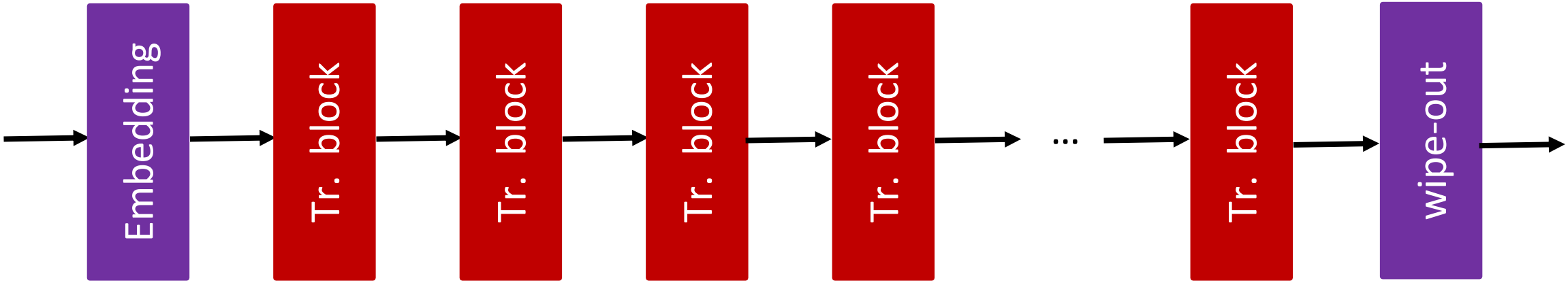
Detection Transformer and Foundation Models

Andrej Lúčny

Cross attention.Detection Transformer (DeTR).
Vision model with text prompt.
Foudation models: DINO, CLIP, Glee, SAM.

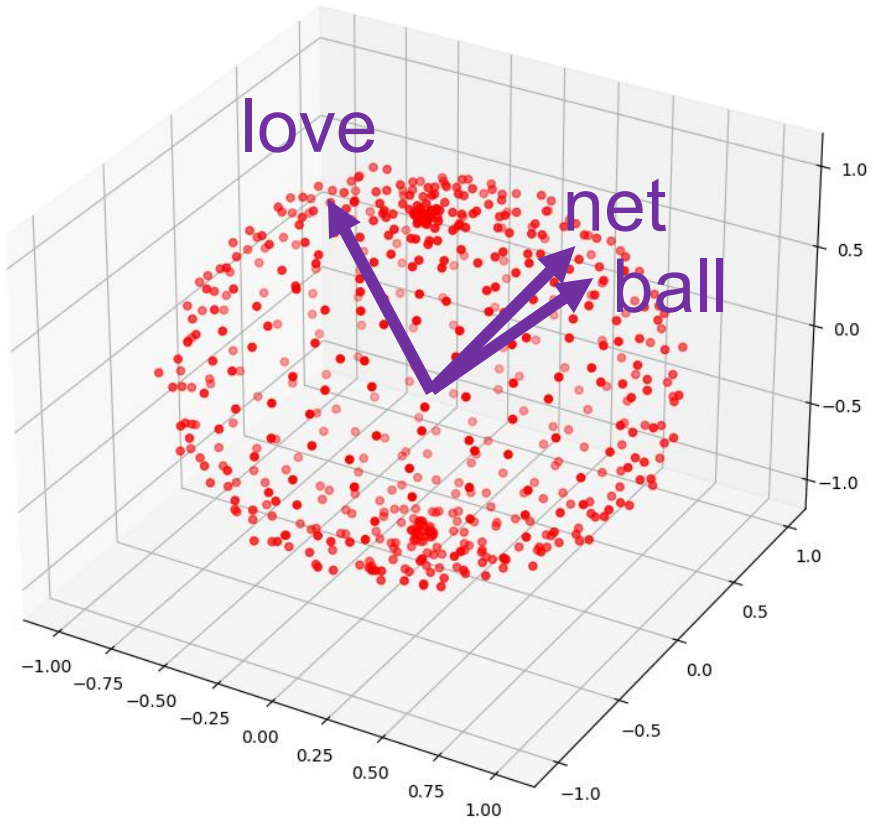
`https://github.com/andylucny/OAI`

Transformer



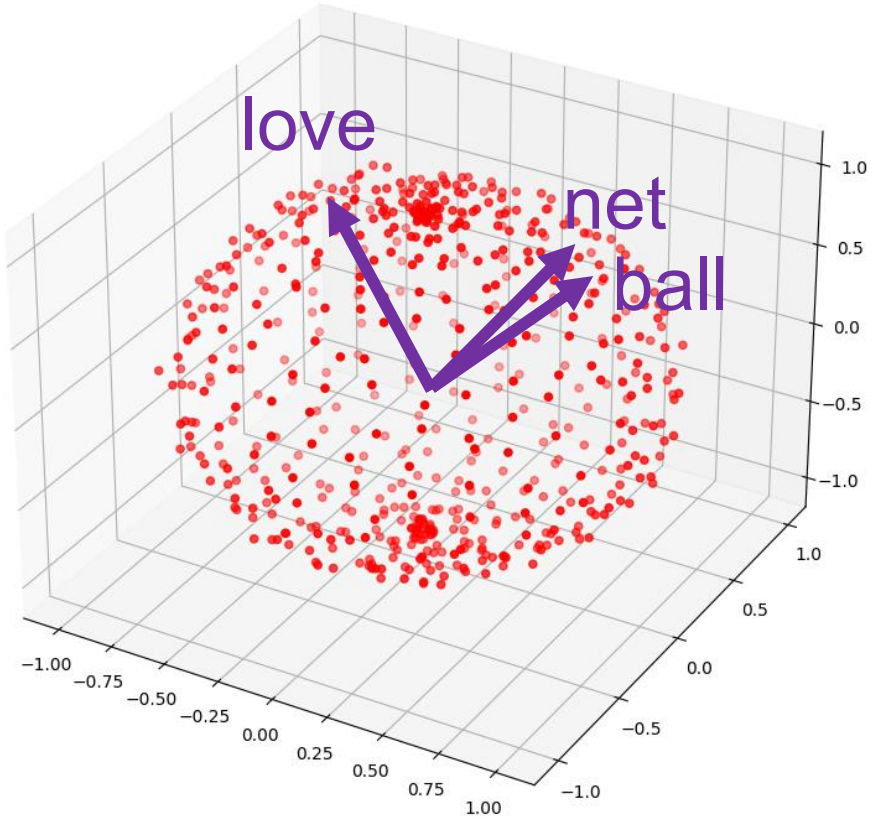
- It gradually transform N hidden states to other N hidden states, N is the size of the input.

Transformer latent spaces



- all have some chosen dimension that determines the number of ways in which two things can differ
- meanings are represented by vectors of this dimension, while their similarity is given by cosine similarity
- vectors (hidden states) also contain information about the position in the input

Cosine similarity



$$sim(\vec{u}, \vec{v}) = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\| \|\vec{v}\|}$$ <-1,1>

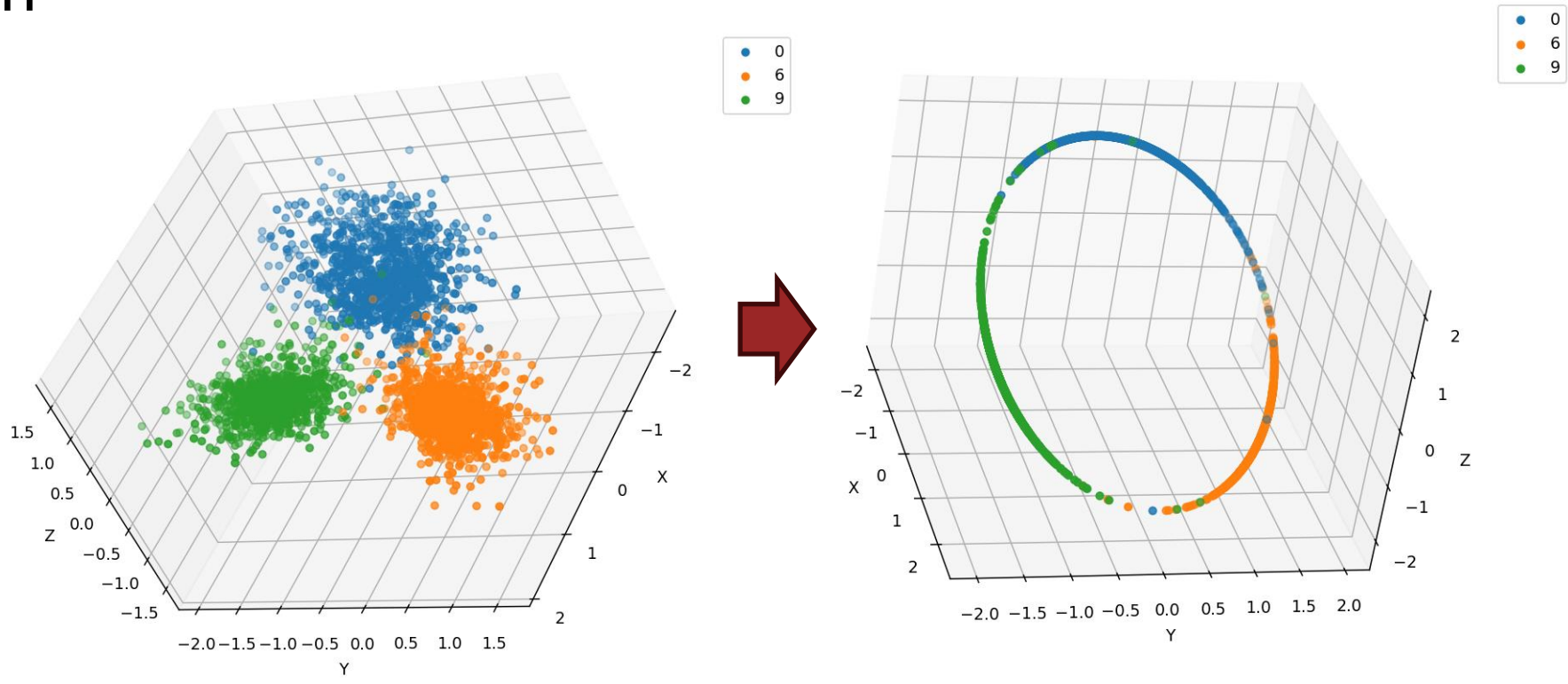
$\cos \phi = 1$ $\cos \phi = 0$ $\cos \phi = -1$

since in the transformer the vectors are of similar size:

$$sim(\vec{u}, \vec{v}) \approx \vec{u} \cdot \vec{v}$$ <- ∞ , ∞ >

Layer Normalization

- layer normalizations, inserted between every two building block, help establish cosine similarity between the data.



$$LN(h) = w\frac{h - \mu_L}{\sqrt{\sigma_L^2 + \epsilon}} + b$$

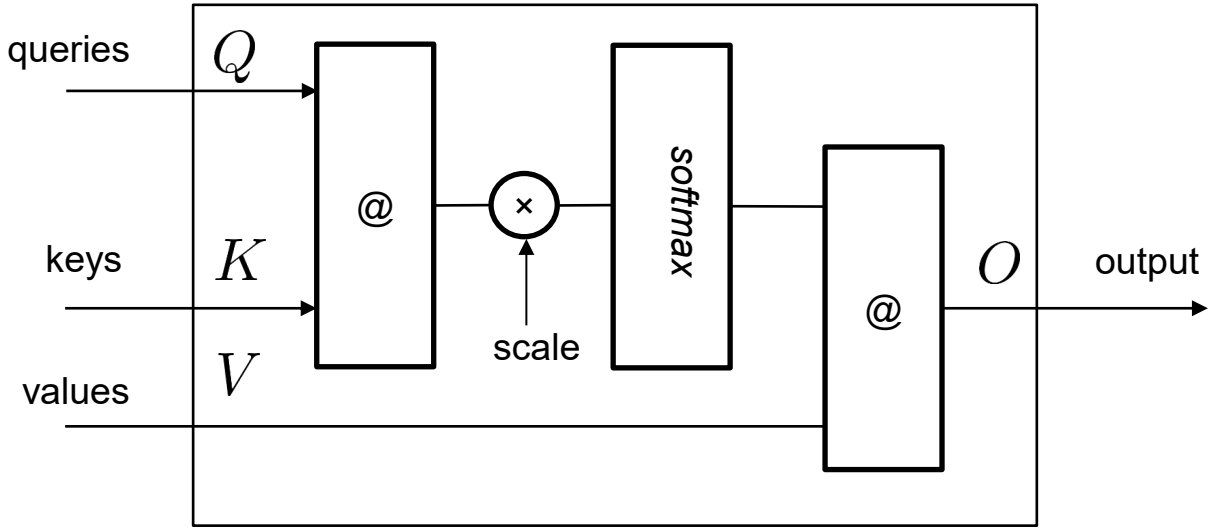
Attention

k – the number of queries (hidden states)

l – the number of the key-value pairs

d – dimension, the number of features

„ try to mix the query from the keys and output the analogous mix of values“



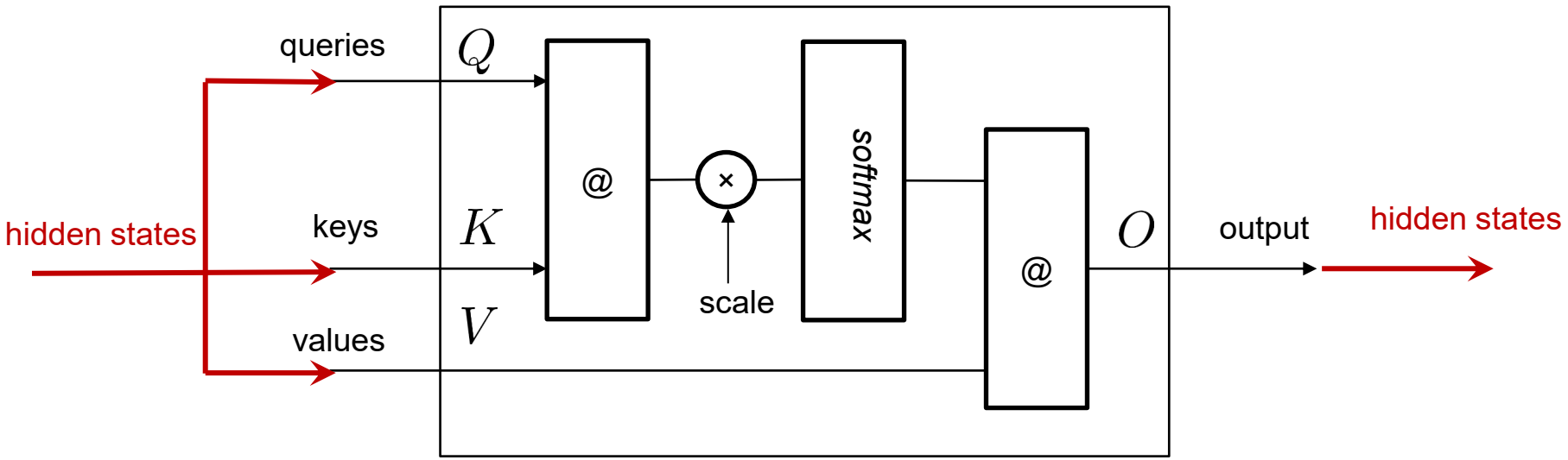
The number of keys and values must be the same

$$O = softmax\left(\frac{QK^T}{\sqrt{d}}\right)V$$

Q: k × d K^T: d × l V : l × d

K: l × d QK^T: k × l O: k × d

Self-Attention



The word ball has different meanings in different contexts

He shot the ball into the net.
(the soccer ball)

The ball tore off his leg.
(the cannonball)

The word ball has different meanings in different contexts

He shot the ball into the net.
(the soccer ball)

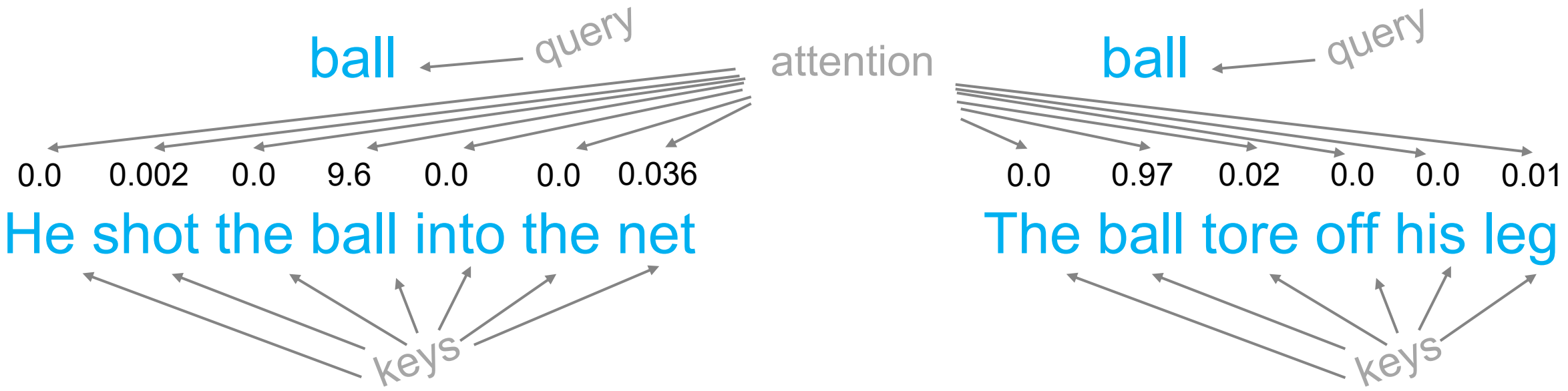
The ball tore off his leg.
(the cannonball)

ball = c_1 ball + c_2 shot + c_3 net

ball = d_1 ball + d_2 tore + d_3 leg

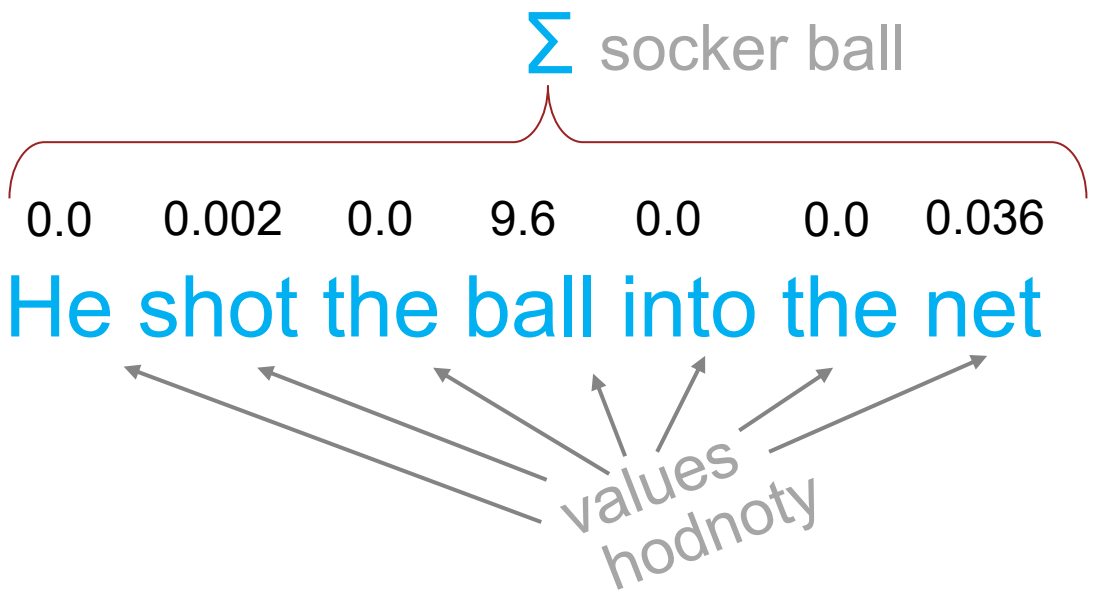
For each word, we calculate the cosine similarity to others (including itself)

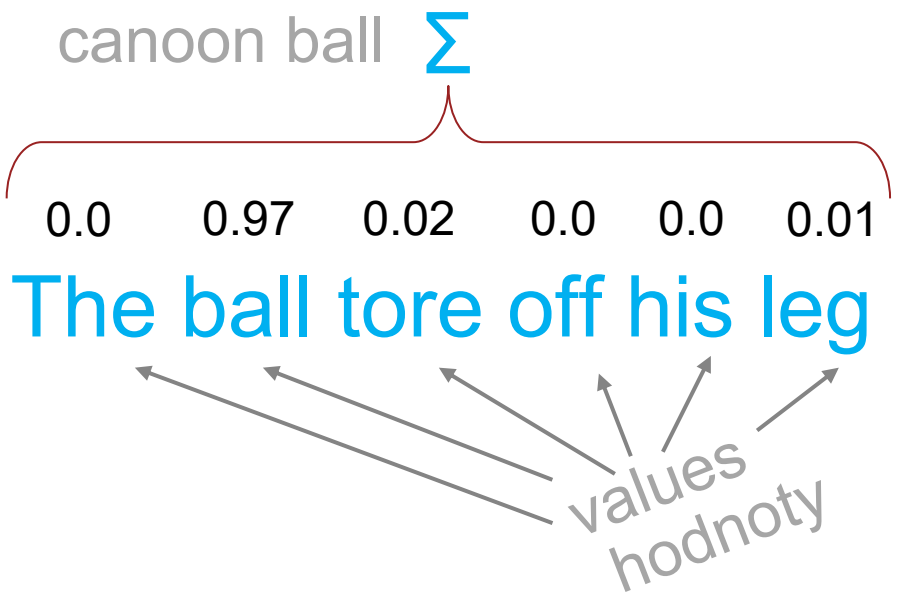
For each word, we calculate the cosine similarity to others (including itself) and apply Softmax



$$softmax_i(x_1, x_2, \ldots, x_n) = \frac{e^{x_i}}{\sum_k e^{x_k}}$$

and then we replace the original representation with a mixture of representations; this gives it a specific meaning





Residual Self-Attention

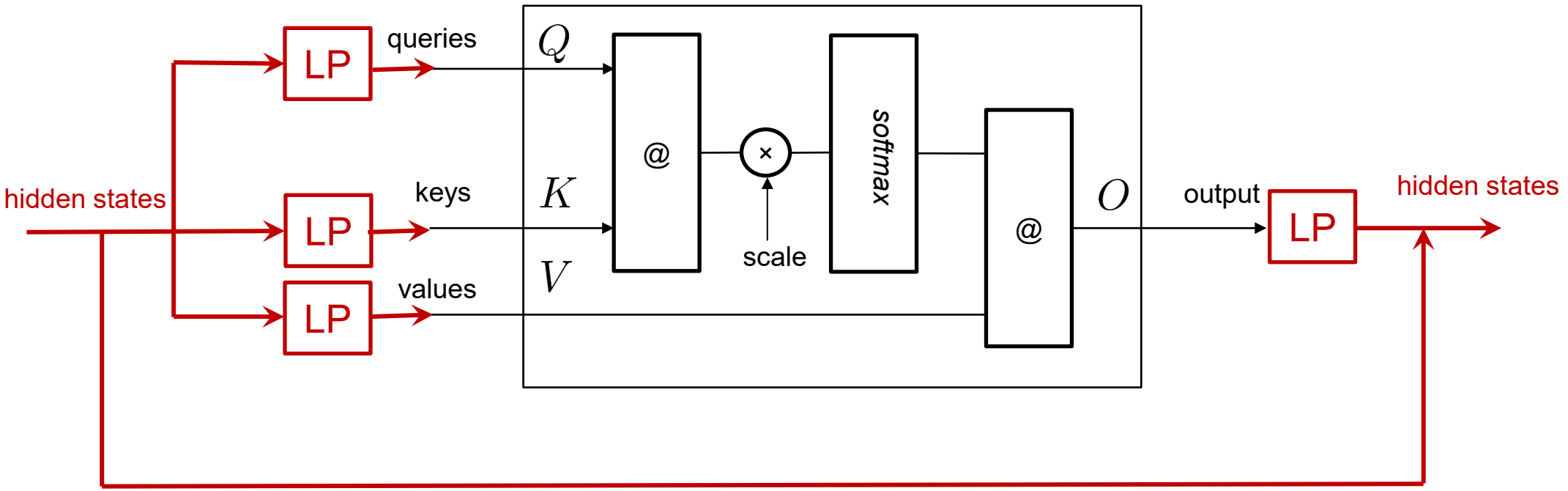
$$ball = c_1\ ball + c_2\ shot + c_3\ net$$

$$c_1 = sim(ball, ball)$$
$$c_2 = sim(ball, shot)$$
$$c_3 = sim(ball, net)$$

$$ball \mathrel{+}= f_o(c_1\ f_v(ball) + c_2\ f_v(shot) + c_3\ f_v(net))$$

$$c_1 = sim(f_q(ball), f_k(ball))$$
$$c_2 = sim(f_q(ball), f_k(shot))$$
$$c_3 = sim(f_q(ball), f_k(net))$$

Residual Self-Attention



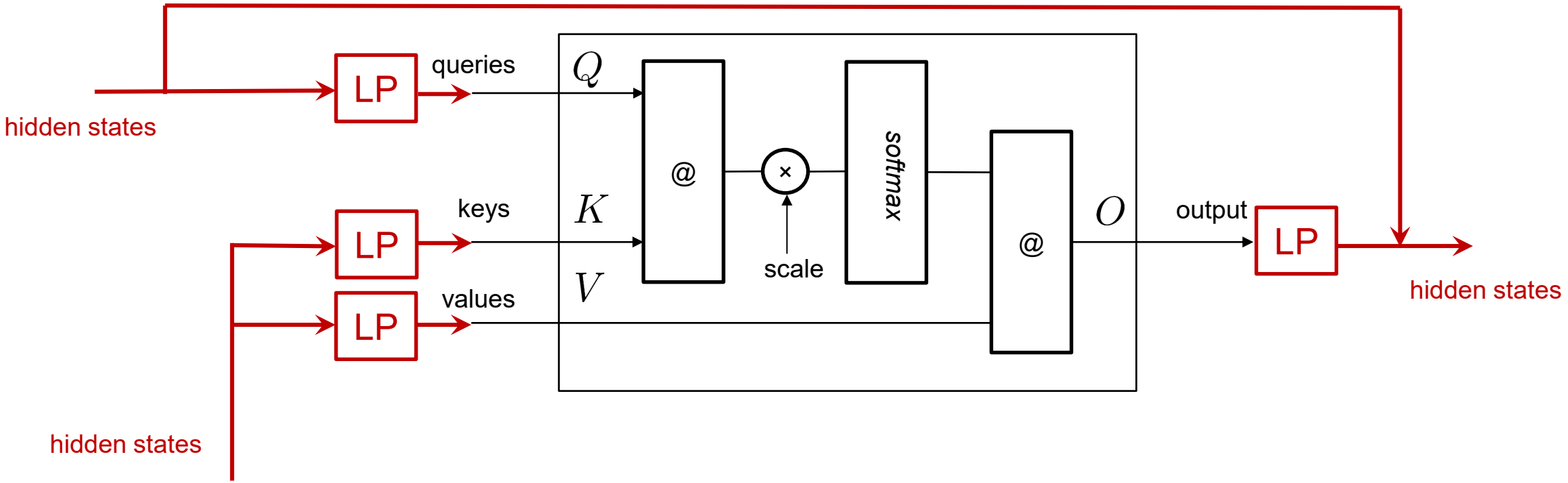
The number of queries, keys and values is the same

Residual Cross-Attention

$$\text{ball} \mathrel{+}= f_o(c_1\, f_v(\text{do}) + c_2\, f_v(\text{you}) + c_3\, f_v(\text{like}) + c_4\, f_v(\text{soccer}))$$

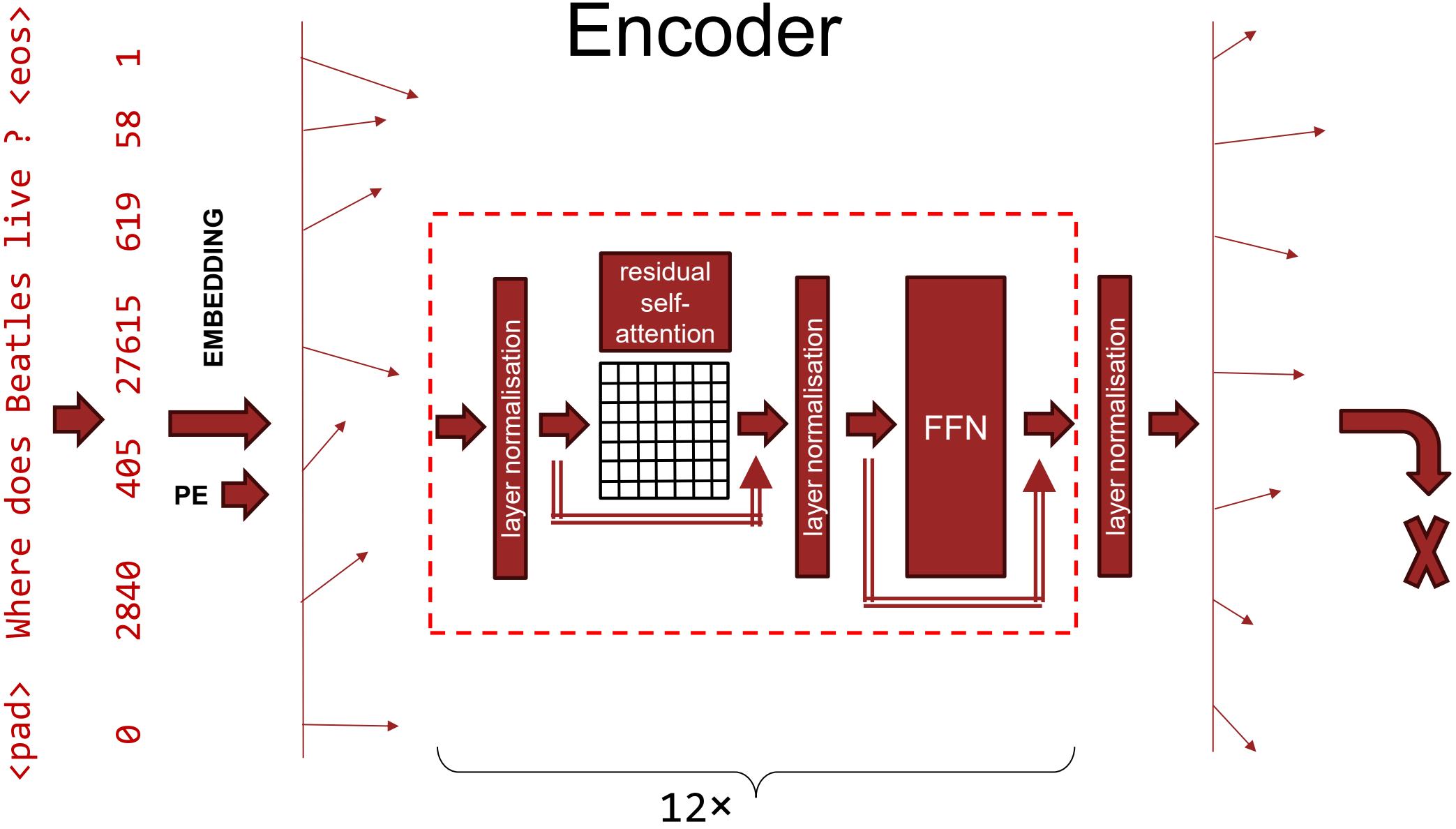
$$c_1 = \text{sim}(f_q(\text{ball}), f_k(\text{do}))$$
$$c_2 = \text{sim}(f_q(\text{ball}), f_k(\text{you}))$$
$$c_3 = \text{sim}(f_q(\text{ball}), f_k(\text{like}))$$
$$c_4 = \text{sim}(f_q(\text{ball}), f_k(\text{soccer}))$$

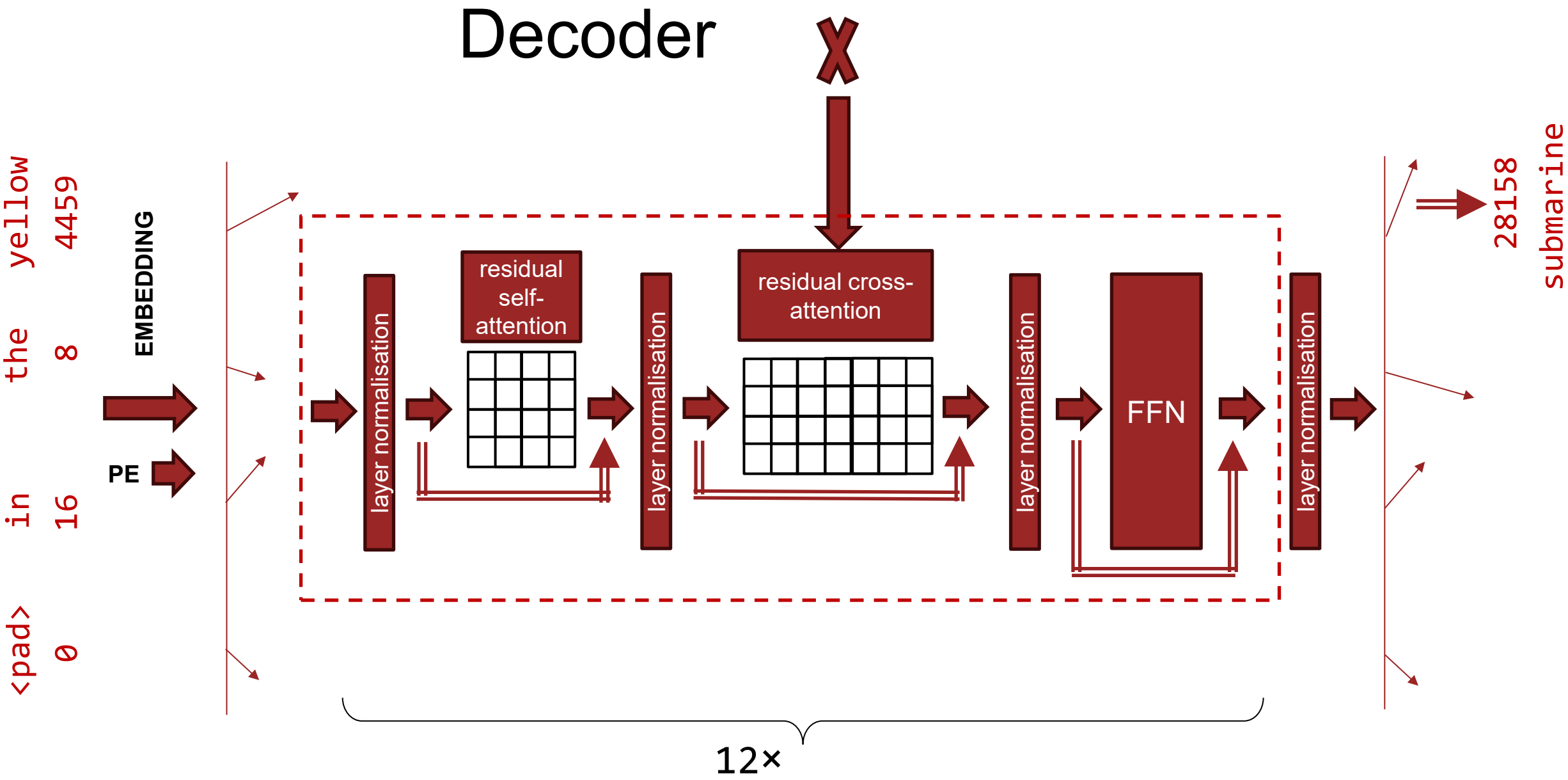
Residual Cross-Attention



The number of queries differs from the number of keys and values

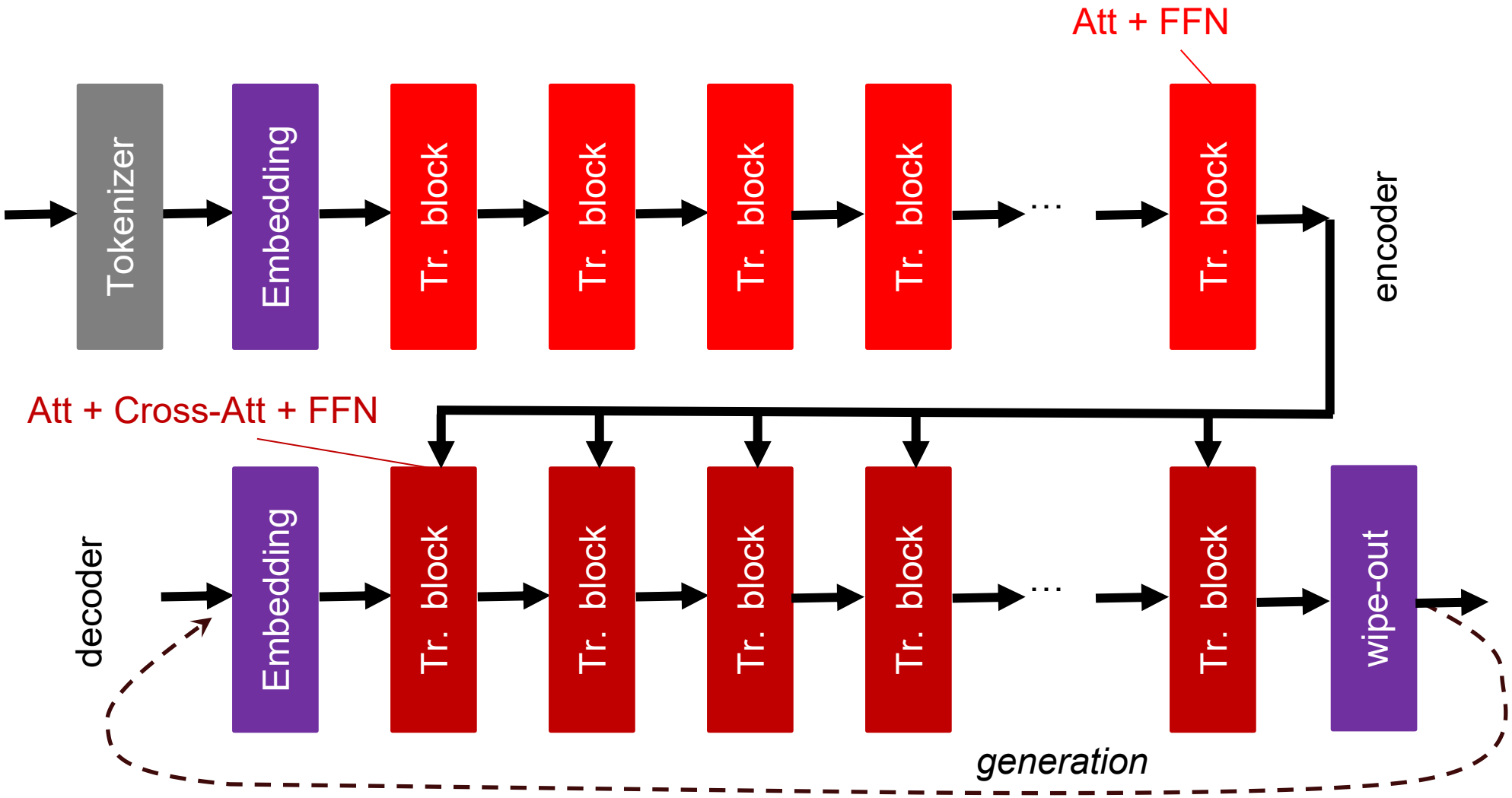
Encoder

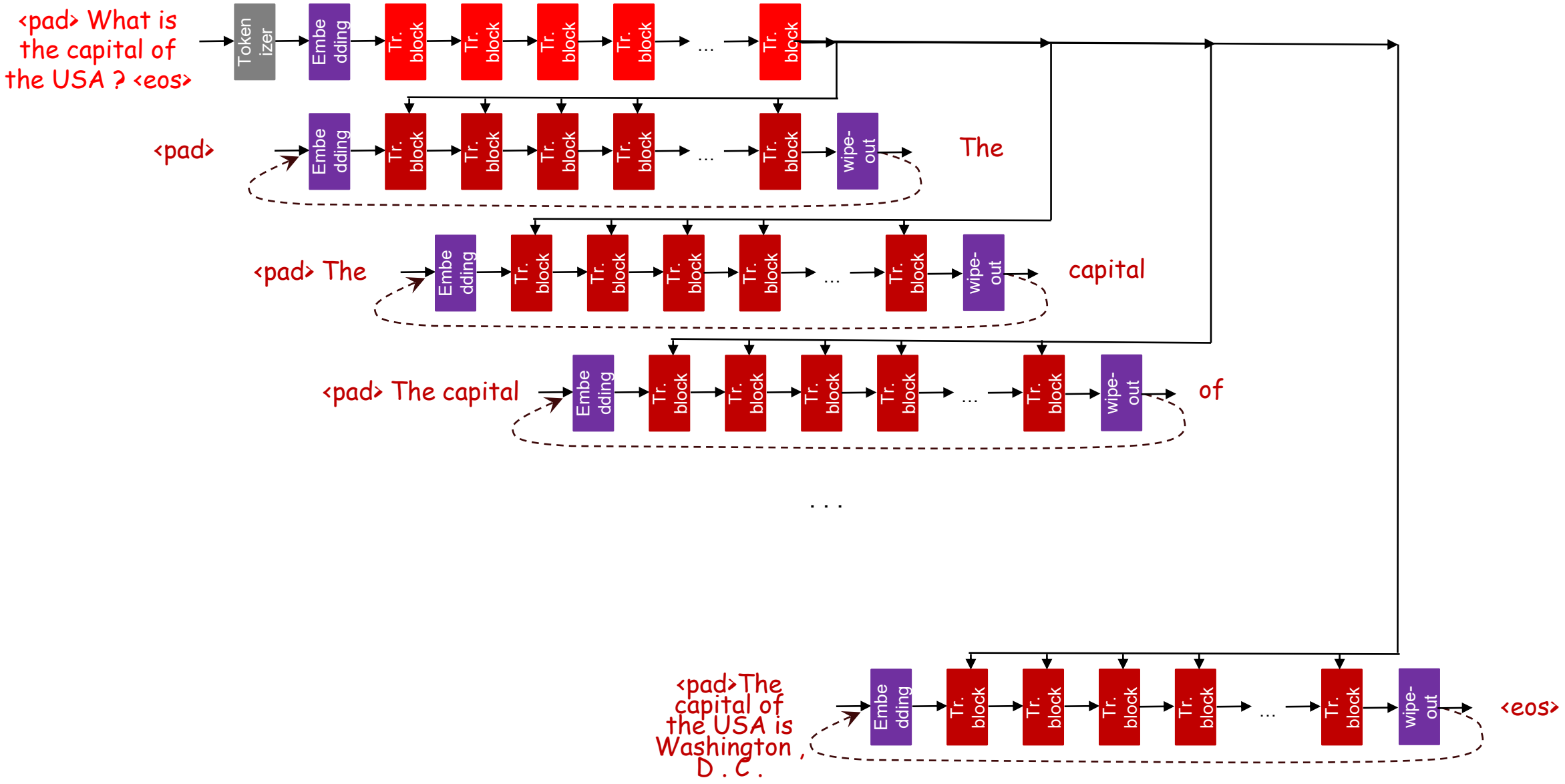

<pad> Where does Beatles live ? <eos>
0 2840 405 27615 619 58 1
EMBEDDING
PE
layer normalisation
residual self-attention
layer normalisation
FFN
layer normalisation
12×


Decoder
X
<pad> in the yellow
0 16 8 4459
EMBEDDING
PE
layer normalisation
residual self-attention
layer normalisation
residual cross-attention
layer normalisation
FFN
layer normalisation
28158
submarine
12×

Chatbot (encoder-decoder type transformer)

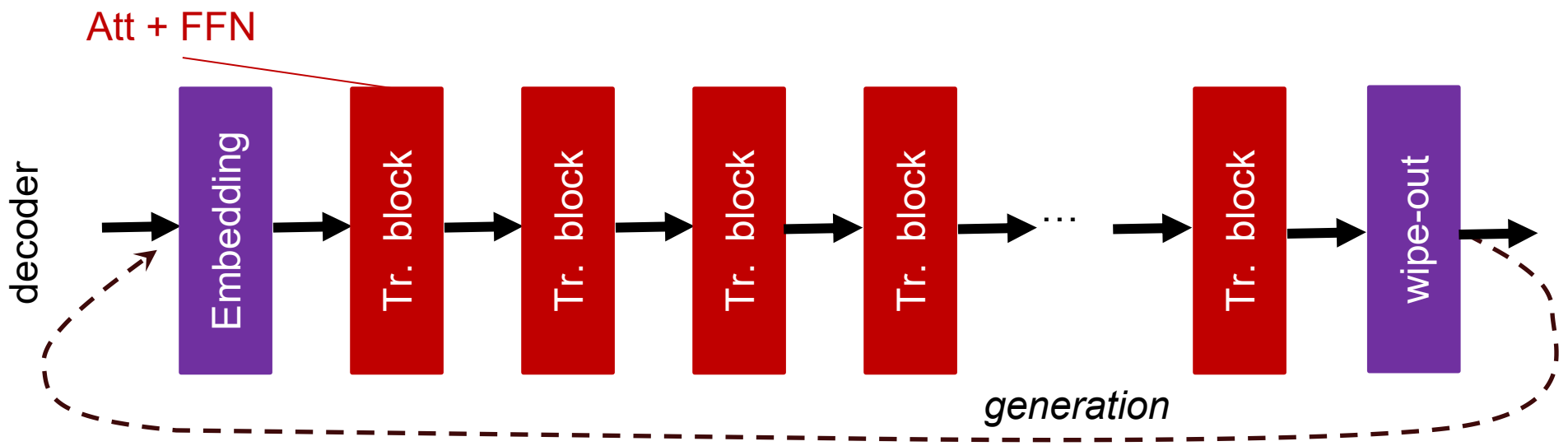
T4, Flann, LaMini



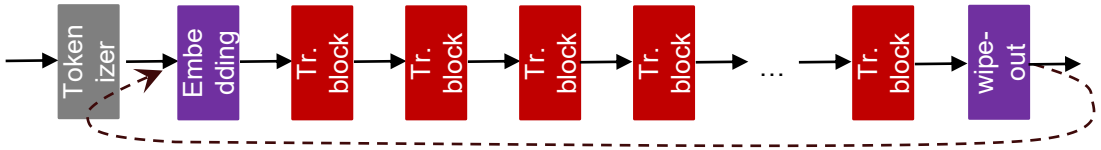

<pad> What is the capital of the USA ? <eos>
Token izer
Embe dding
Tr. block
...
wipe- out
<pad>
The
<pad> The
capital
<pad> The capital
of
. . .
<pad>The capital of the USA is Washington , D . C .
<eos>

Chatbot (decoder-only type transformer)

GPT

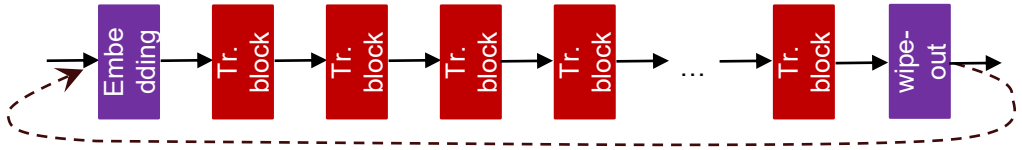


<User:> What is the capital of the USA ? <Assistant:>



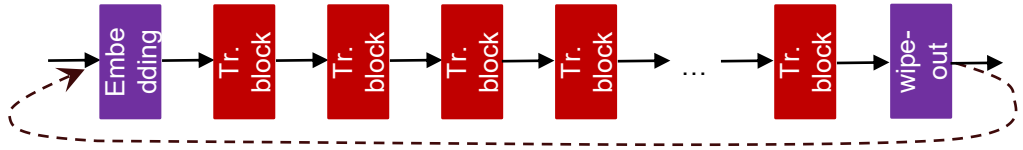
The

<User:> What is the capital of the USA ? <Assistant:> The



capital

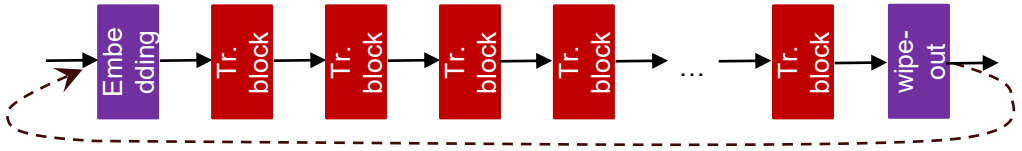
<User:> What is the capital of the USA ? <Assistant:> The capital



of

. . .

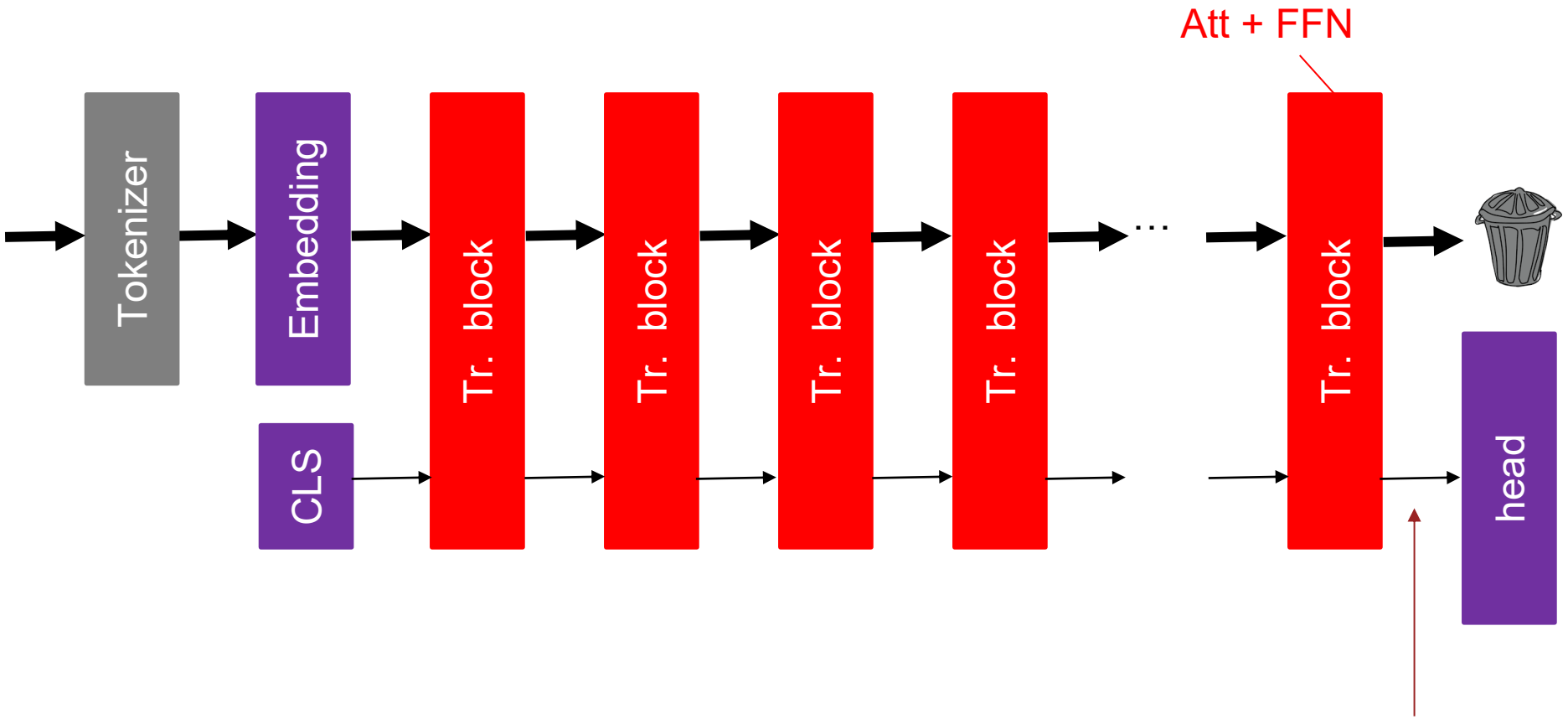
<User:> What is the capital of the USA ? <Assistant:> The capital of the USA is Washington , D . C .



<eos>

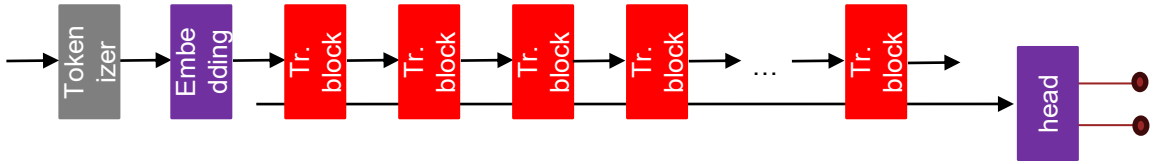
Classifier (encoder-only type transformer)

BERT, RoBERTa

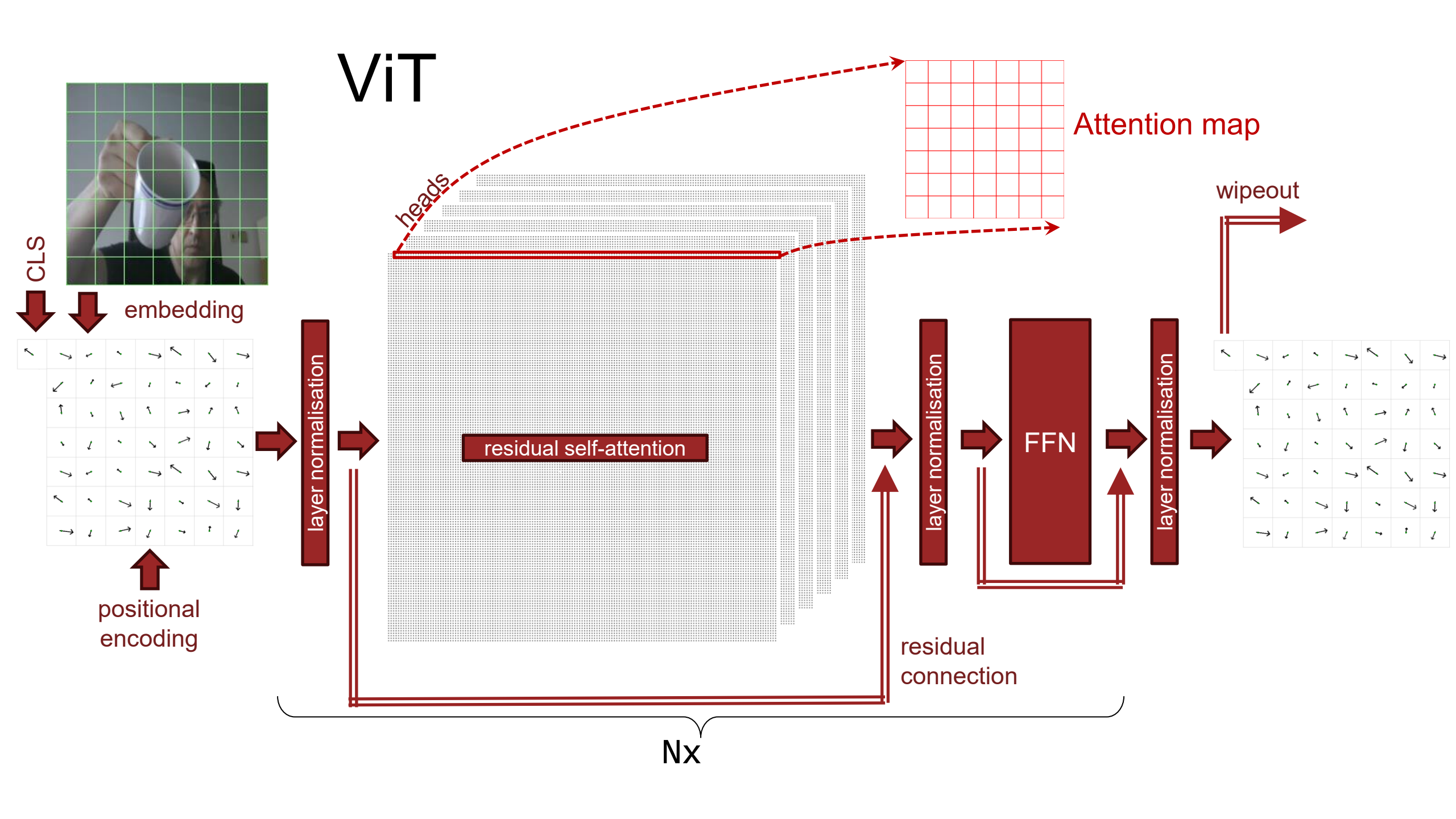


The classification hidden state is a single vector with latent space dimension

<pad> What is the capital of the USA ? <eos>



<a question on the capital cities>
<other question>


ViT
Attention map
heads
wipeout
CLS
embedding
positional
encoding
layer normalisation
residual self-attention
layer normalisation
FFN
layer normalisation
residual
connection
Nx

Detection transformer (DeTR)

- Type: encoder – decoder
- Encoder encode image (ViT without CLS)
- The decoder is called once only
- The decoder emits N (=100) detections into the image and gradually moves them to their correct locations.

How to represent detections

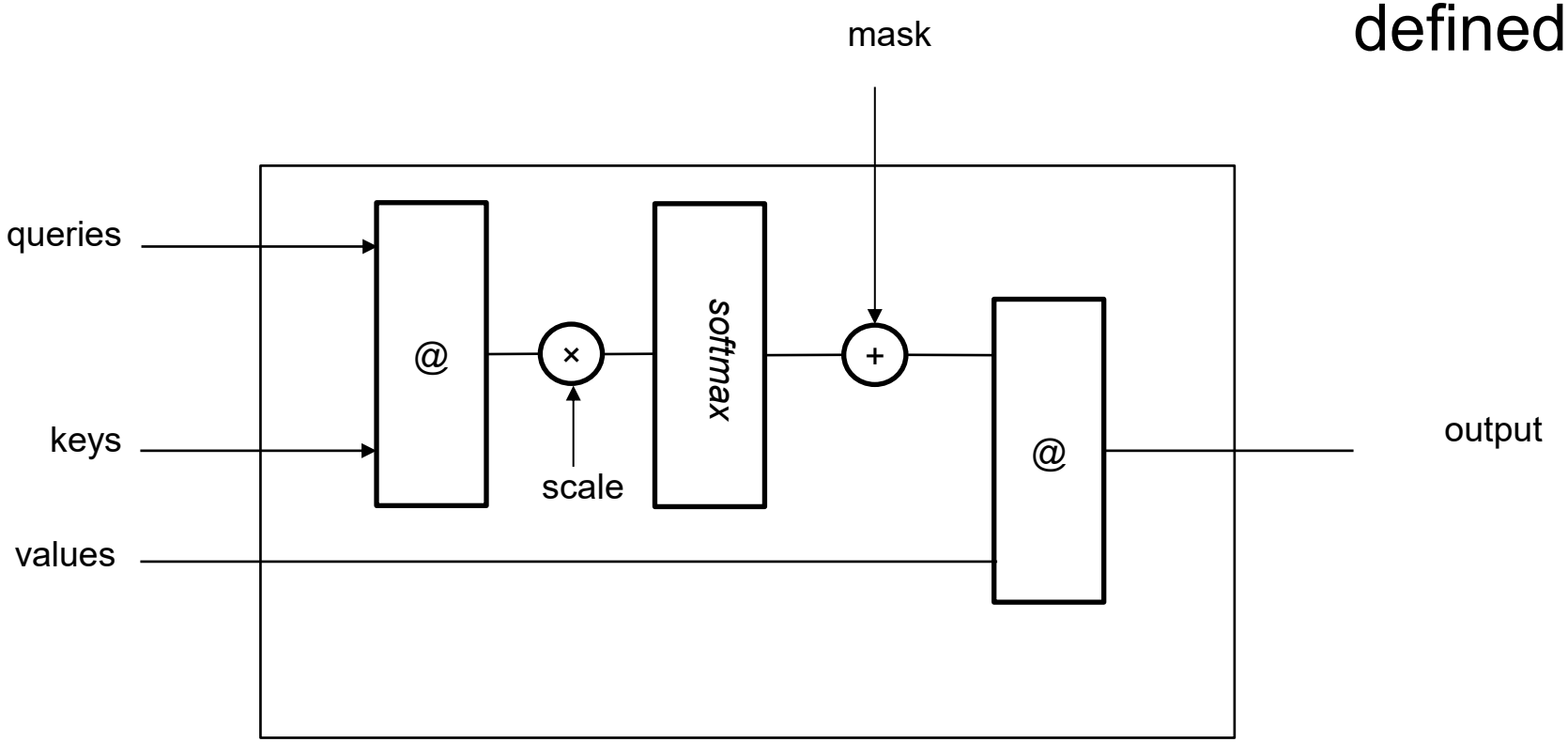
Detection contains:

- Bounding box
- confidence
- class

- DeTR encodes the detection as a vector of length equal to the latent space dimension. How? We leave that to the network itself, let it come up with a representation that suits it.

Masked Attention

- DeTR allows us to search for objects only within a specific region of the image, defined by a mask.



$$Att(Q, K, V) = \left[\text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) + \text{mask}\right] V$$

DeTR

pixel mask
0 .. attend
1.. omit

1 x 800 x 1066

interpolate

pixel masks

1 x 256 x 25 x 34

flatten

1 x 850 x 256

att-mask

640 x 480 x 3

1 x 3 x 800 x 1066

ResNet

features

1 x 2048 x 25 x 34

project

1 x 850 x 256

6x

LN

Attn

LN

FFN

LN

hidden states

1 x 850 x 256

transformer encoder

sinusoidal
positional
encoding

1 x 256 x 25 x 34

flatten

cross

att-mask

zeros

1 x 100 x 256

position embedding

1 x 100 x 256

6x

LN

Attn

LN

Attn

LN

FFN

LN

hidden states

1 x 100 x 256

transformer decoder

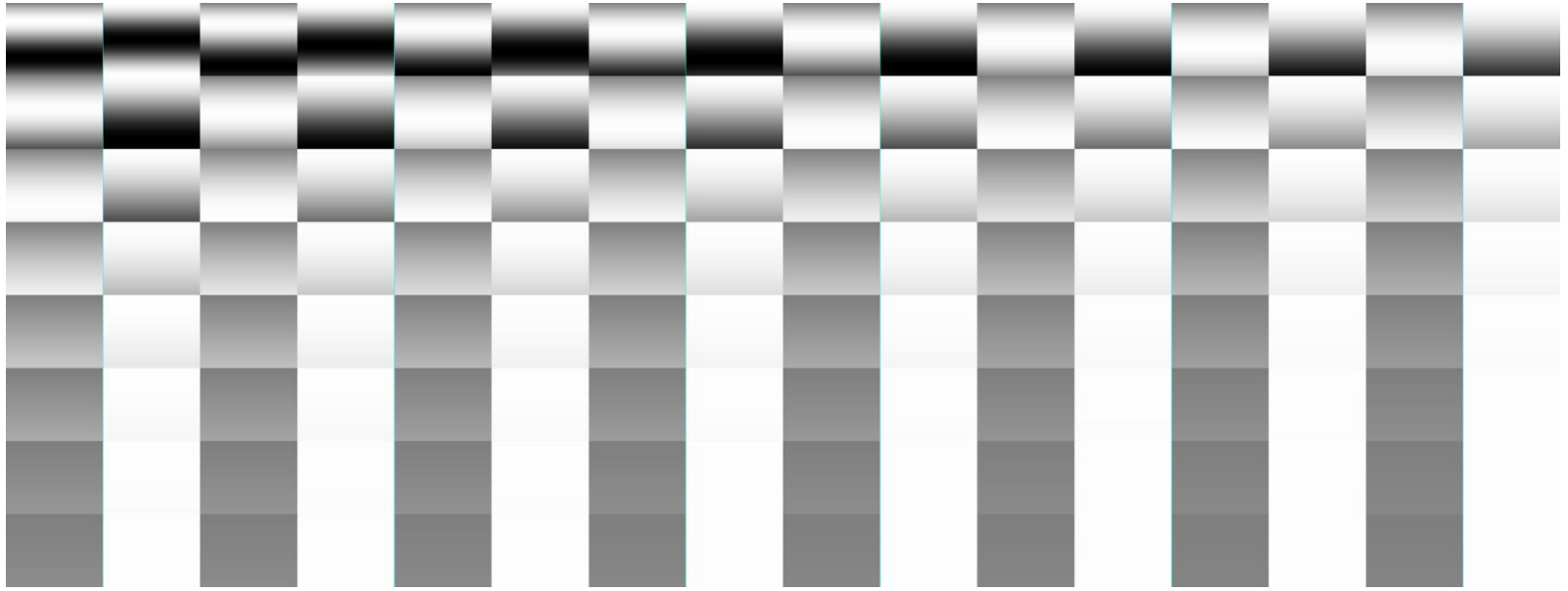
FFN

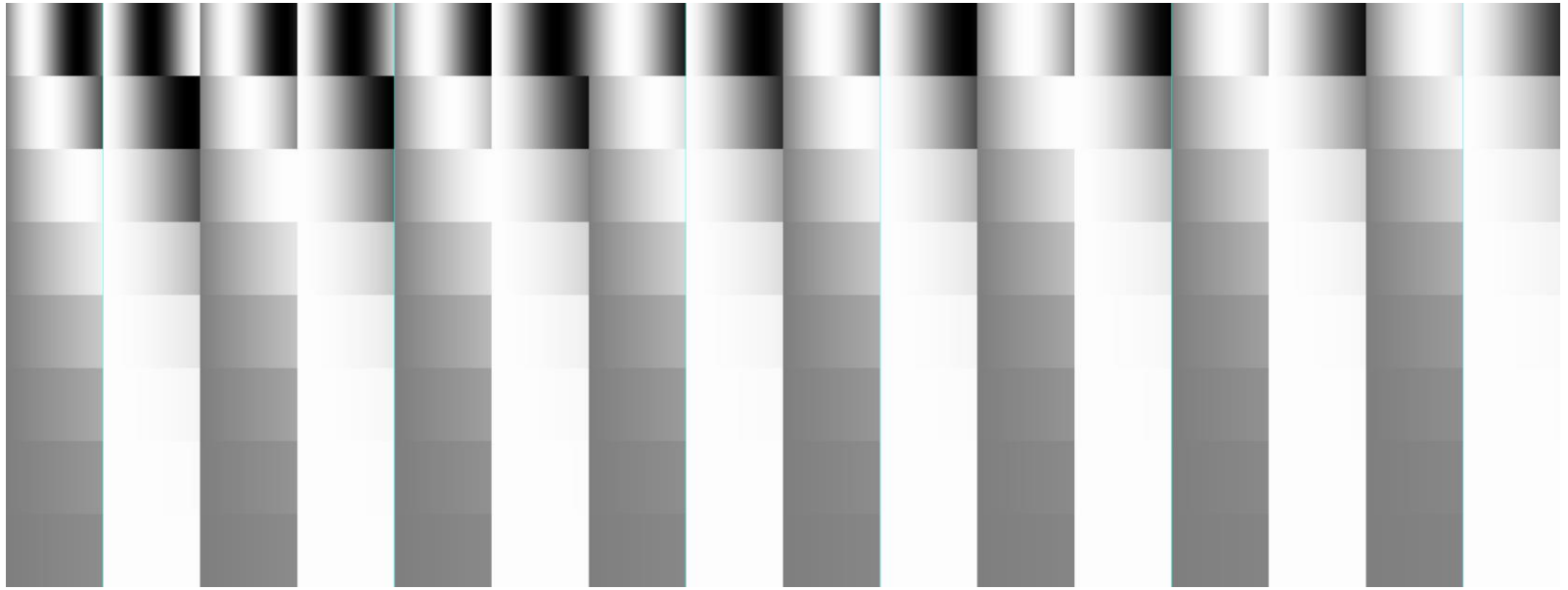
logits

1 x 100 x 92

boxes

1 x 100 x 4

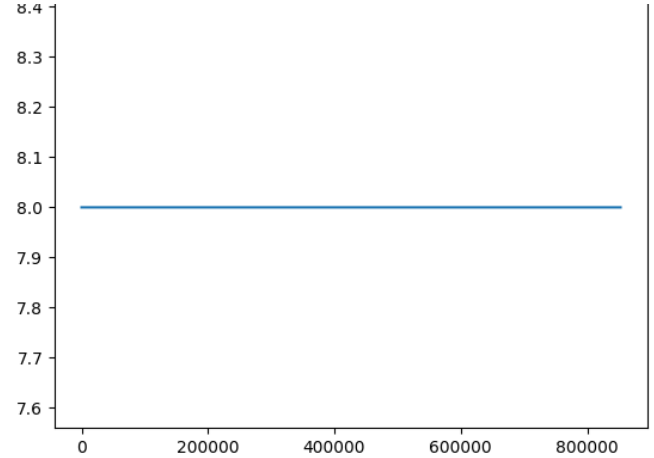




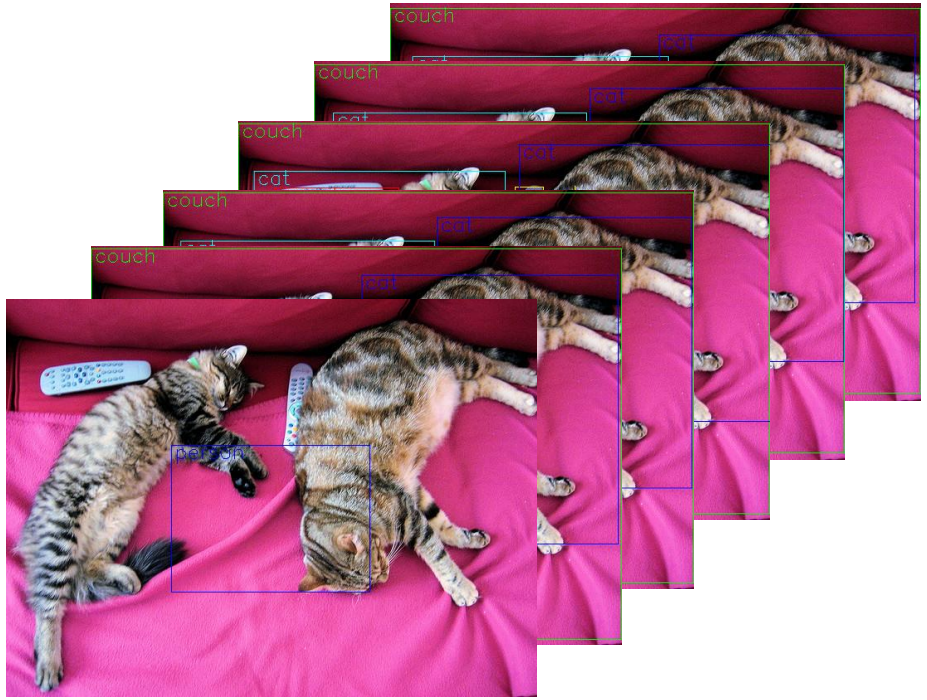
Sinusoidal positional encoding

$$P(k, 2i) = \sin\left(\frac{k}{n^{2i/d}}\right)$$

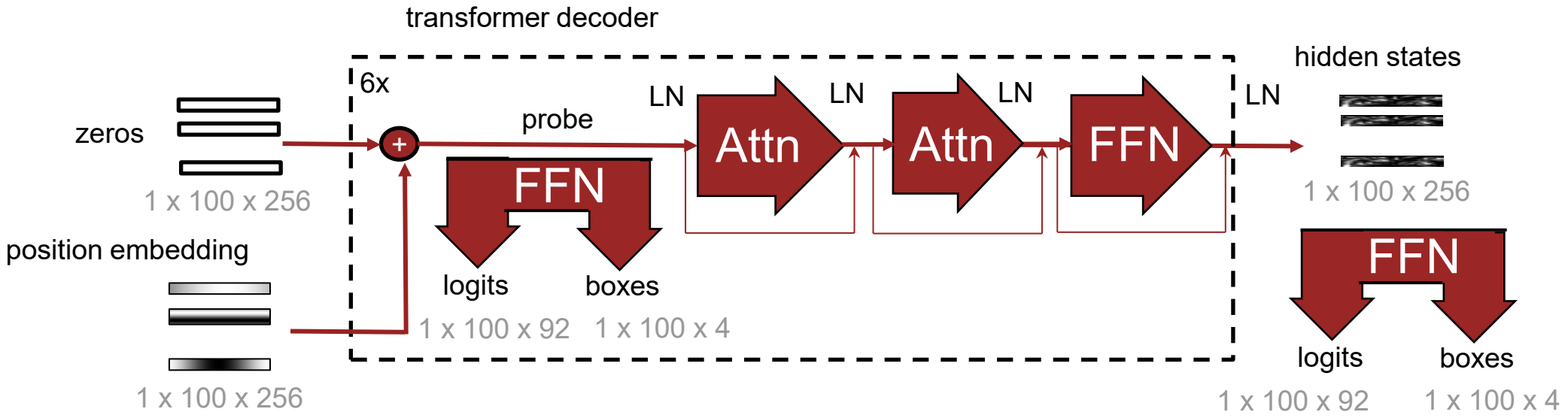
$$P(k, 2i+1) = \cos\left(\frac{k}{n^{2i/d}}\right)$$

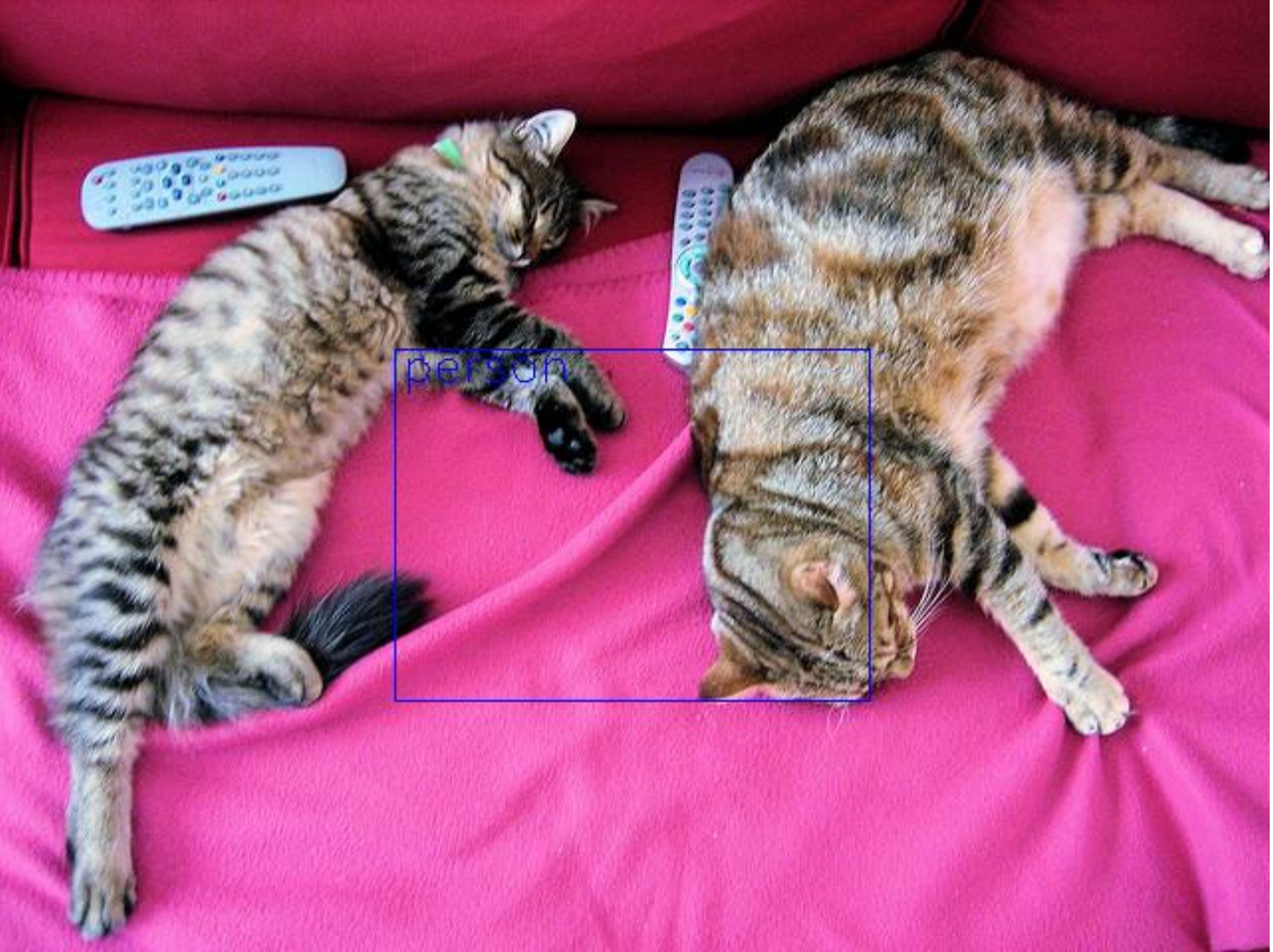


Probing


couch
cat
person


couch
cat
remote

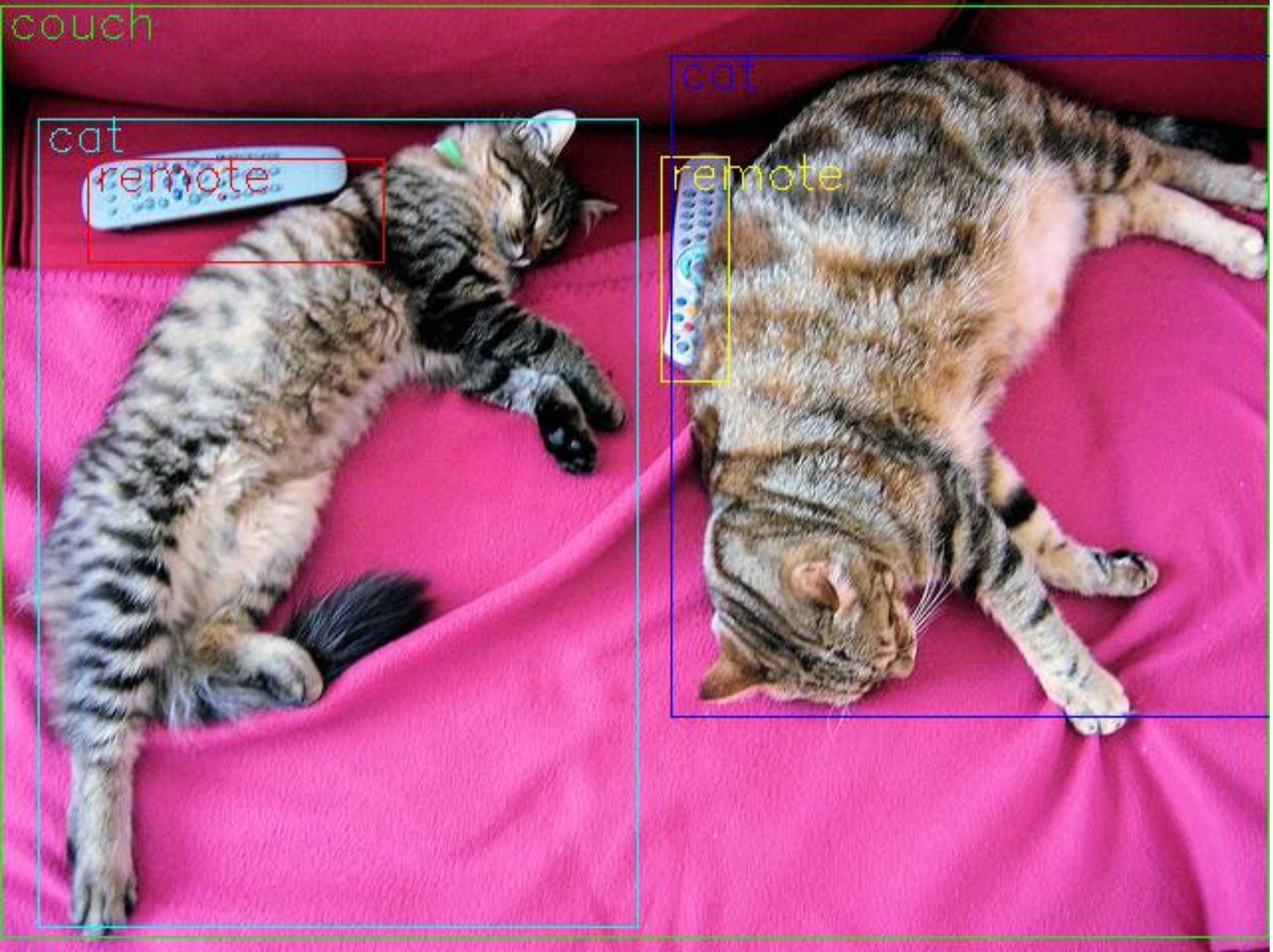

transformer decoder
6x
zeros
1 x 100 x 256
position embedding
1 x 100 x 256
probe
LN
Attn
LN
Attn
LN
FFN
LN
FFN
logits
1 x 100 x 92
boxes
1 x 100 x 4
hidden states
1 x 100 x 256
FFN
logits
1 x 100 x 92
boxes
1 x 100 x 4



0

1

2


couch
cat
remote
cat
remote

4

5


couch
cat
remote
cat
remote

Foundation models

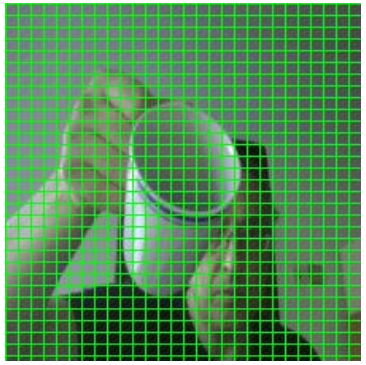
- We train ViT and DeTR for a given set of object categories

- Can we develop models which:
  - classify anything
  - detect anything
  - we do not train, just use (zero-shot),
    or treat a bit only (few shot)

- Why not?
  - we have large dataset
  - we can't annotate them, not only are there so many,
    but we don't know what "anything" means

DINO

DIstillation with NO labels



CLS

embedding

1+28x28=785

12x

attention map

28x28

layer normalisation

residual self-attention

6x 785x785

layer normalisation

FFN

layer normalisation

CLS

384

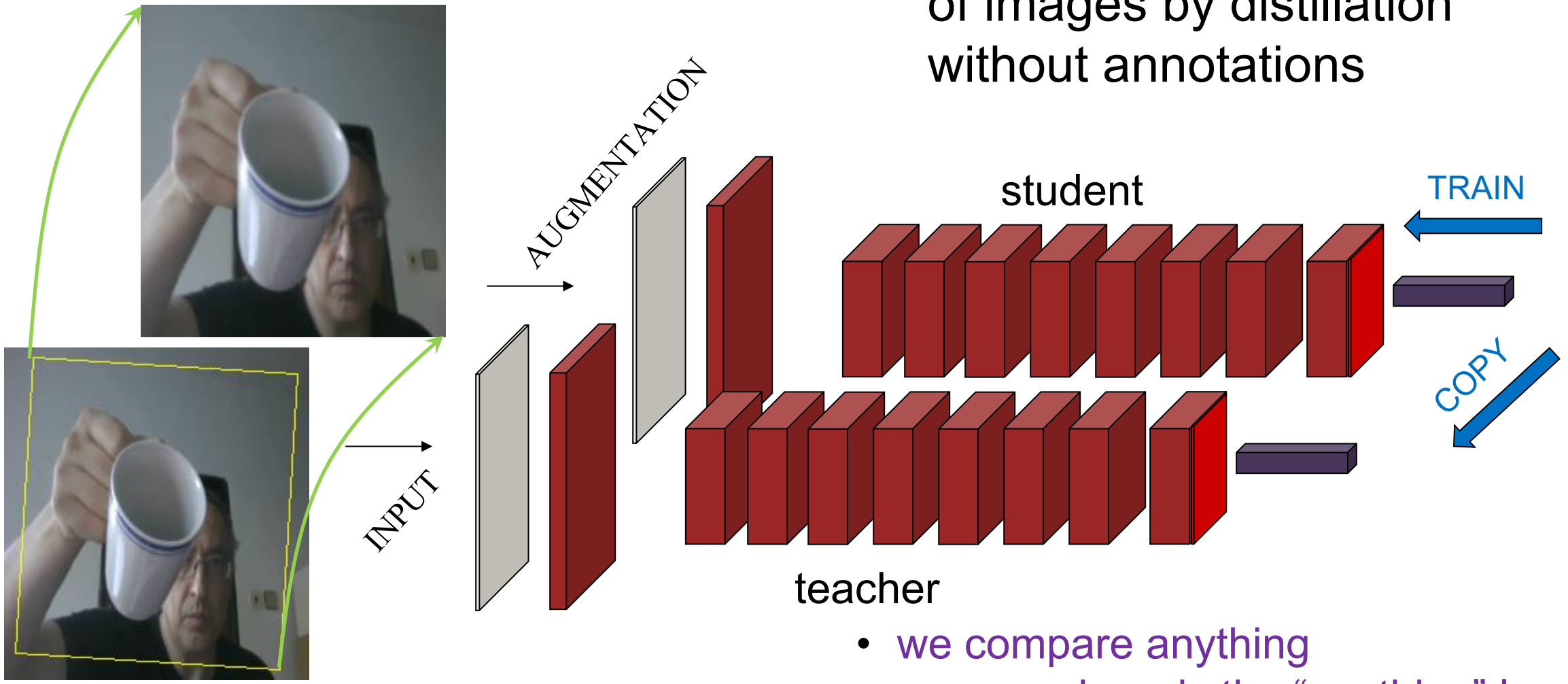
1+28x28=785

no prompt

DINO

DIstillation with
NO labels

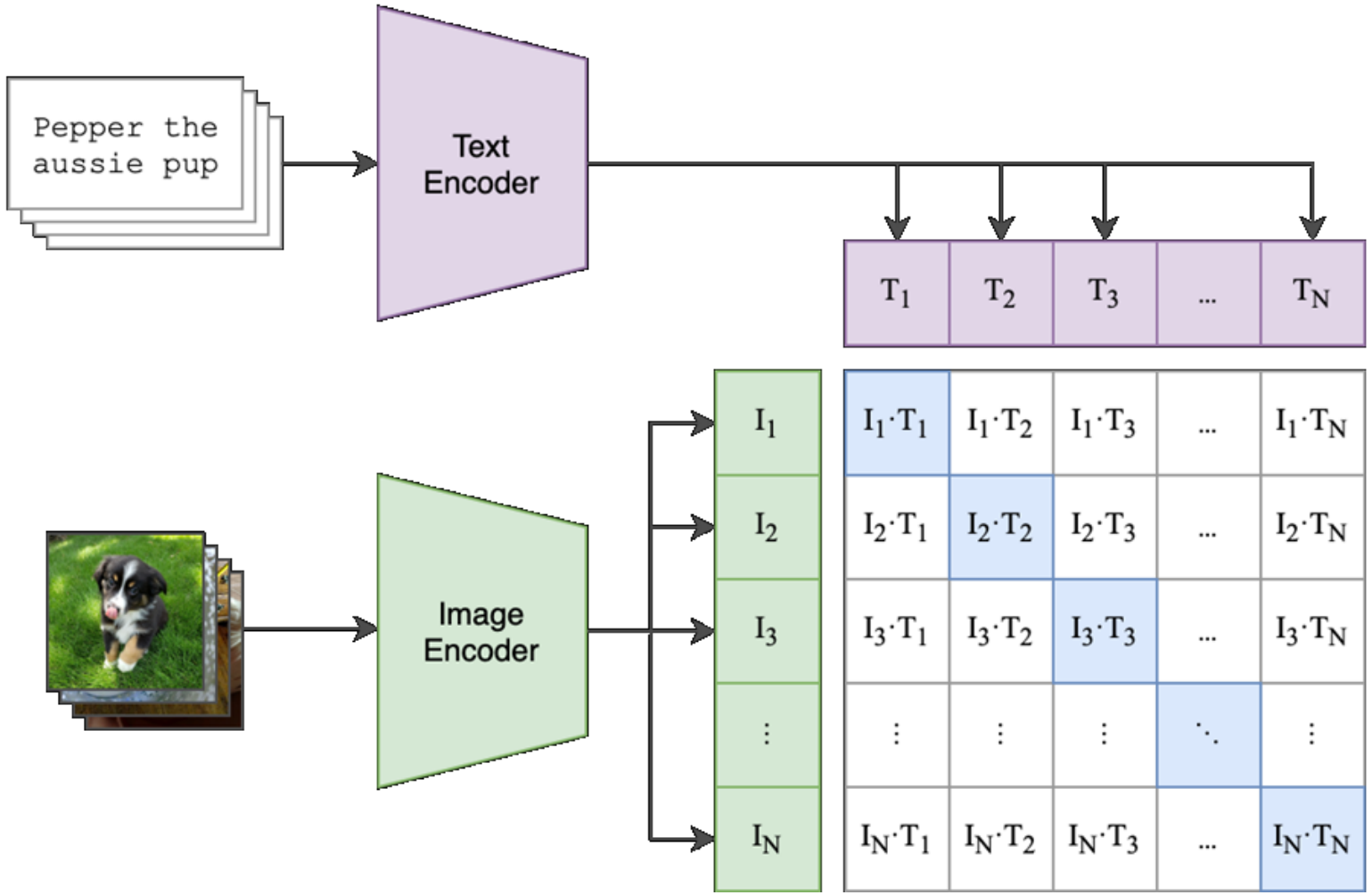
- we train from a large number of images by distillation without annotations



- we compare anything
- we see where is the “anything” located

CLIP

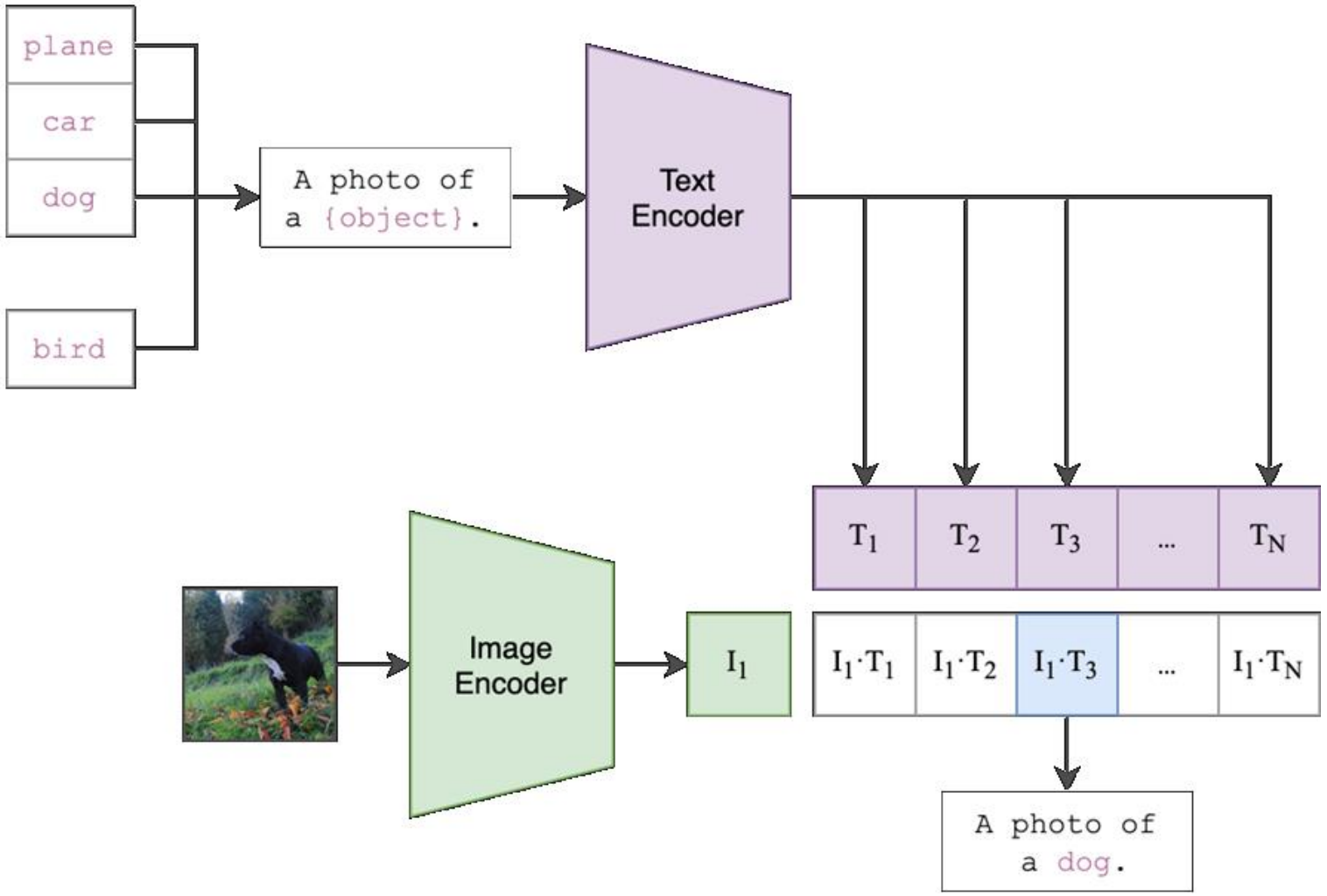
Contrastive Language-Image Pre-training



CLIP is a specially trained visual transformer + text transformer (large language model), where both encoders are coupled by training with a contrastive error function, where we require that the dot product of the codes of matching samples be greater than the codes of those that do not match.

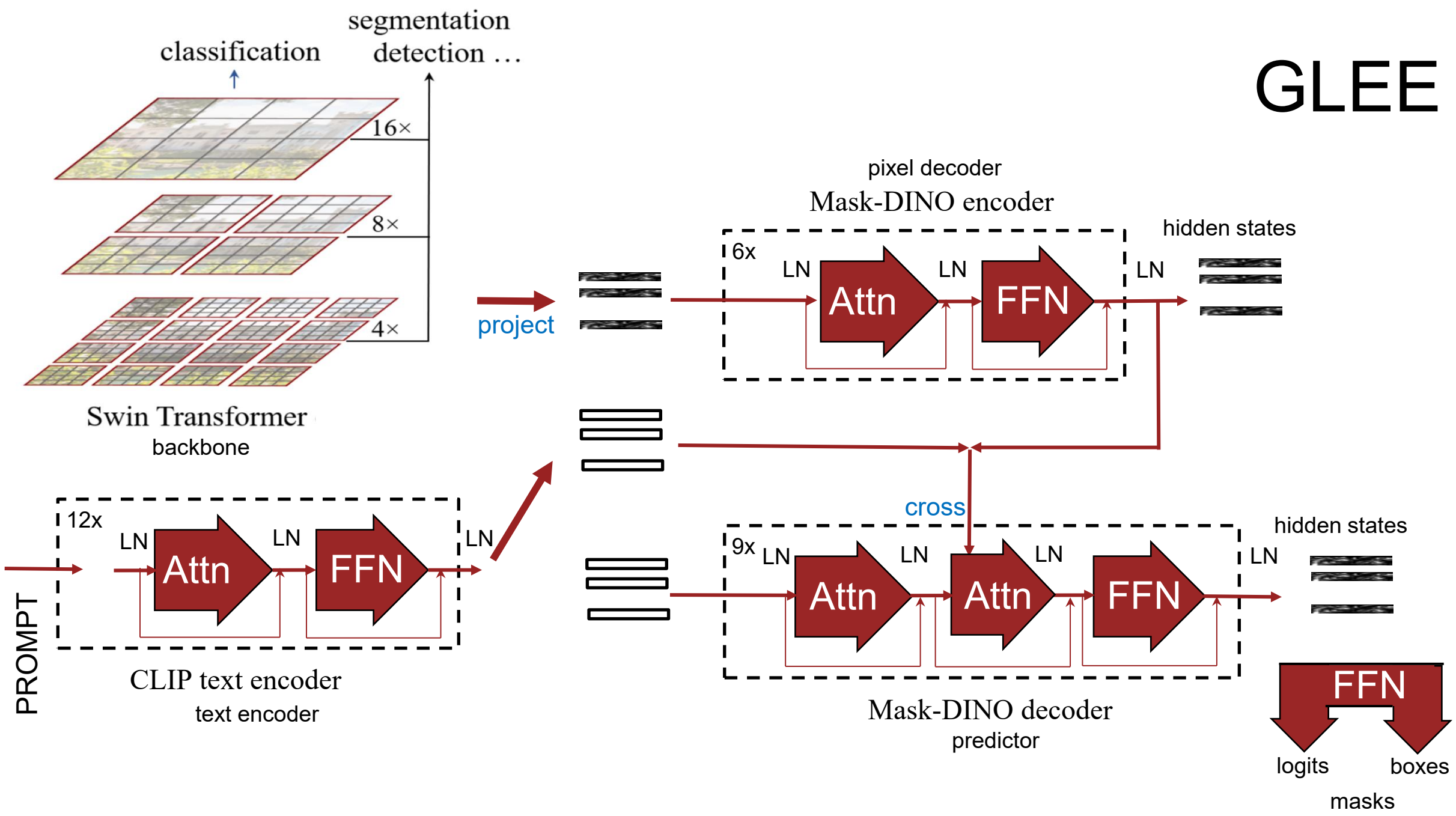
CLIP

Contrastive Language-Image Pre-training



- When using it, we then compare one current code (corresponding to the question asked or the image seen) to a set of previously prepared codes (from texts or images).
- Where the largest scalar product is, that is the pair that corresponds (there must be one).
- we can classify anything we can name
- we can distinguish between text-specified categories

GLEE


classification
segmentation
detection …
16×
8×
4×
Swin Transformer
backbone
project
pixel decoder
Mask-DINO encoder
6x
LN
Attn
LN
FFN
LN
hidden states
cross
12x
LN
Attn
LN
FFN
LN
PROMPT
CLIP text encoder
text encoder
9x
LN
Attn
LN
Attn
LN
FFN
LN
hidden states
Mask-DINO decoder
predictor
FFN
logits
boxes
masks

GLEE

The first dog from the right

- Segment anything that we can describe by text, prompt by points, or scribble

SAM3 Model Architecture with Full LoRA Adaptation