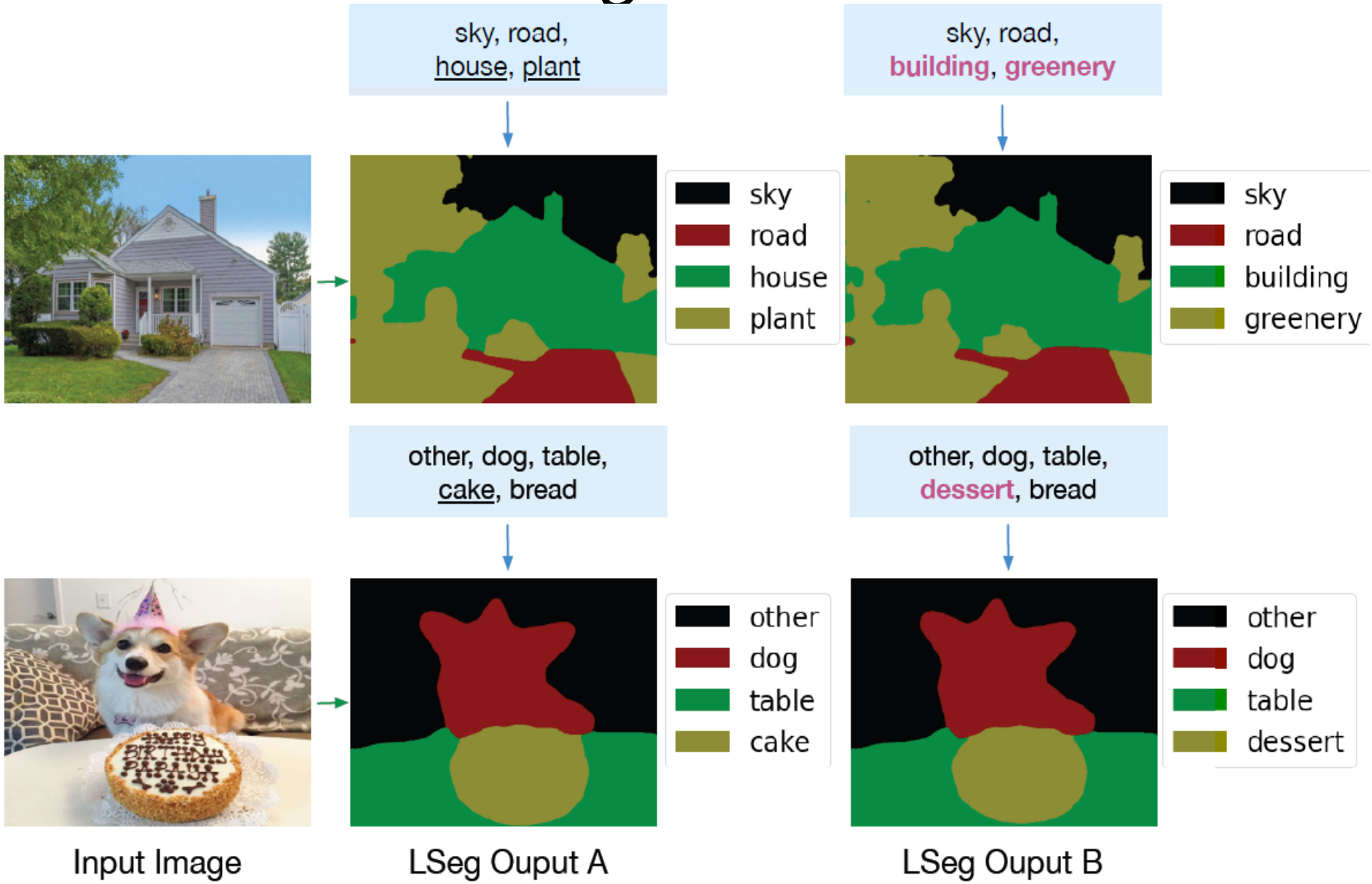
- You can arbitrarily expand, shrink, or reorder the label set for any image at test time
- No need of additional training samples



Li, Boyi, Kilian Q. Weinberger, Serge Belongie, Vladlen Koltun, René Ranftl. "Language-driven semantic segmentation." ICLR 2022

Zero-Shot Semantic Segmentation

Red classes are added at test time (never annotated)



Li, Boyi, Kilian Q. Weinberger, Serge Belongie, Vladlen Koltun, René Ranftl. "Language-driven semantic segmentation." ICLR 2022

Concluding Remarks

ConvNeXt

A ConvNet for the 2020s

Zhuang Liu[1,2*] Hanzi Mao[1] Chao-Yuan Wu[1] Christoph Feichtenhofer[1] Trevor Darrell[2] Saining Xie[1†]

[1]Facebook AI Research (FAIR) [2]UC Berkeley

Code: https://github.com/facebookresearch/ConvNeXt

Liu, Zhuang, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. "A convnet for the 2020s." CVPR 2022.

Bring Imaging-based Inductive Bias Back!

... many of the advancements of Transformers for computer vision have been aimed at bringing back convolutions. These attempts, however, come at a cost: a naive implementation of sliding window selfattention can be expensive [...] or more sophisticated in design.

On the other hand, it is almost ironic that a ConvNet already satisfies many of those desired properties, albeit in a straightforward, no-frills way.

[..]

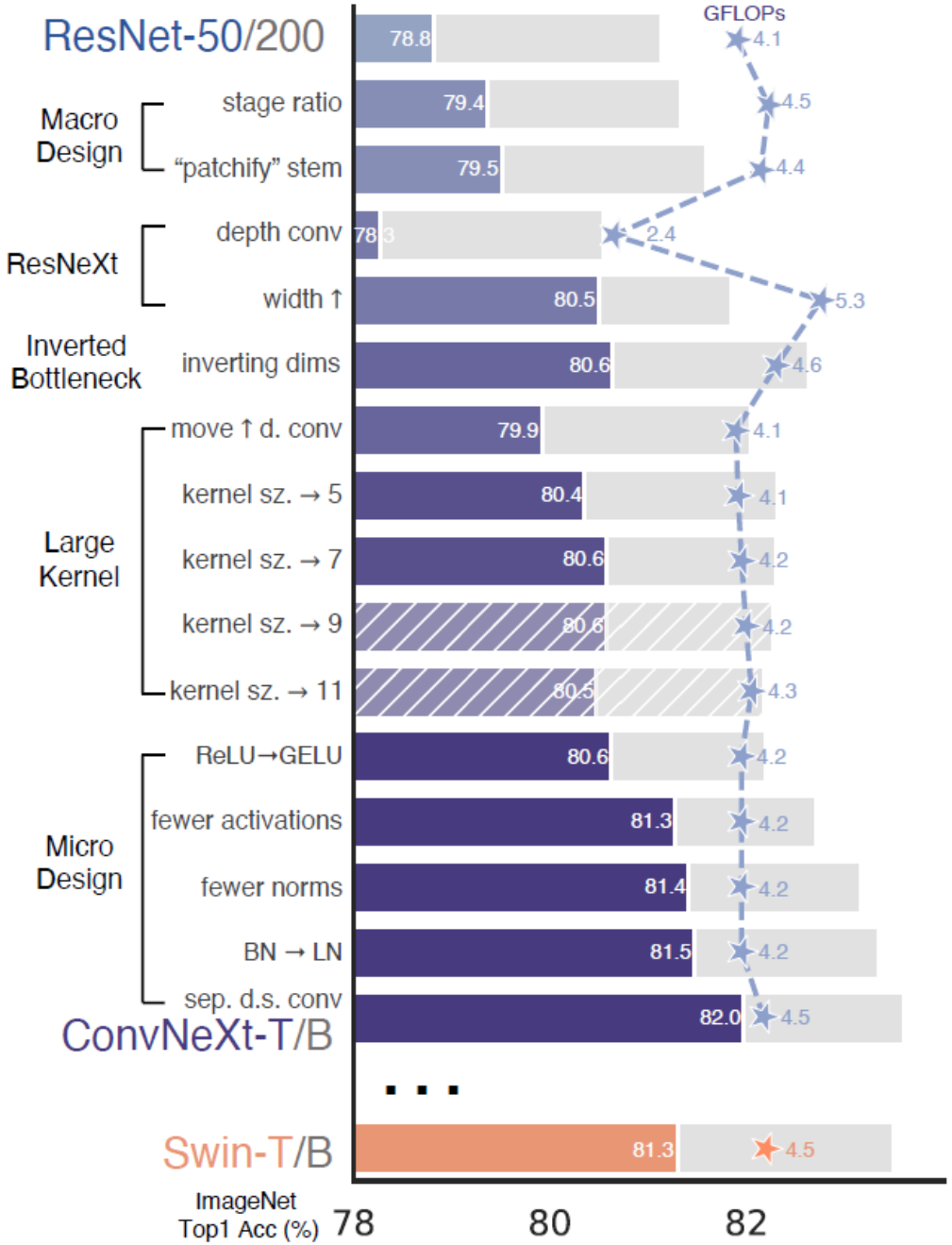
Our research is intended to ***bridge the gap between the pre-ViT and post-ViT*** *eras for ConvNets, as well as to* ***test the limits of what a pure ConvNet can achieve.***

Liu, Zhuang, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, Saining Xie. "A convnet for the 2020s." CVPR 2022.

ConvNeXt

To do this, we start with a standard ResNet (e.g. ResNet- 50) trained with an improved procedure. We gradually "modernize" the architecture to the construction of a hierarchical vision Transformer (e.g. Swin-T).

As a result, we propose a ***family of pure ConvNets dubbed ConvNeXt***



Liu, Zhuang, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, Saining Xie. "A convnet for the 2020s." CVPR 2022.

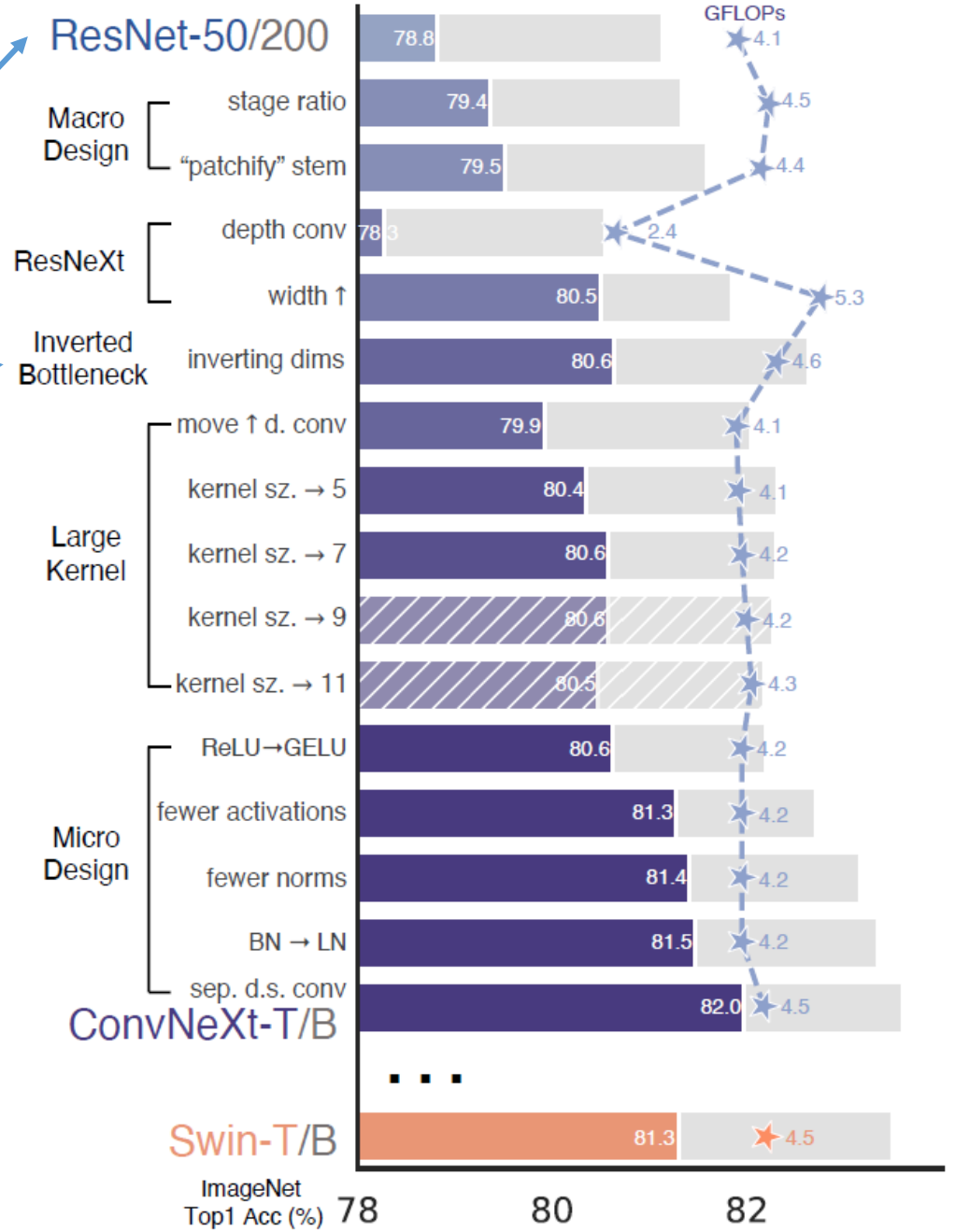
ConvNeXt

AdamW optimizer (used in Transformers) and modern data augmentation

Expand the latent dimension among residual connections (separable convolutions as in MobileNet)

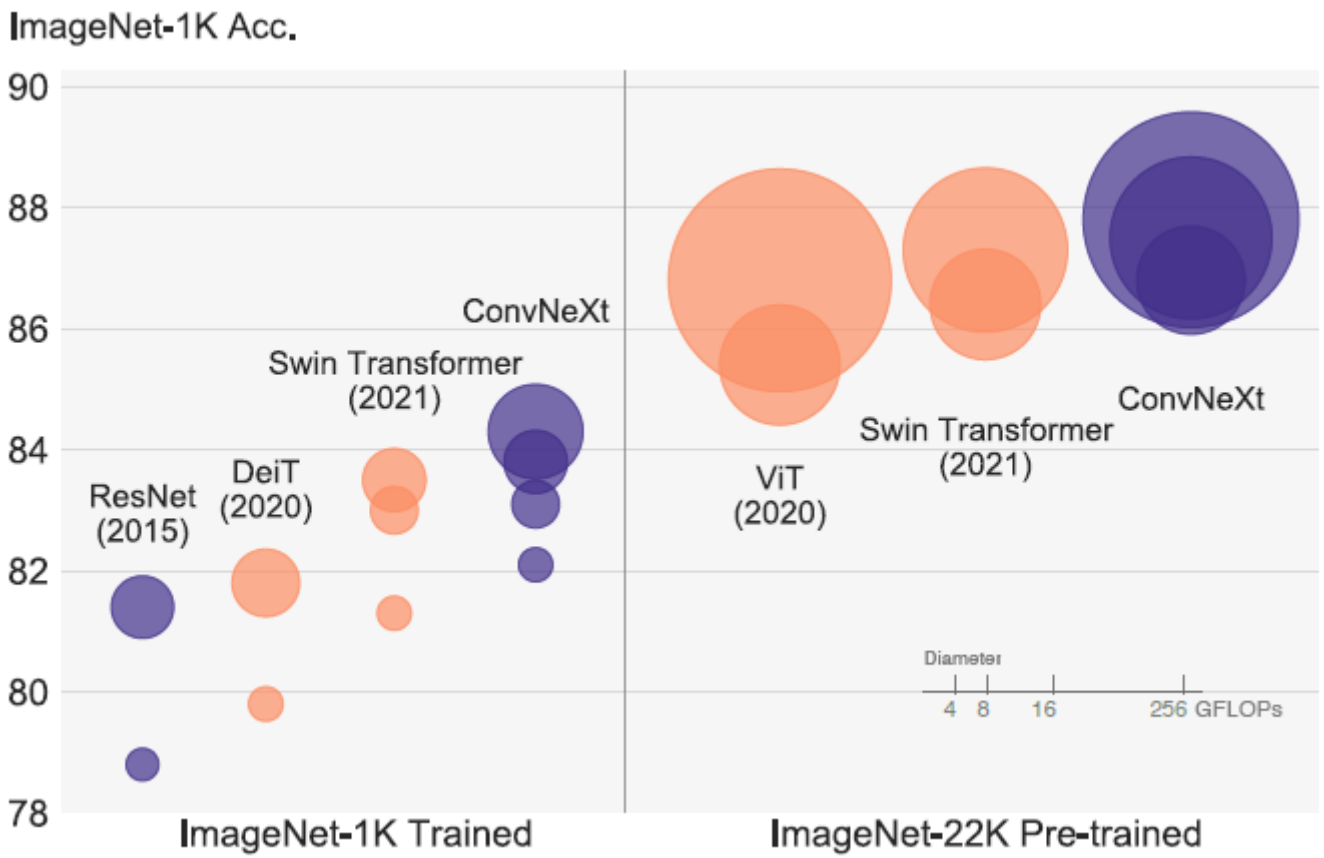
Gold Standard in CNN is to use 3x3 kernel size to get efficient processing and more nonlinearities (from VGG)
Hierarchical transformers use larger windows

All these variants are inspired from Transformers but uses methods that were already present in CNN.
Simply, these were never collectively used



Liu, Zhuang, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, Saining Xie. "A convnet for the 2020s." CVPR 2022.

ConvNeXt



Liu, Zhuang, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, Saining Xie. "A convnet for the 2020s." CVPR 2022.

Concluding Remarks

- The Advent of transformers represent a breakthrough in visual recognition
- It is surprising that they do not rely on “inductive biases” typical of images (and quite depressing from a computer-vision perspective). Huge training set seems enough
- Still, they seem to revert to implement “principles of vision” to solve advanced visual recogntition tasks
- They motivated a restyling of CNN that achived surprising results: ConvNeXt (nice to see inductive biases are back!)

- Transformers and Vision models (either CNNs or ViT) opens for new surprising applications combining text and vision