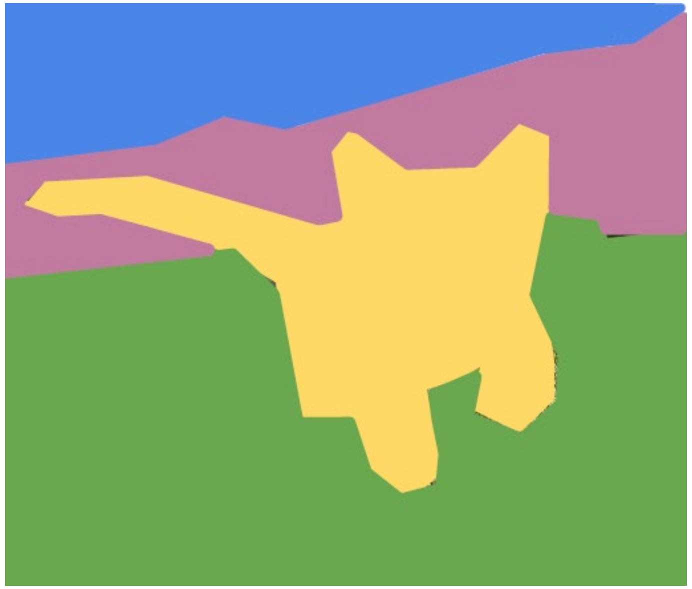
DEEP LEARNING APPLICATIONS

Computer Vision Tasks

Image Classification

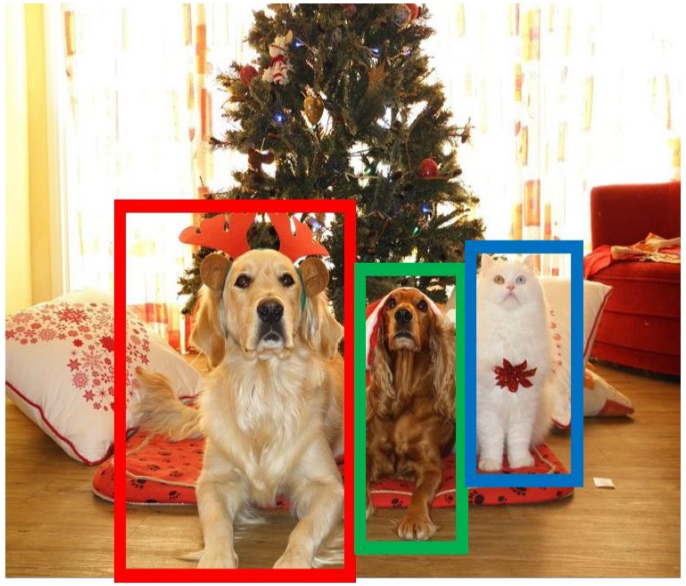
CAT

Semantic Segmentation



CAT, GRASS, TREE, SKY

Object Detection



DOG, DOG, CAT

Image Classification

Given a set of discrete labels

(dog, cat, truck, plane, ...)

→ **CAT**

Image Classification

Image Classification

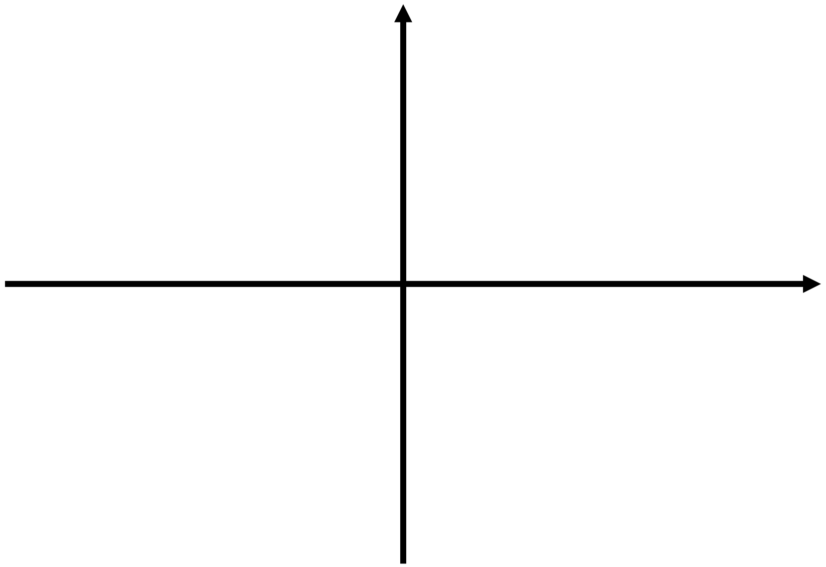


Image Classification

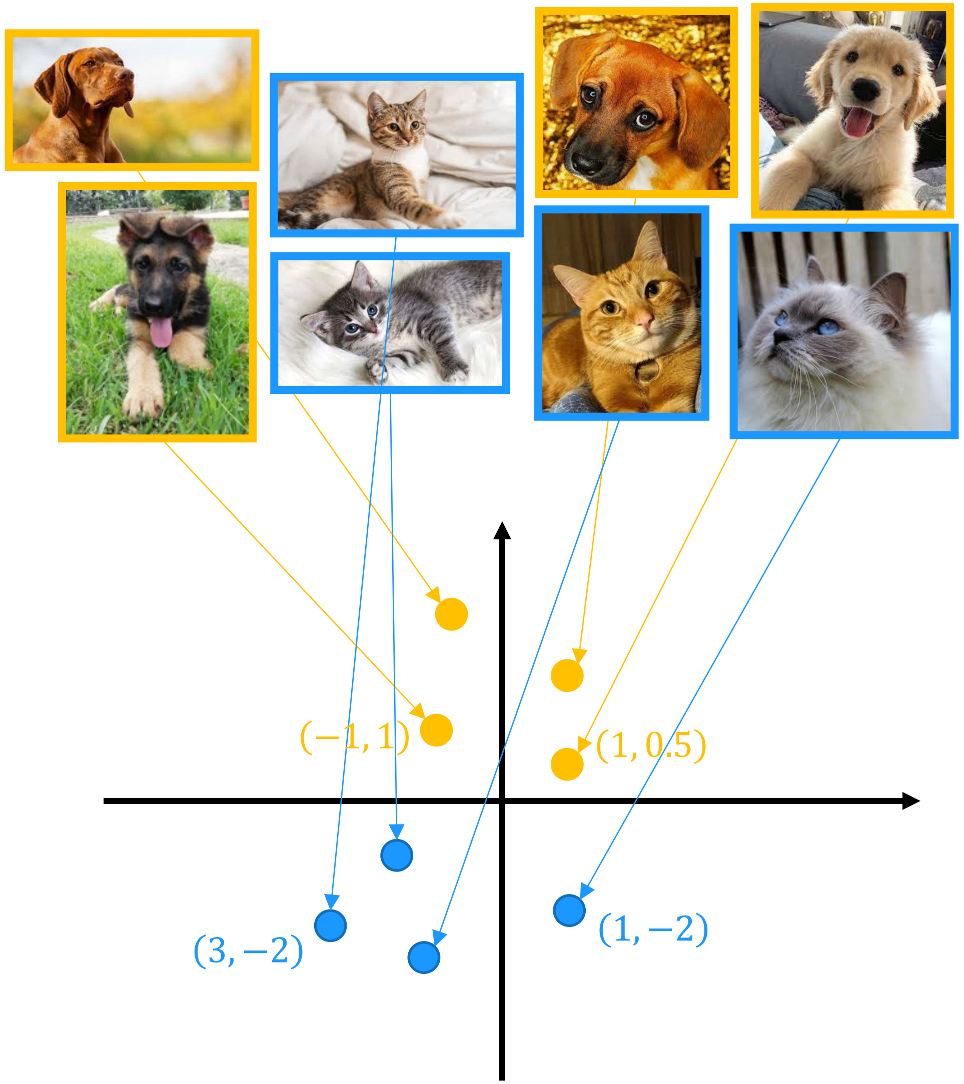
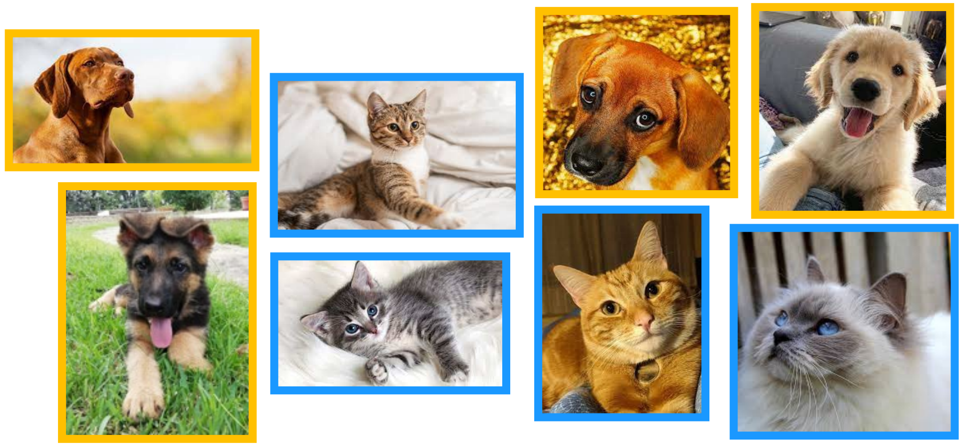


Image Classification



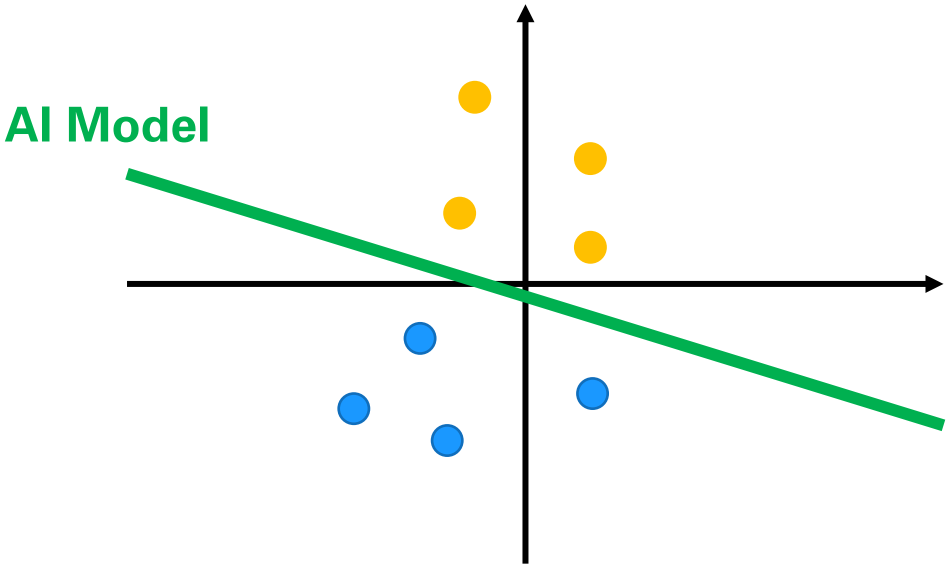
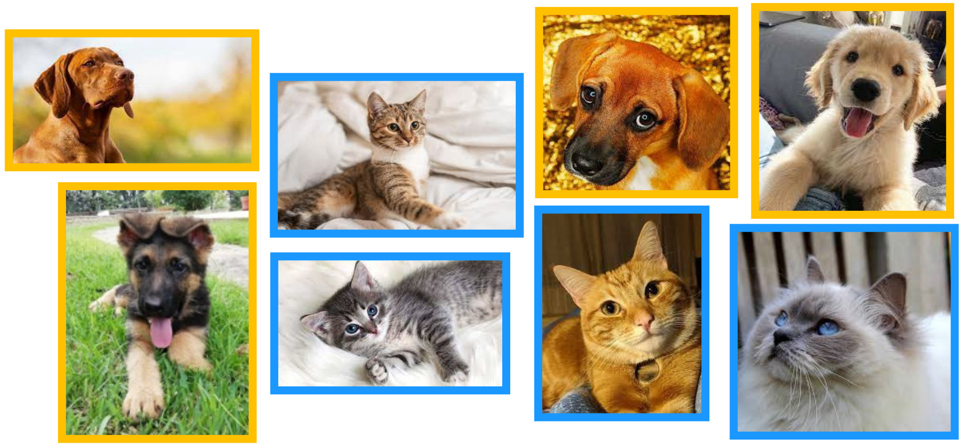


Image Classification



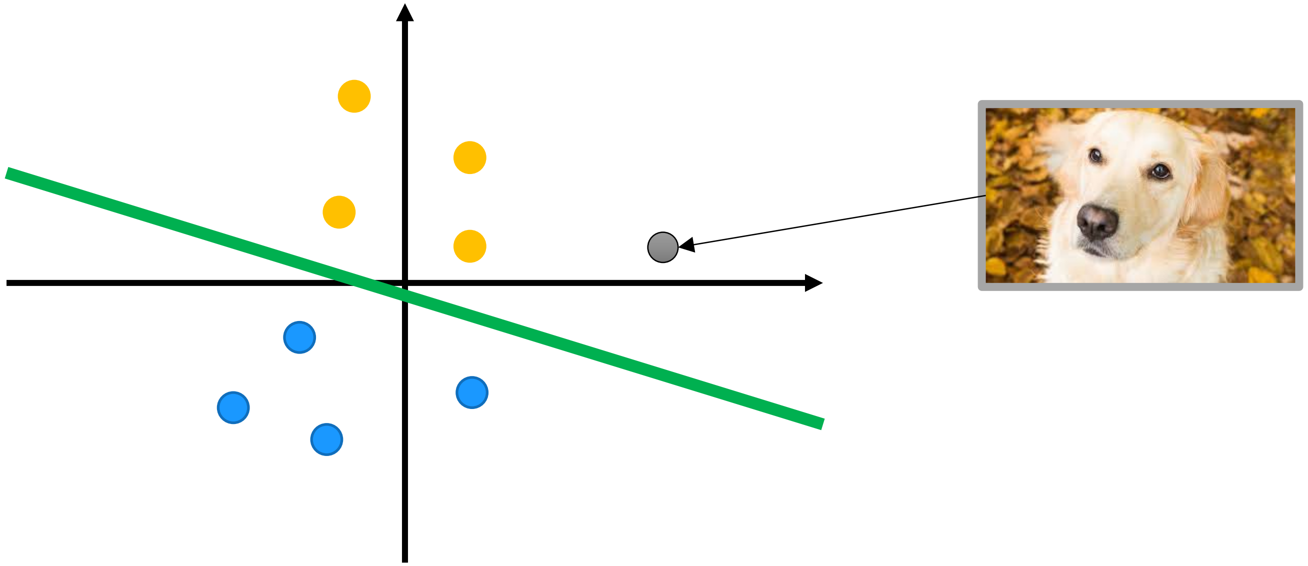
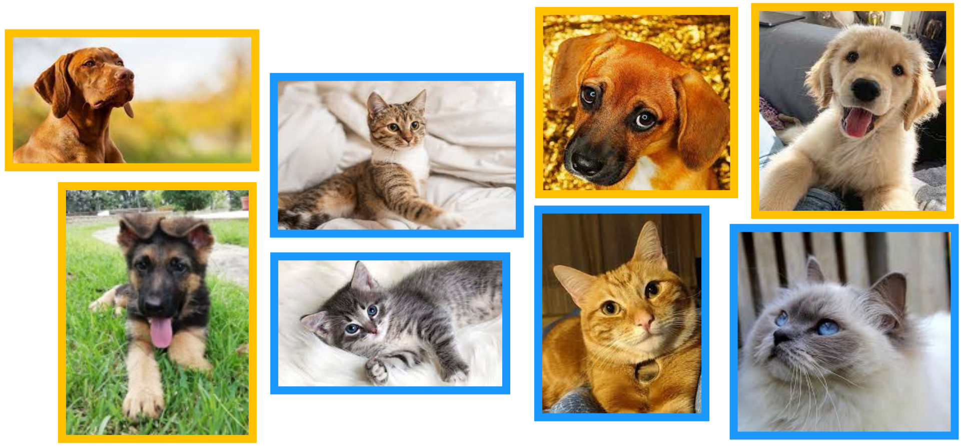
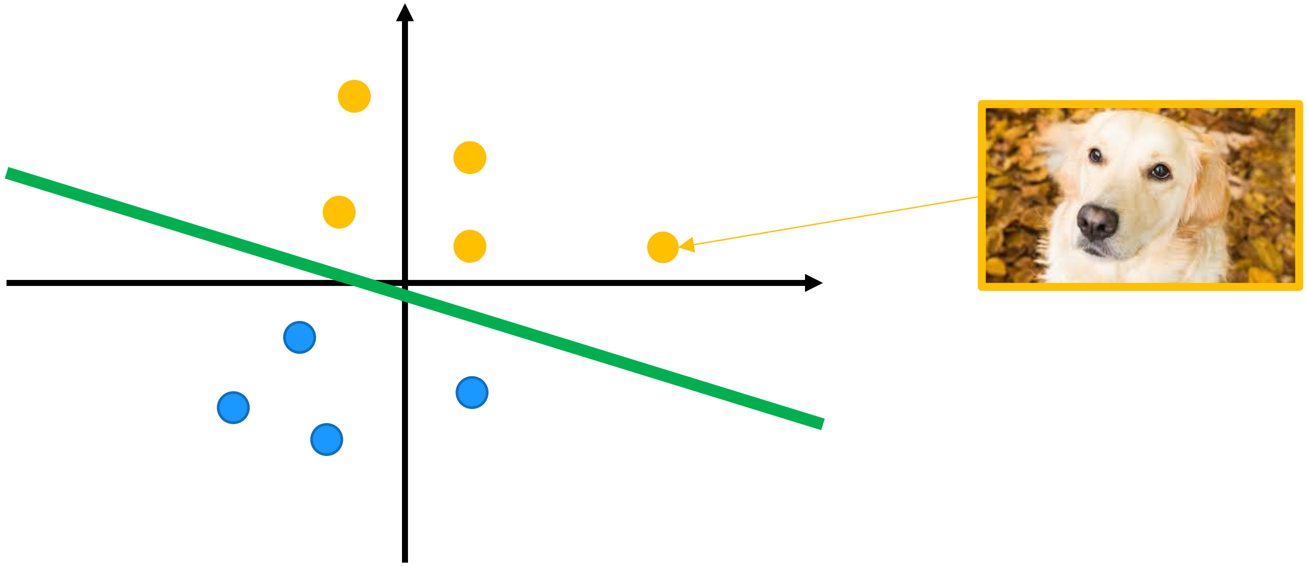


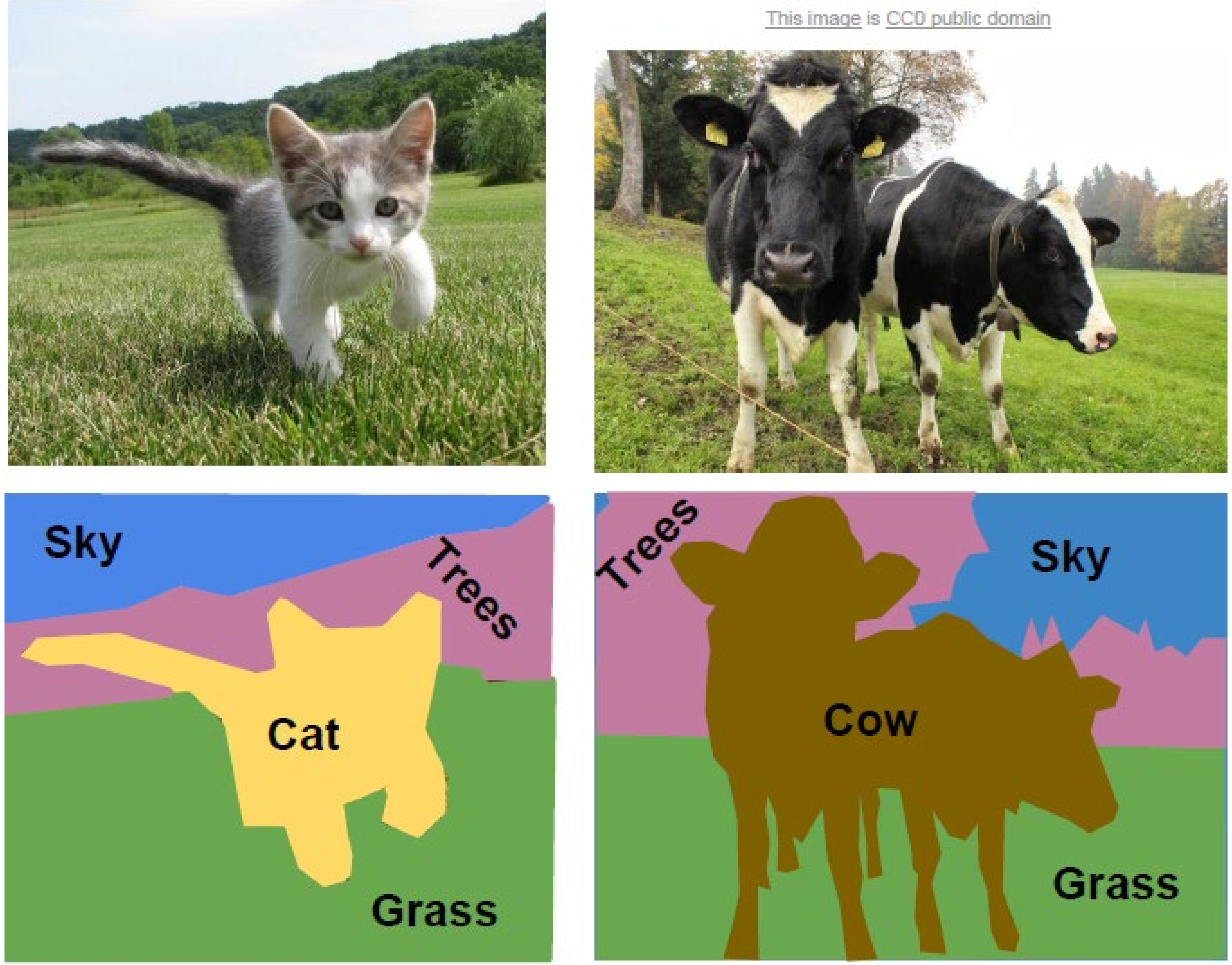
Image Classification



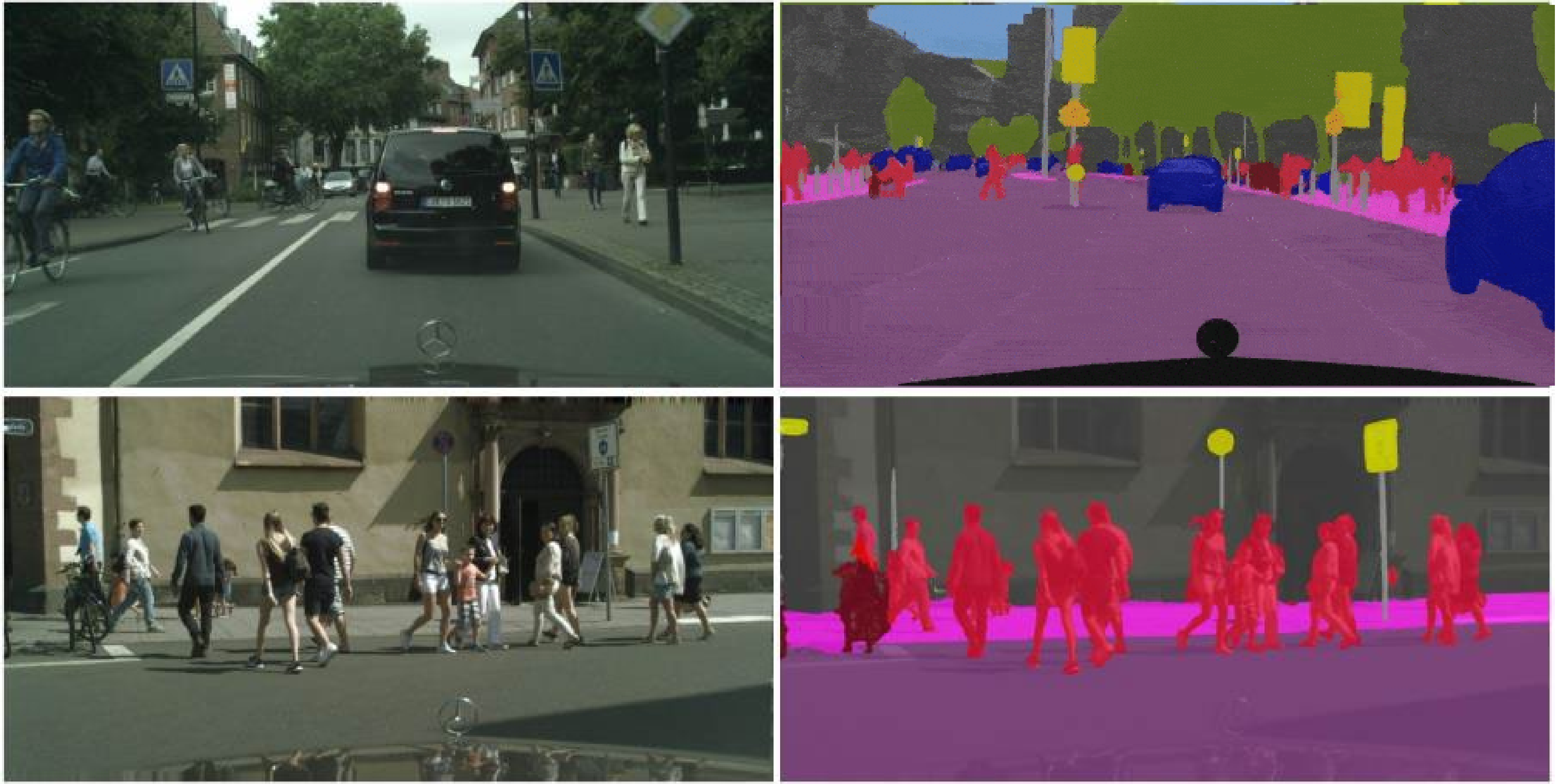


SEMANTIC SEGMENTATION

Semantic Segmentation

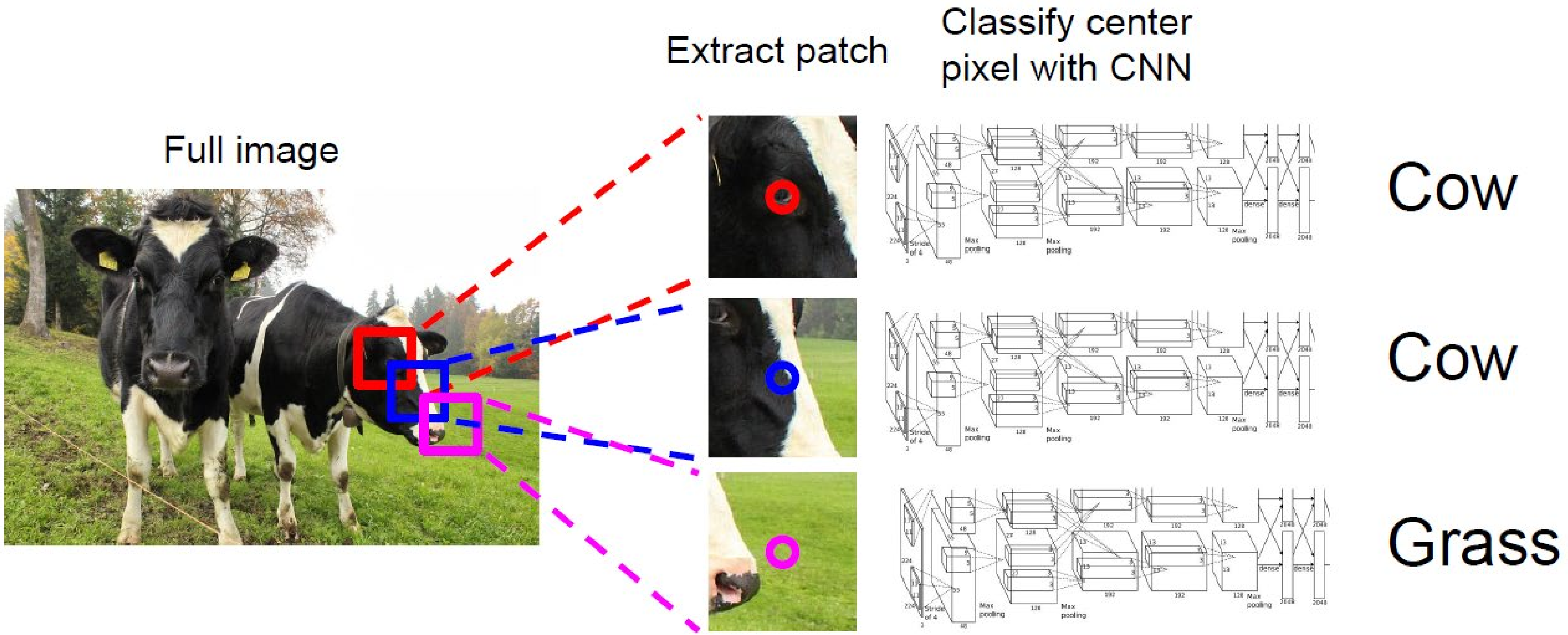


Semantic Segmentation



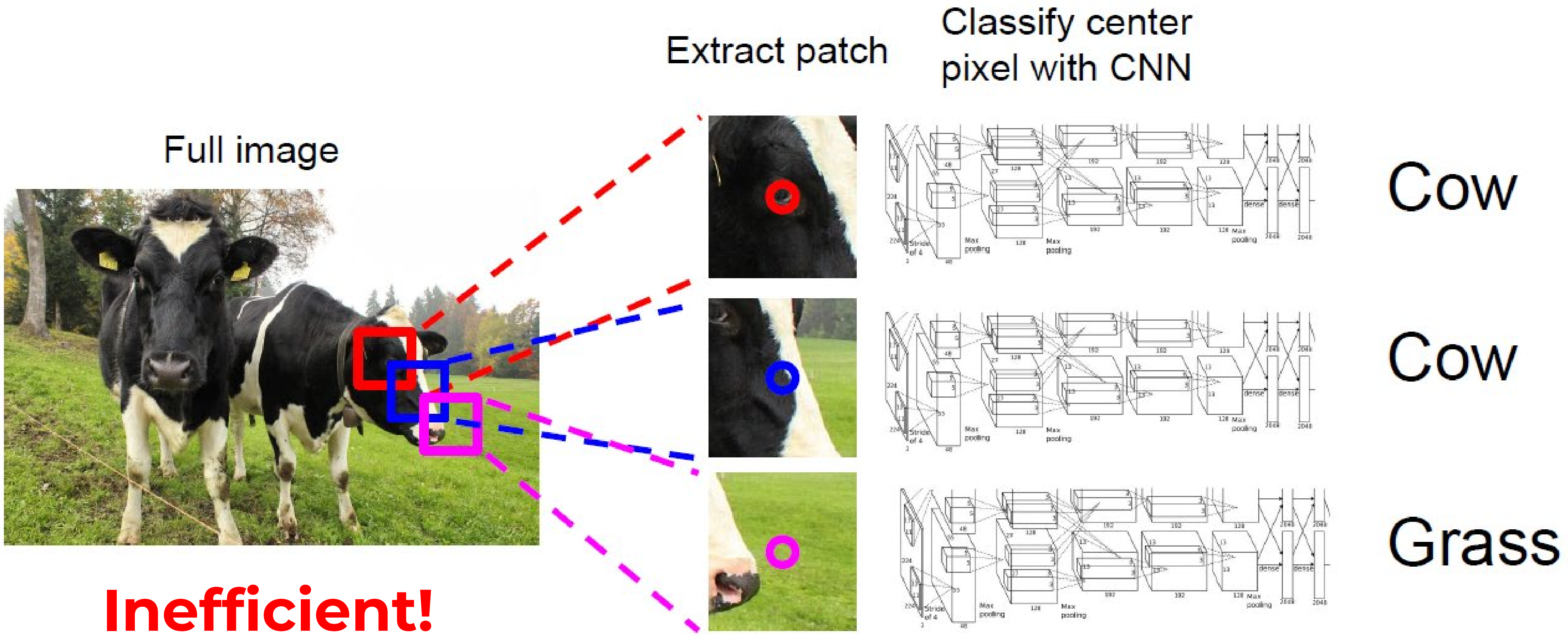
A naïve approach

Semantic segmentation is basically a classification problem



A naïve approach

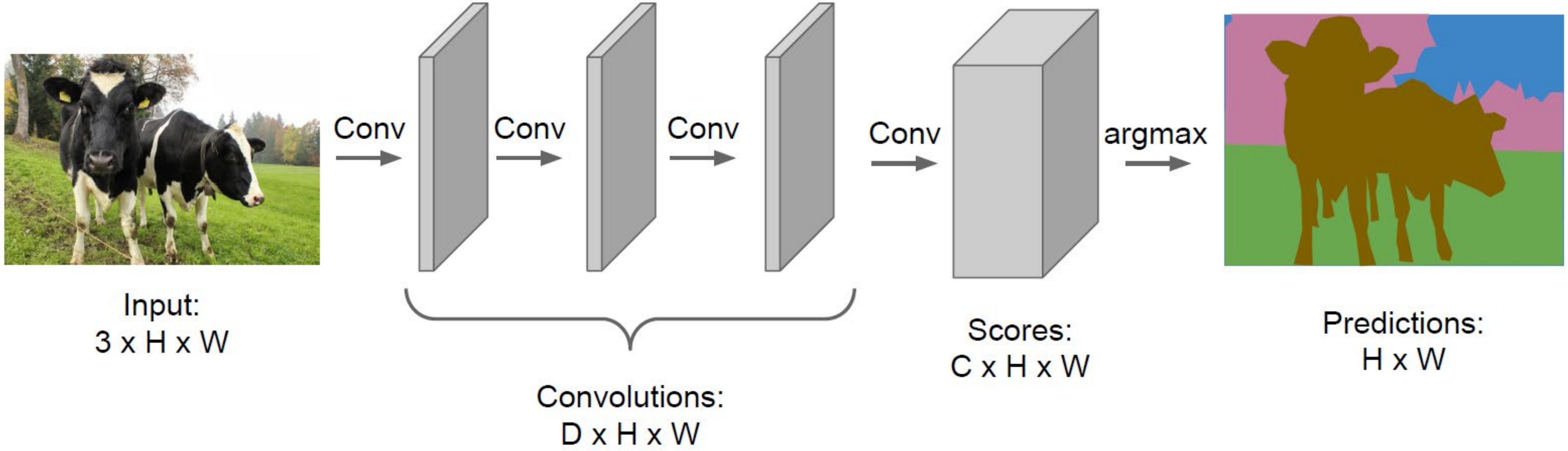
Semantic segmentation is basically a classification problem



Inefficient!

Fully convolutional networks (FCNs)

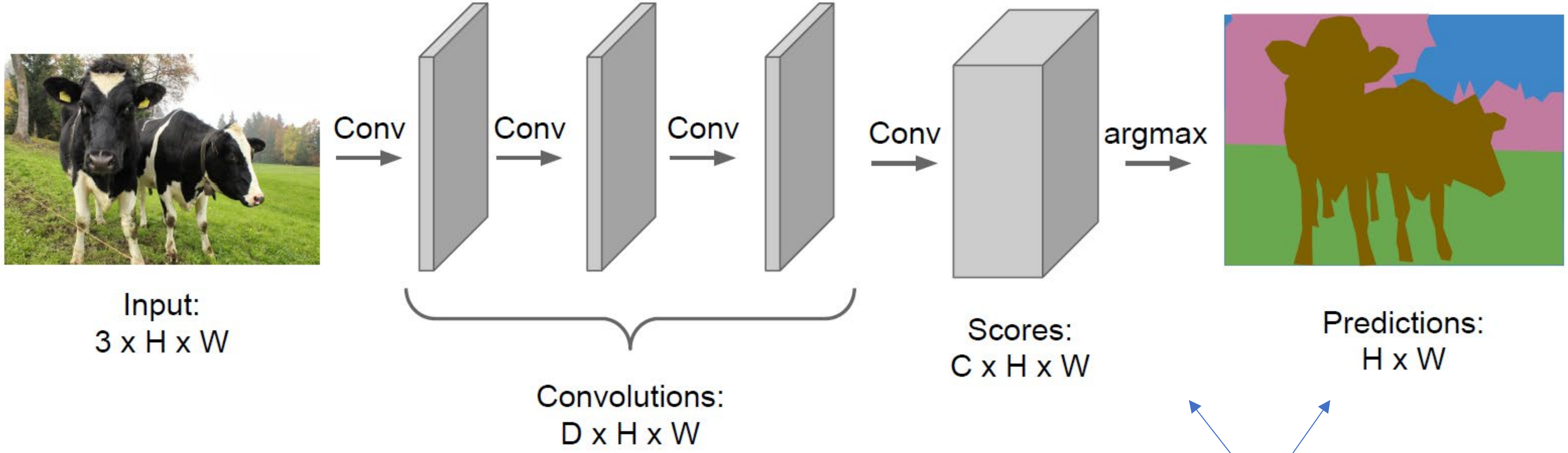
A stack of conv layers to make predictions all at once



Fully convolutional networks (FCNs)

A stack of conv layers to make predictions all at once

Pixel-wise softmax activations

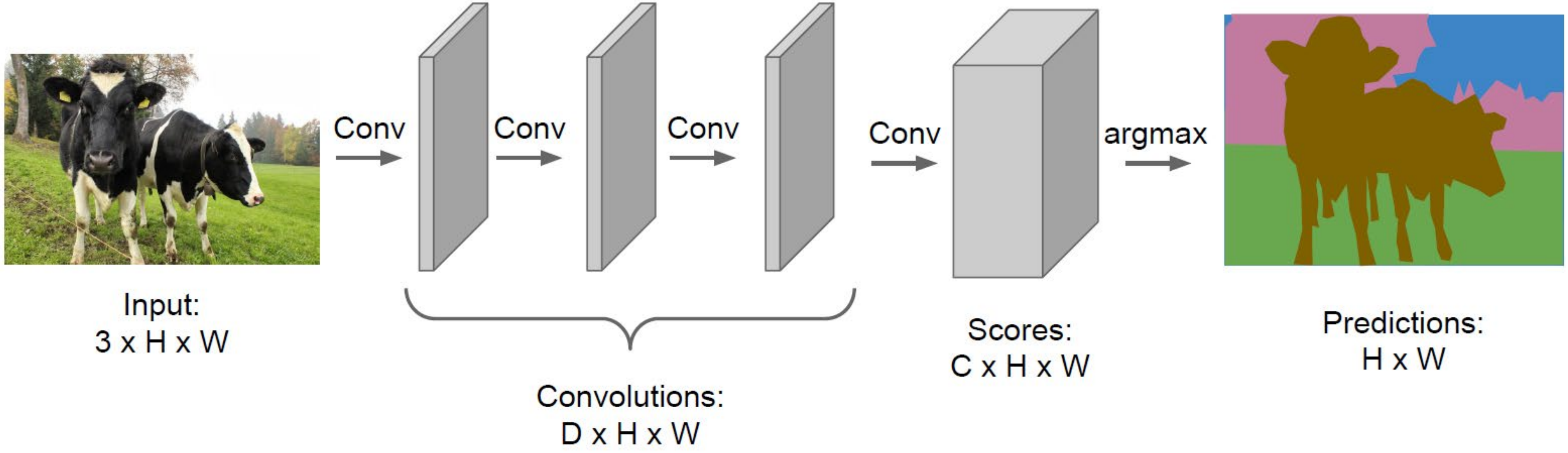


Feature extractor

Final output retains original image dimensions

Fully convolutional networks (FCNs)

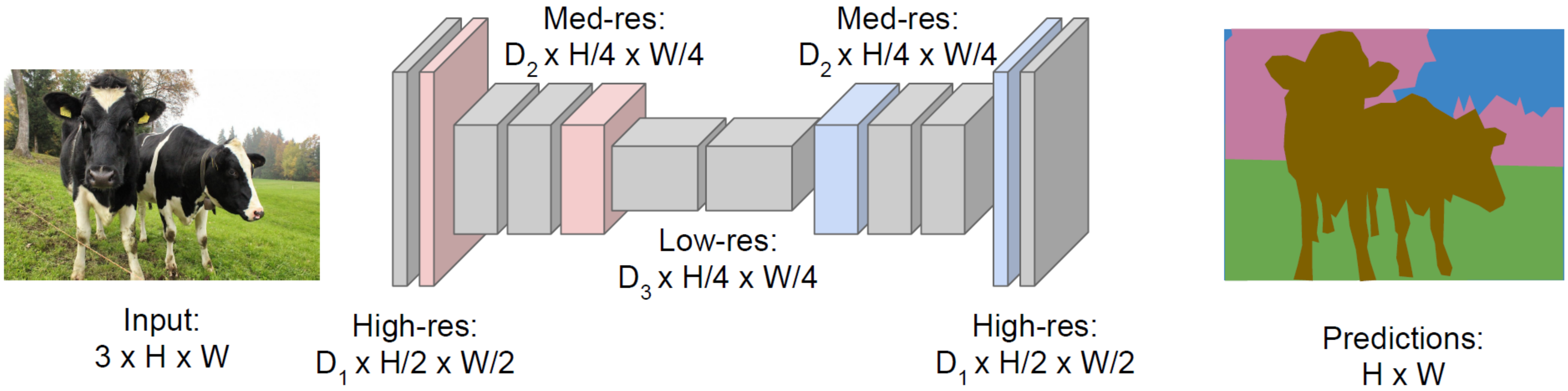
A stack of conv layers to make predictions all at once



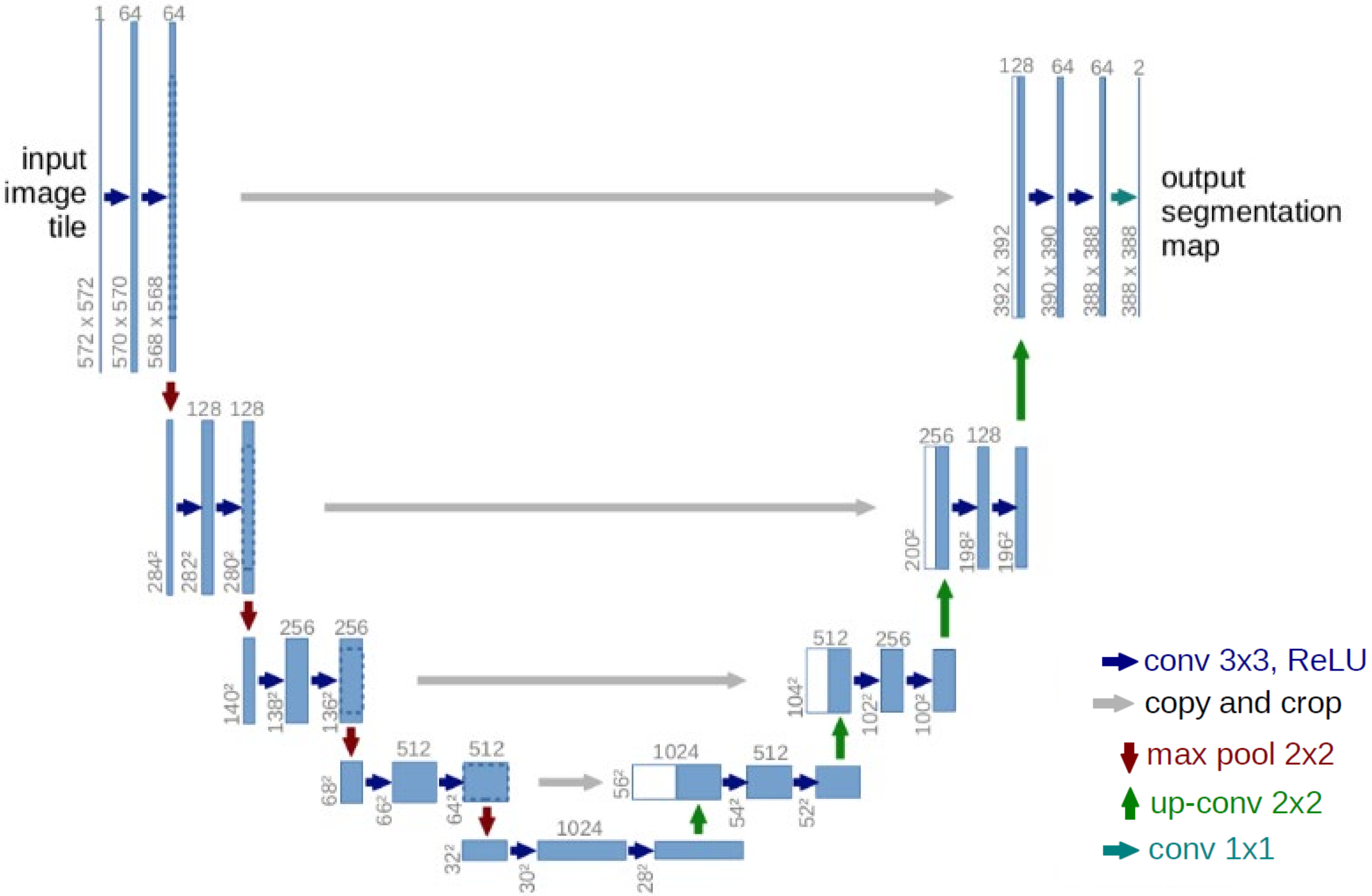
Convolutions at original image resolutions could be very expensive

Fully convolutional networks (FCNs)

A stack of conv layers, with **downsampling** and **upsampling**, to make predictions all at once

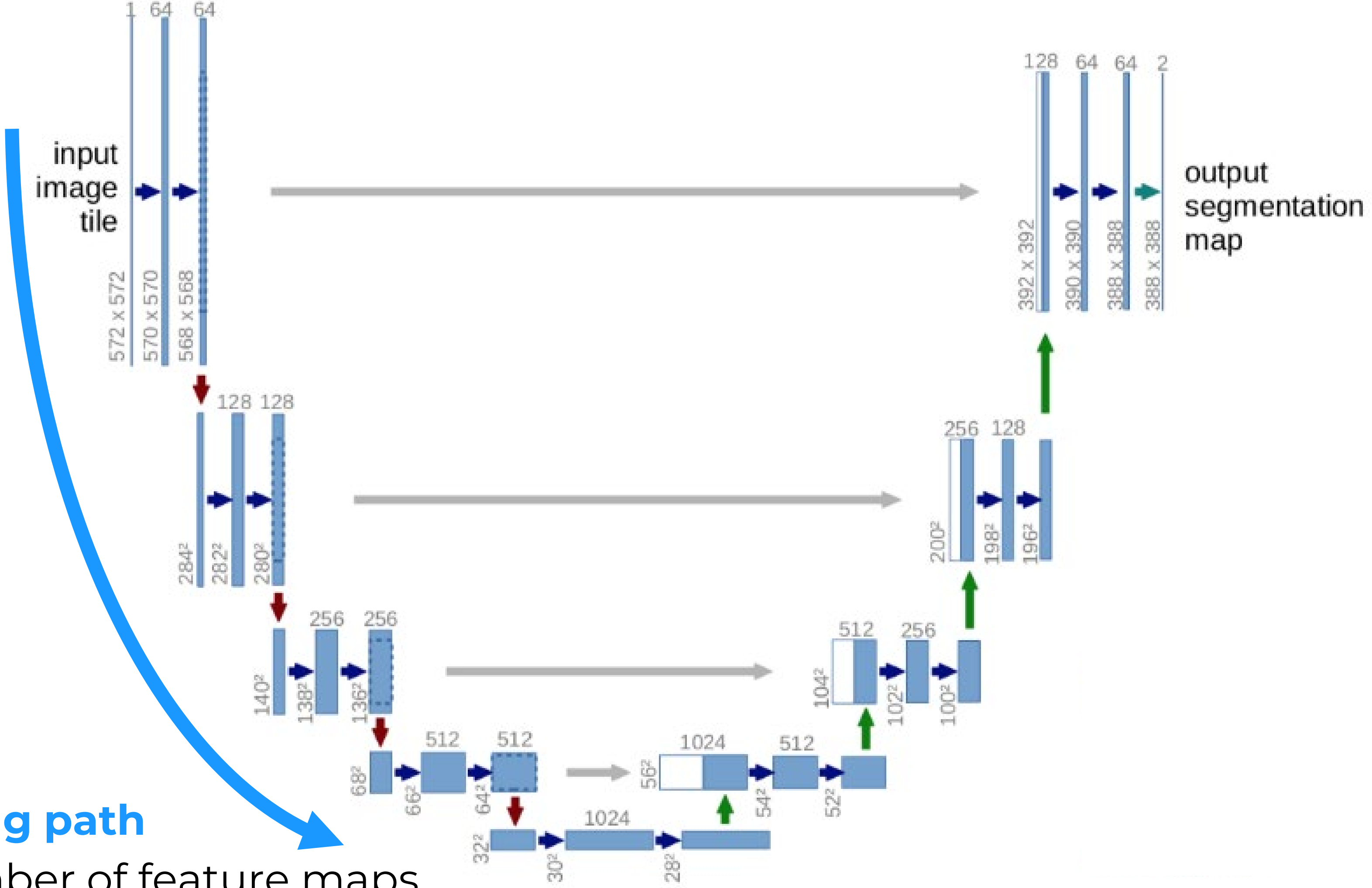


U-Net



Ronneberger et al. "U-net: Convolutional networks for biomedical image segmentation." MICCAI2015. [Paper]

U-Net



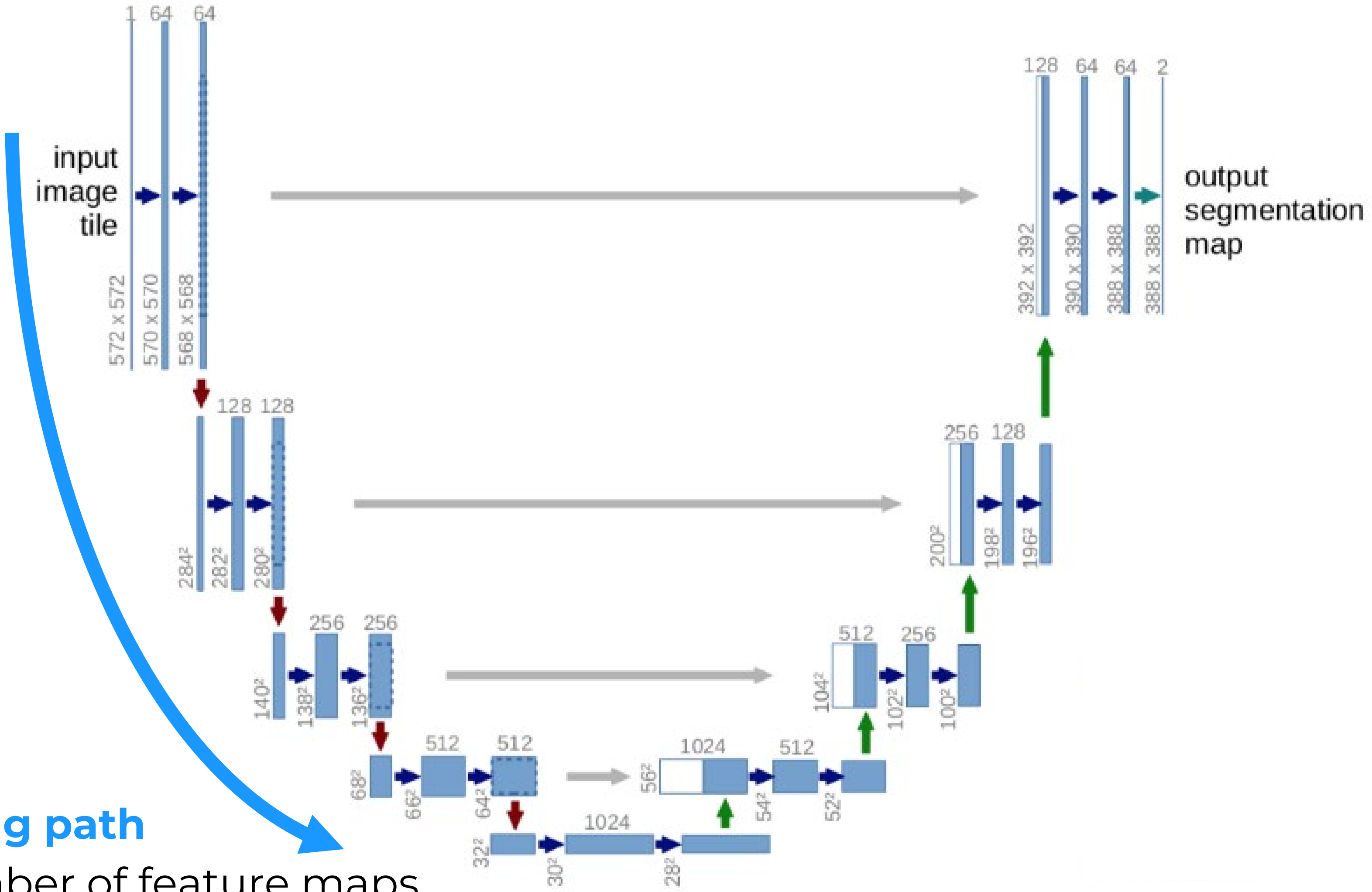
Downsampling path

Increases number of feature maps

Reduces spatial size

«Compresses» information

U-Net



Downsampling path

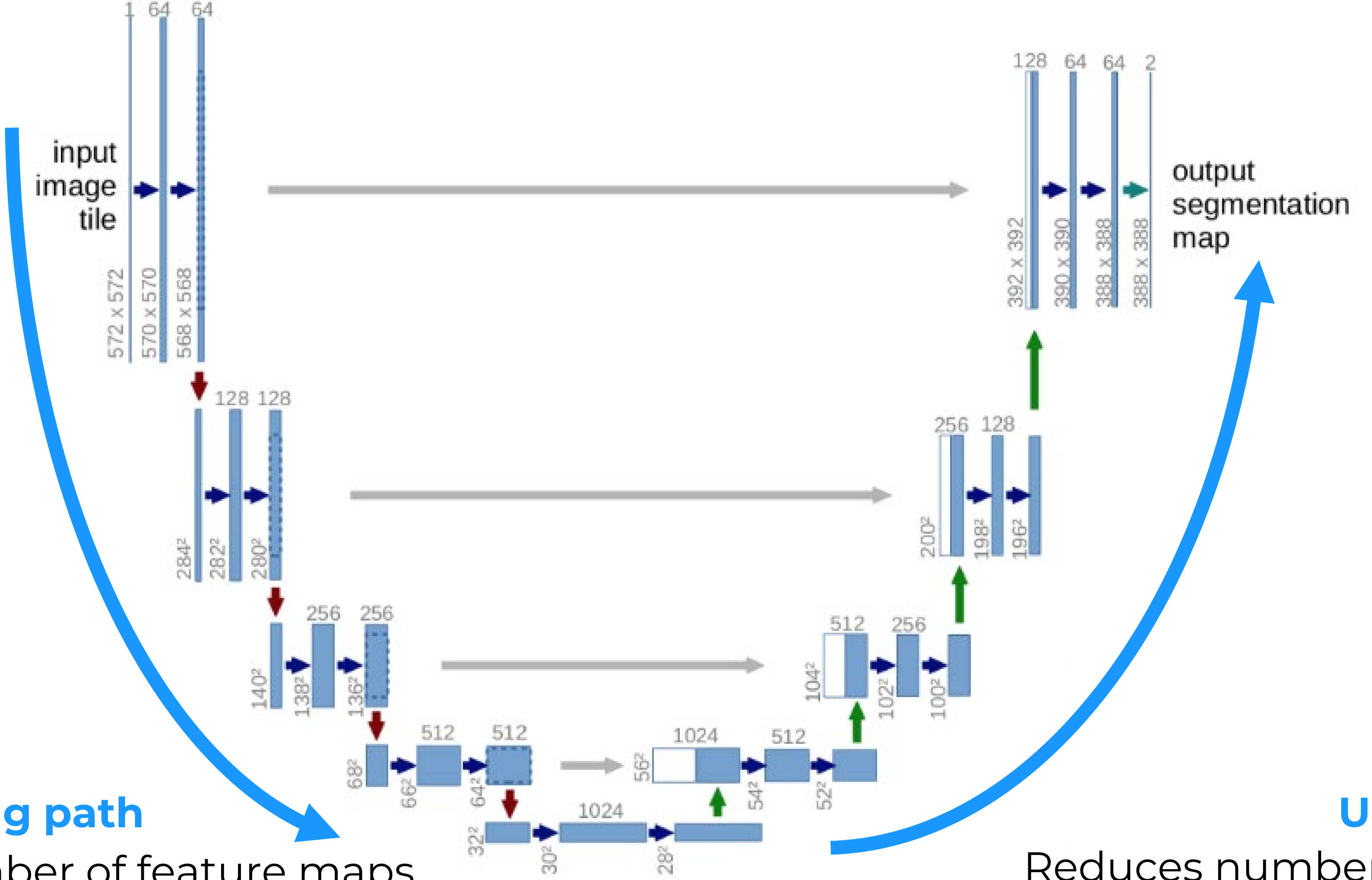
Increases number of feature maps

Reduces spatial size

«Compresses» information

Bottleneck

U-Net



Downsampling path

Increases number of feature maps

Reduces spatial size

«Compresses» information

Bottleneck

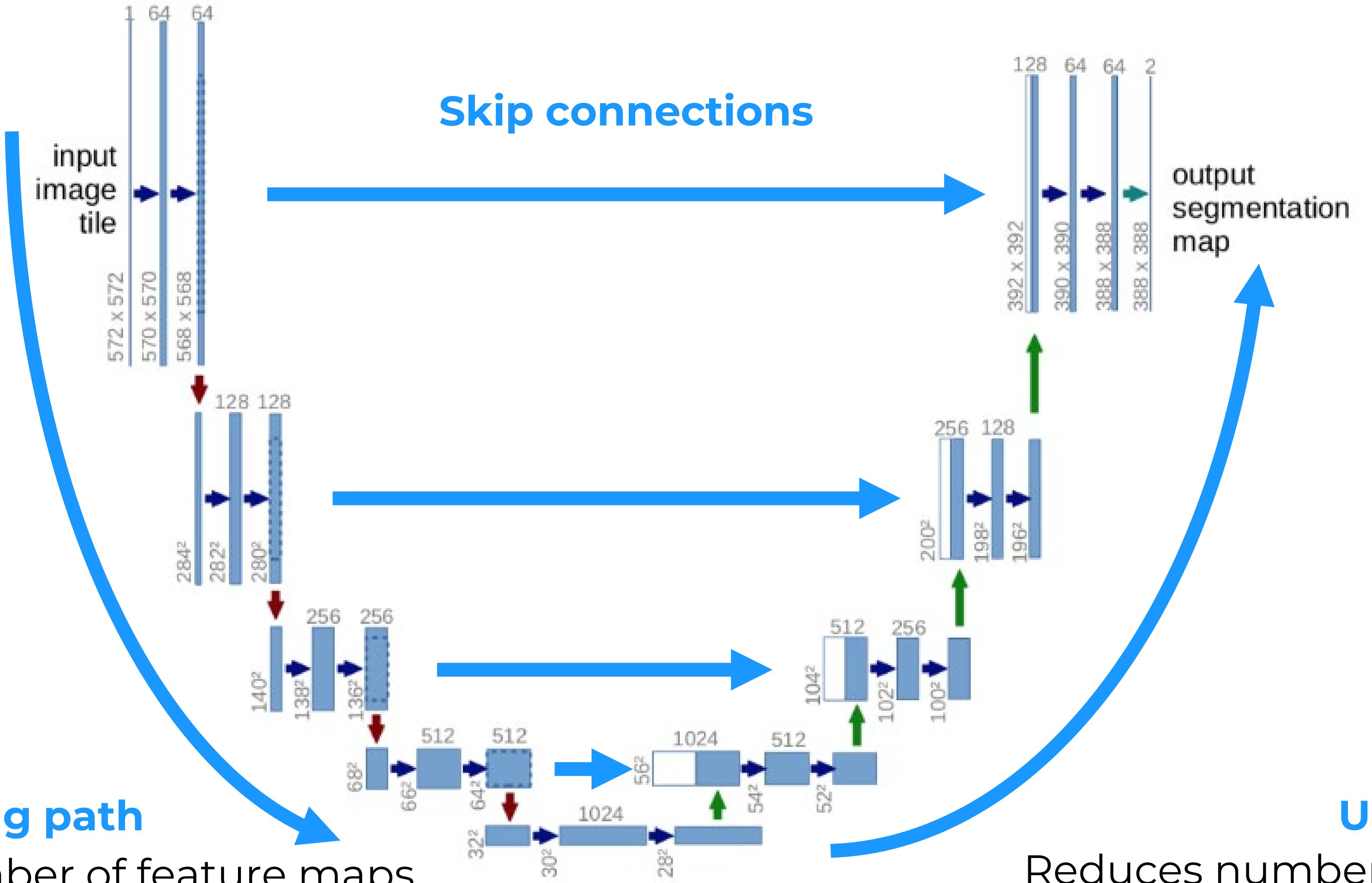
Upsampling path

Reduces number of feature maps

Increase spatial size

«Expands» information

U-Net



Downsampling path

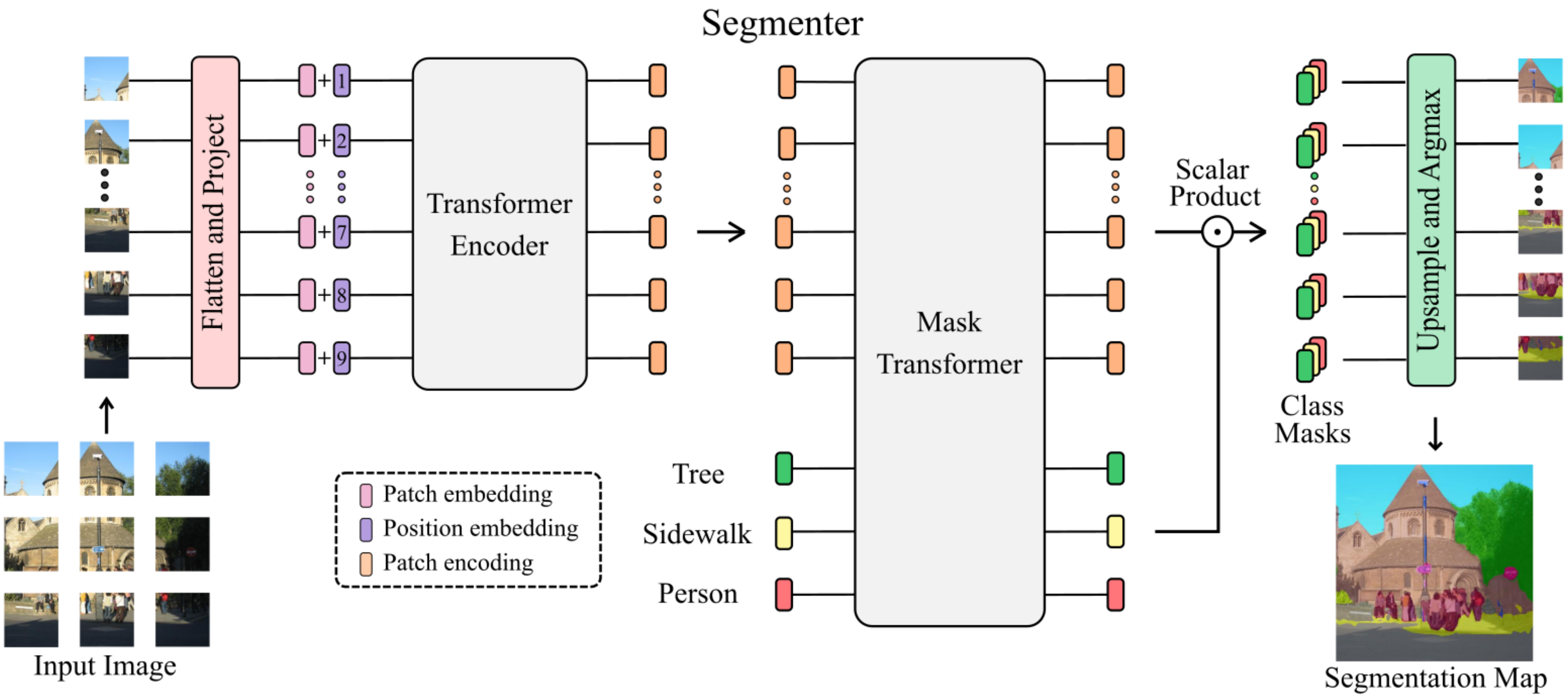
Increases number of feature maps
Reduces spatial size
«Compresses» information

Bottleneck

Upsampling path

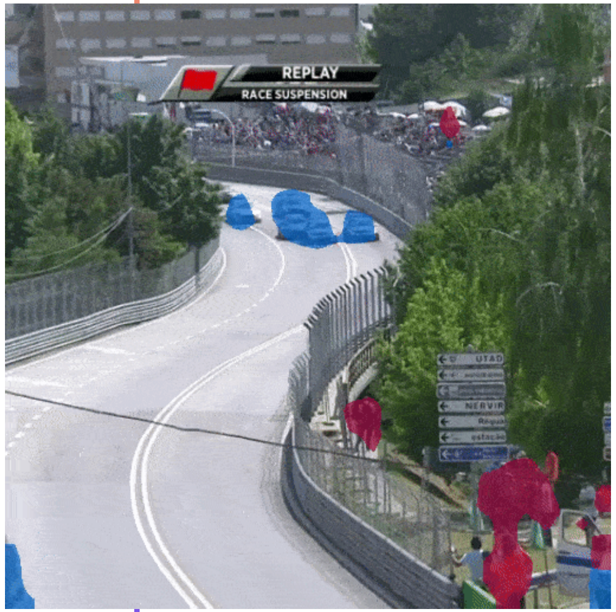
Reduces number of feature maps
Increase spatial size
«Expands» information

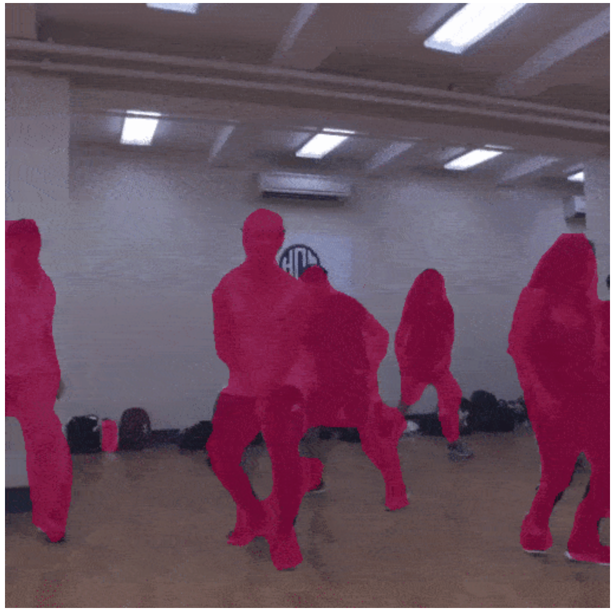
Segmenter



Strudel et al. "Segmenter: Transformer for semantic segmentation." ICCV2021. [Paper][GitHub]

Segmenter



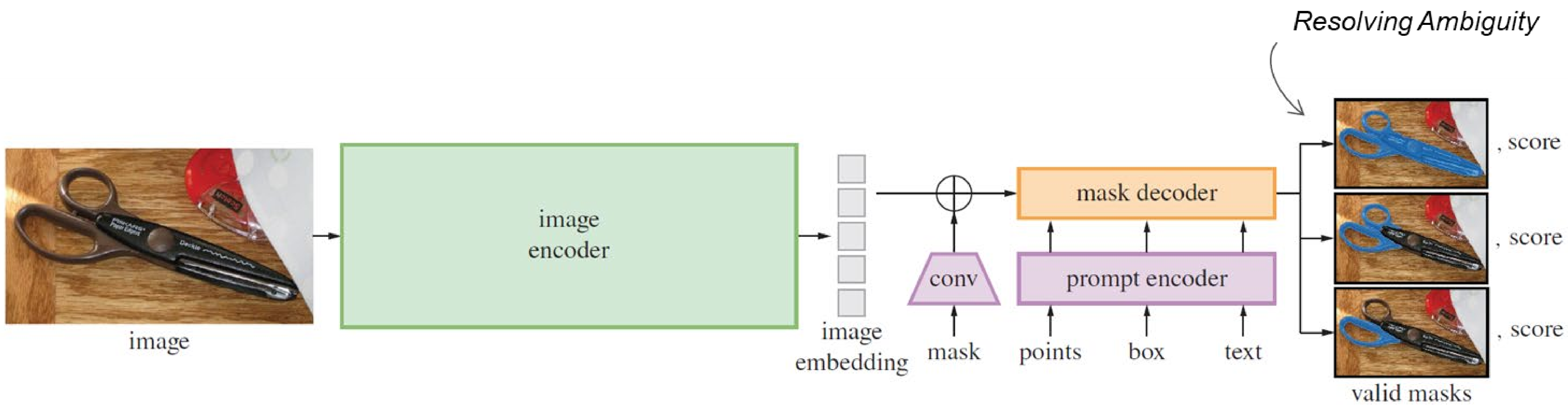


Strudel et al. "Segmenter: Transformer for semantic segmentation." ICCV2021. [Paper][GitHub]

Segment Anything Model (SAM)

Kirillov et al. “Segment anything”. ICCV2023. [Paper][GitHub] Image by Meta AI

SAM: overview

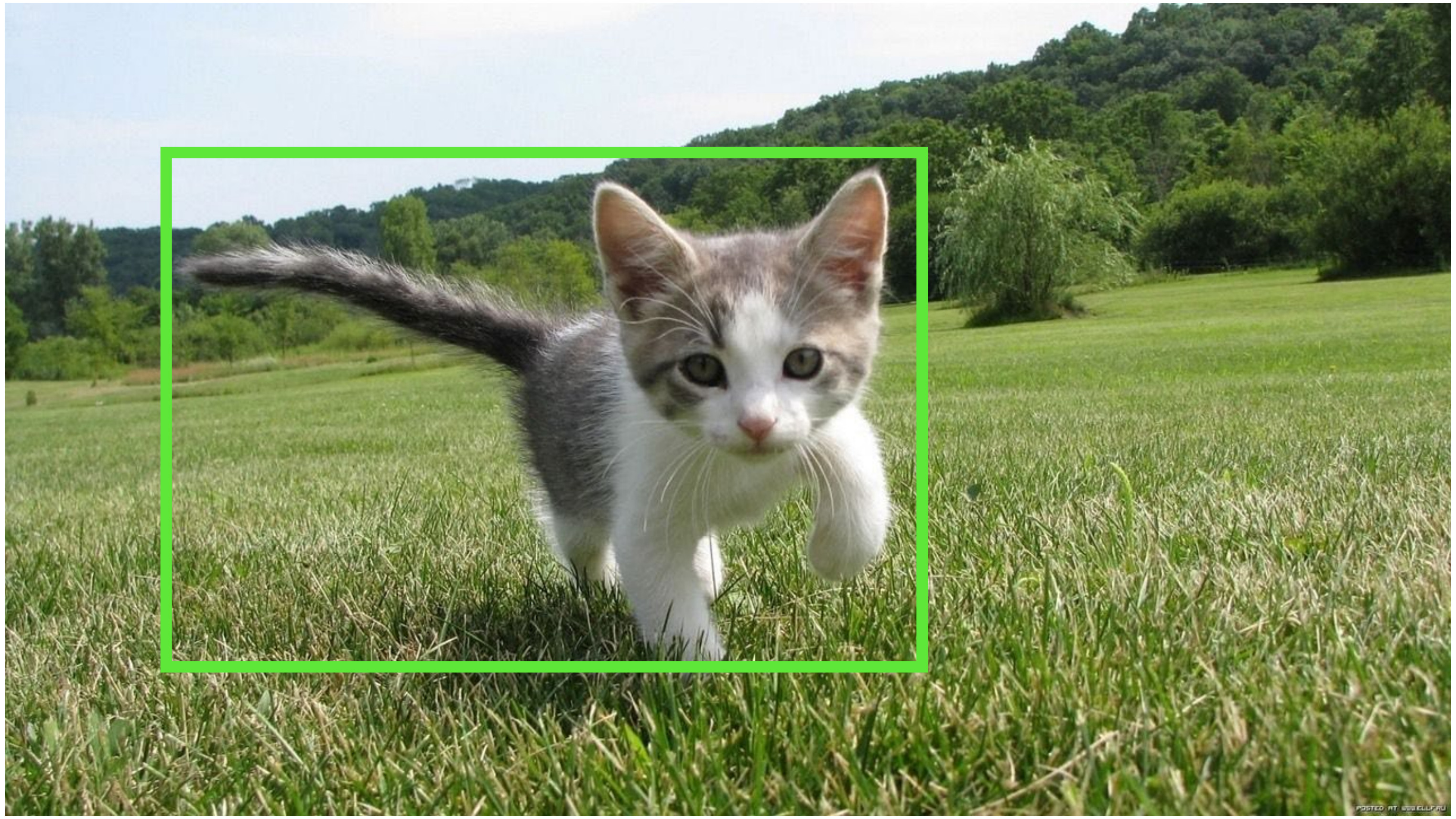


Kirillov et al. “Segment anything”. ICCV2023. [Paper][GitHub]

Kirillov et al. “Segment anything”. ICCV2023. [Paper][GitHub]

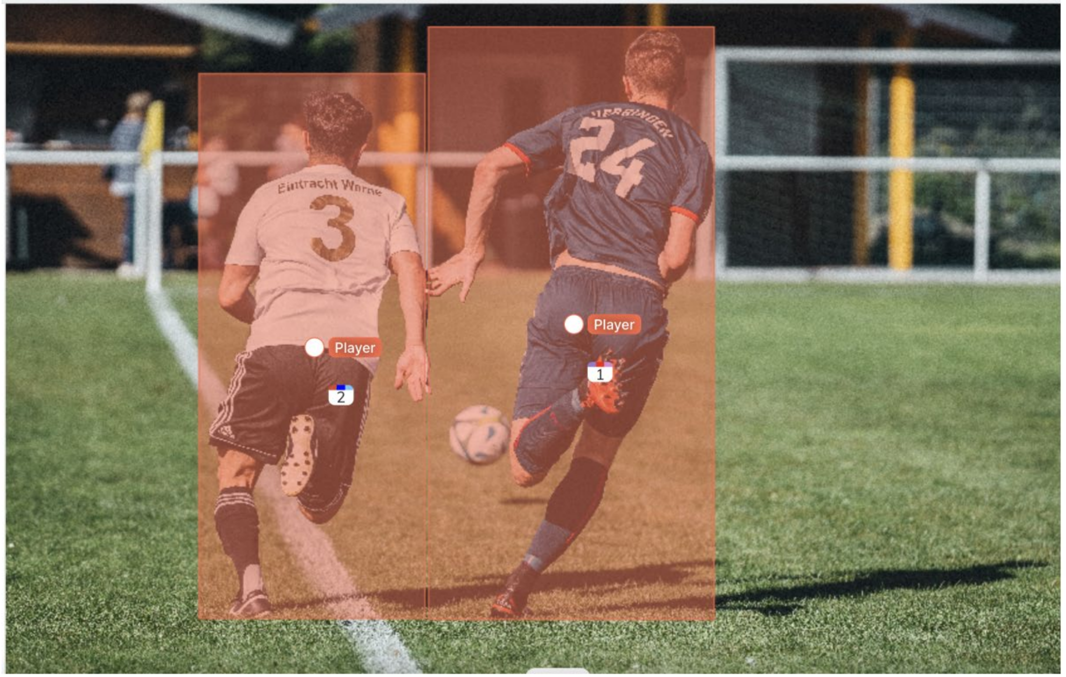
OBJECT DETECTION

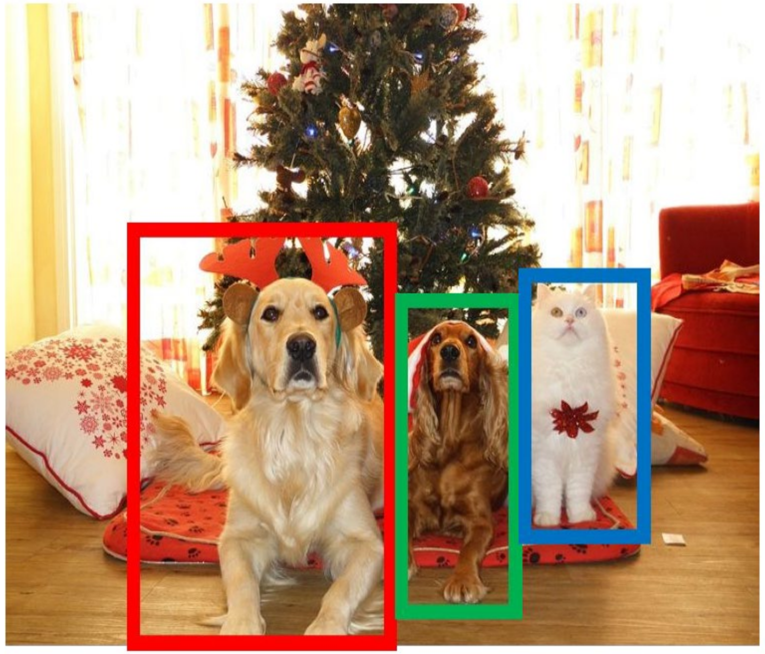
Localization



Predict coordinates of a bounding box (x, y, w, h) that *contains* an entity.

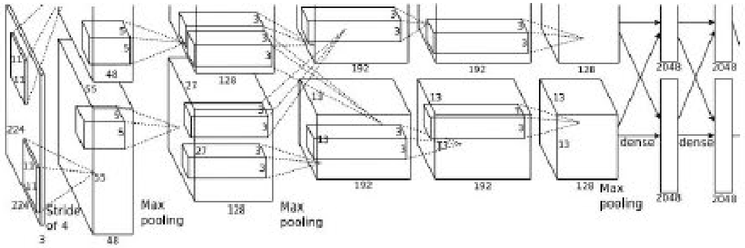
Object detection



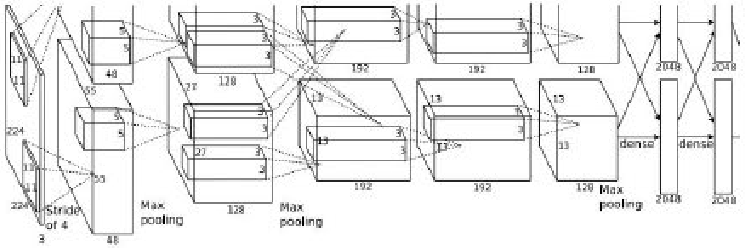


DOG, DOG, CAT

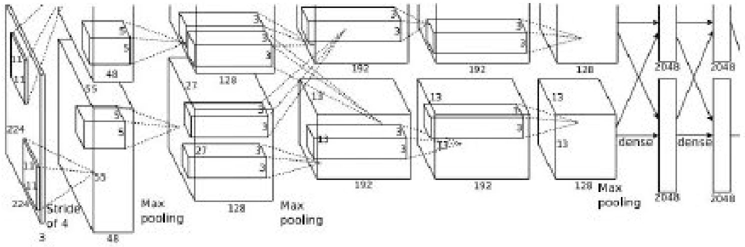
Object detection as a Regression problem

CAT: (x, y, w, h)

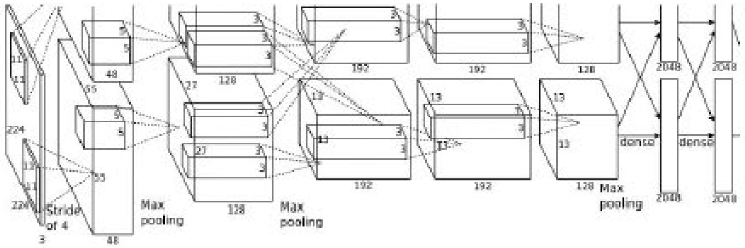



DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

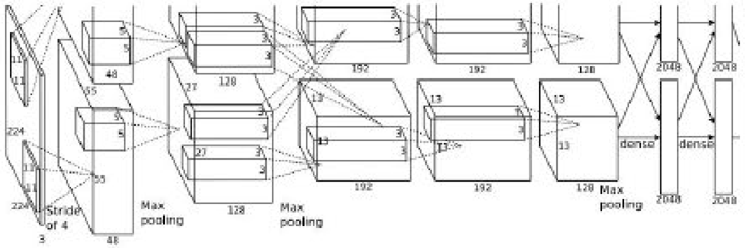
DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

Object detection as a Regression problem



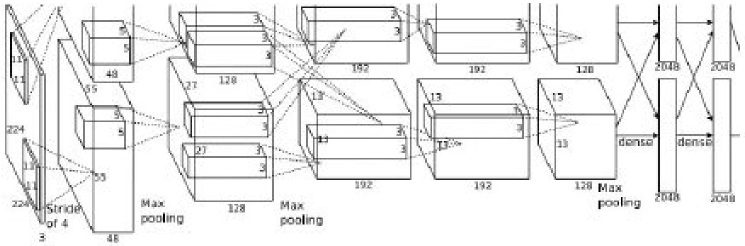
CAT: (x, y, w, h)

4 numbers



DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

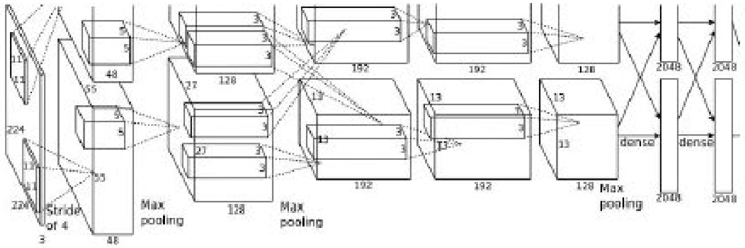
12 numbers



DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

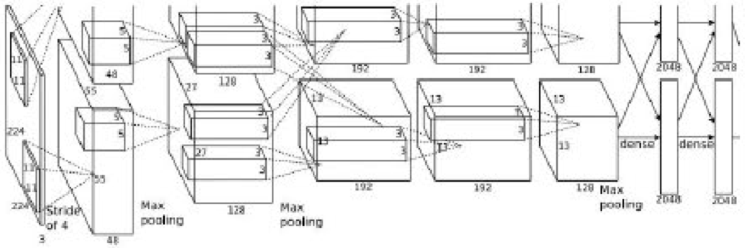
***many* numbers**

Object detection as a Regression problem



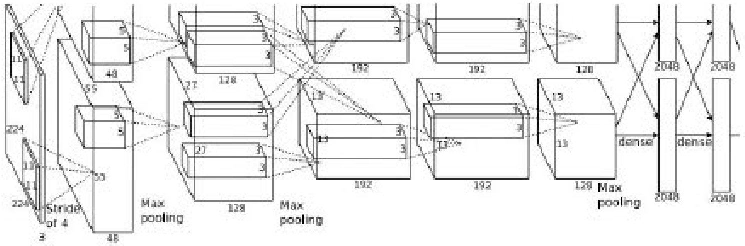
CAT: (x, y, w, h)

4 numbers



DOG: (x, y, w, h)
DOG: (x, y, w, h)
CAT: (x, y, w, h)

12 numbers

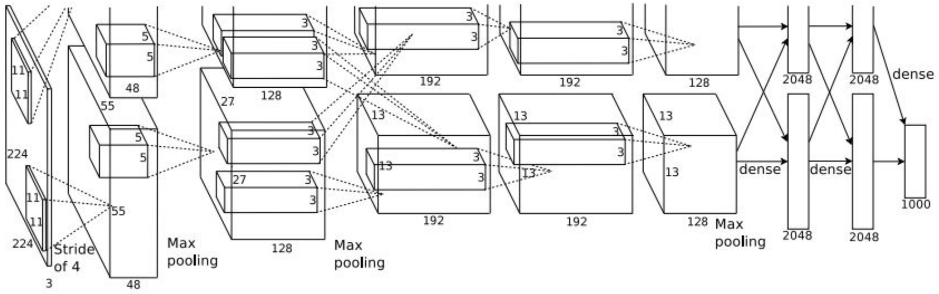


DUCK: (x, y, w, h)
DUCK: (x, y, w, h)
....

***many* numbers**

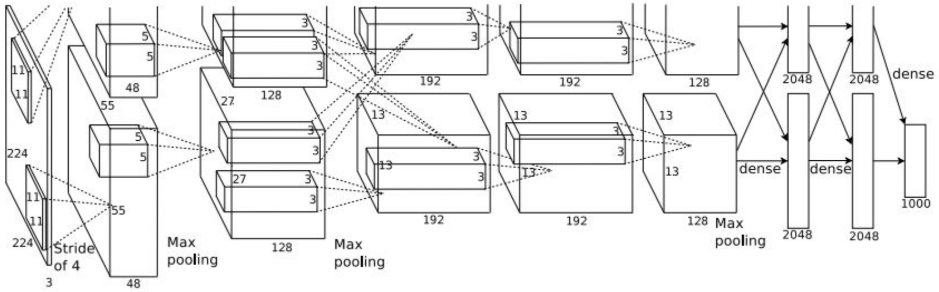
Each image can contain different number of entities

Object detection as Classification



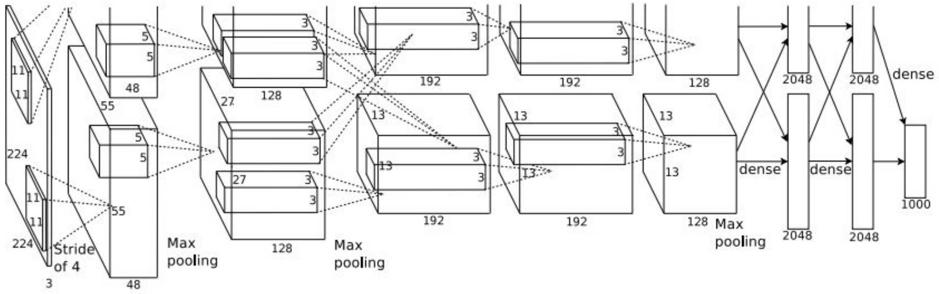
DOG? NO
CAT? NO
Background? YES

Object detection as Classification



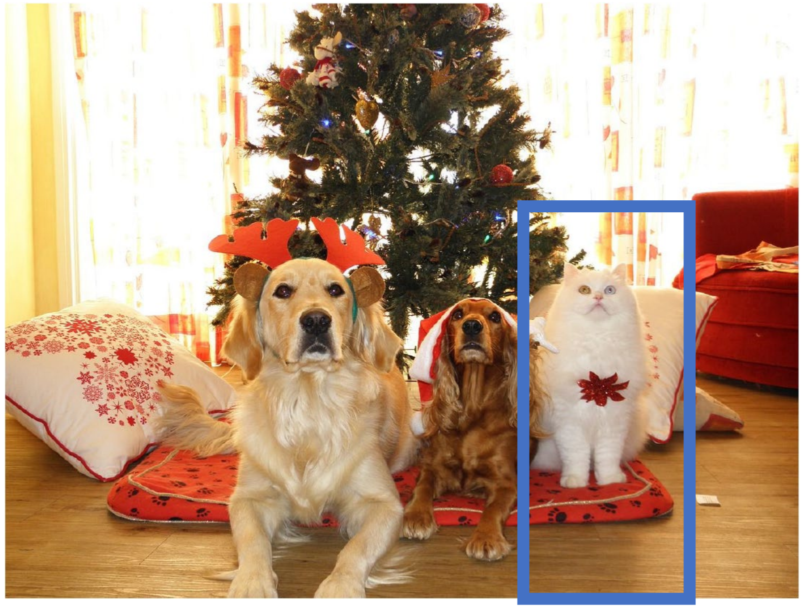
DOG? YES
CAT? NO
Background? NO

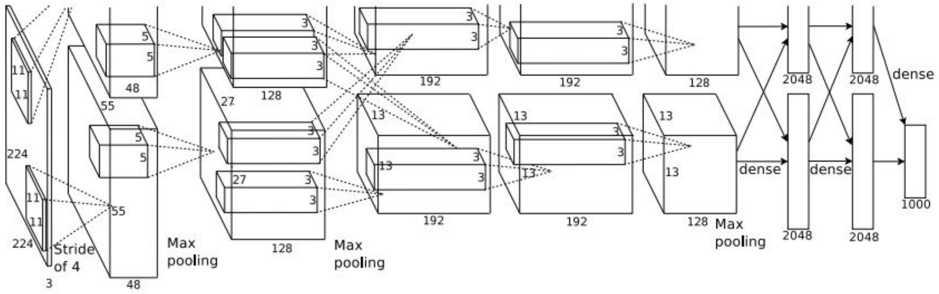
Object detection as Classification



DOG? YES
CAT? NO
Background? NO

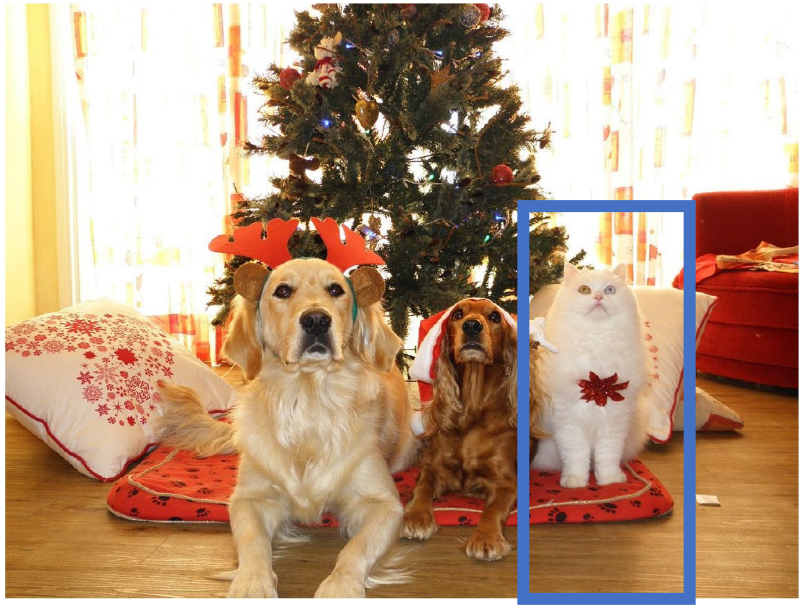
Object detection as Classification

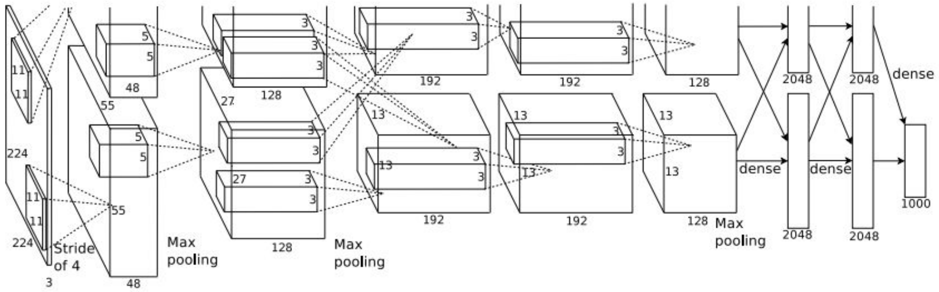




DOG? NO
CAT? YES
Background? NO

Object detection as Classification

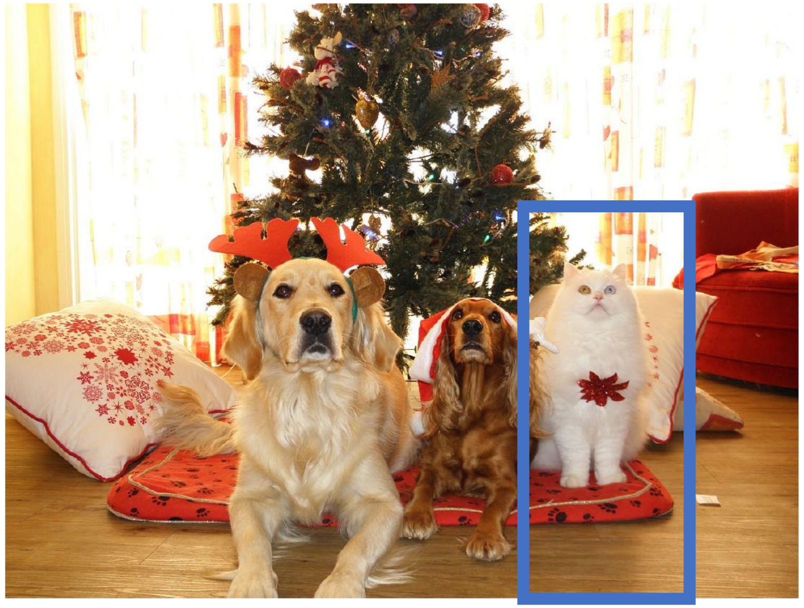


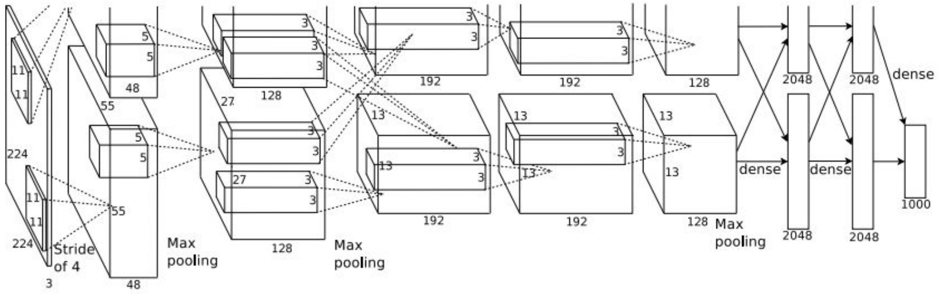


DOG? NO
CAT? YES
Background? NO

SLIDING WINDOW

Object detection as Classification



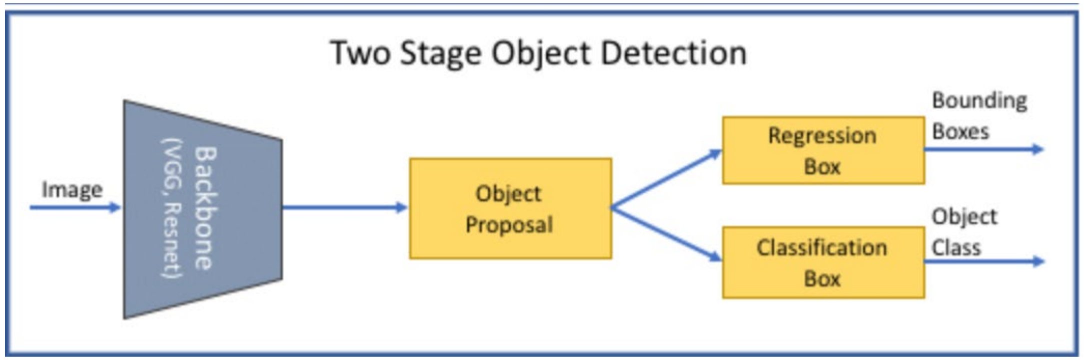


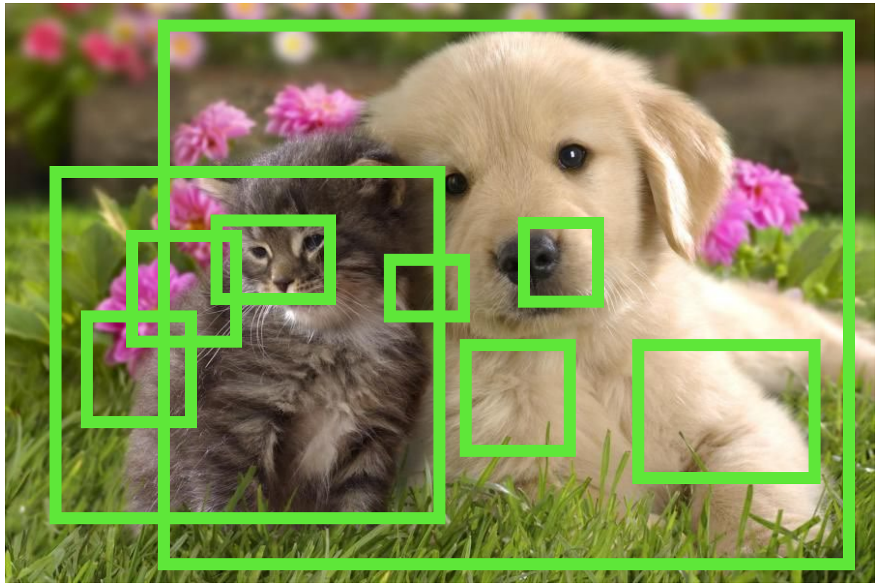
DOG? NO
CAT? YES
Background? NO

SLIDING WINDOW

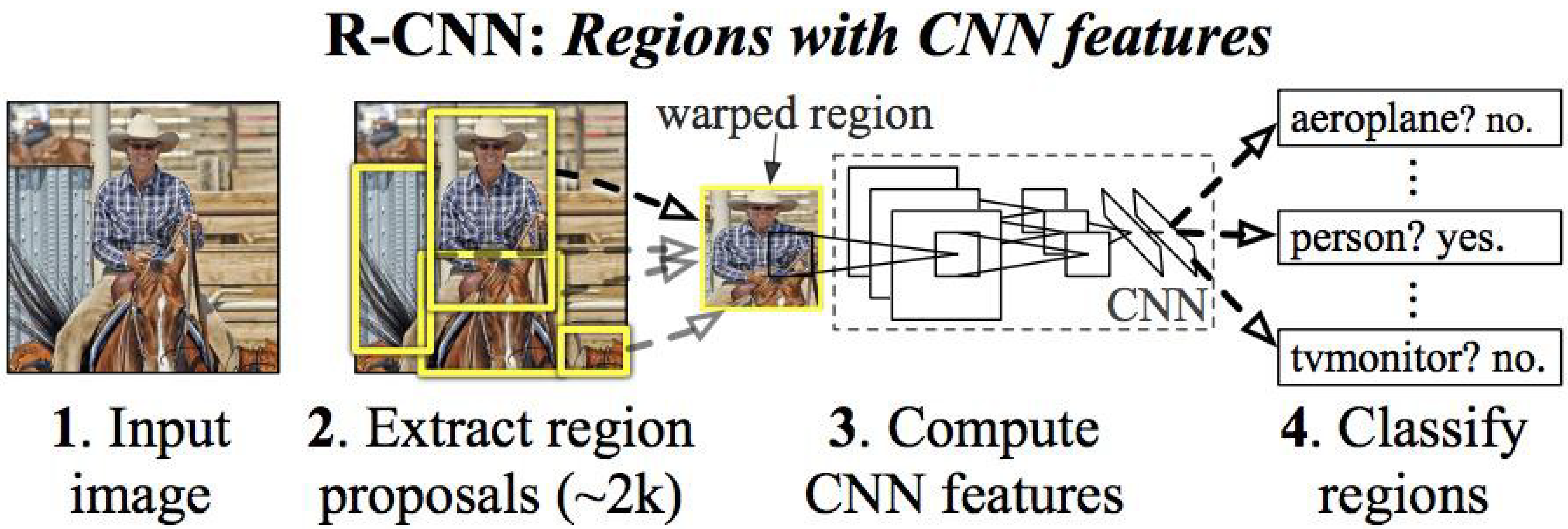
Need to apply CNN to huge number of locations and scales

Region Proposals





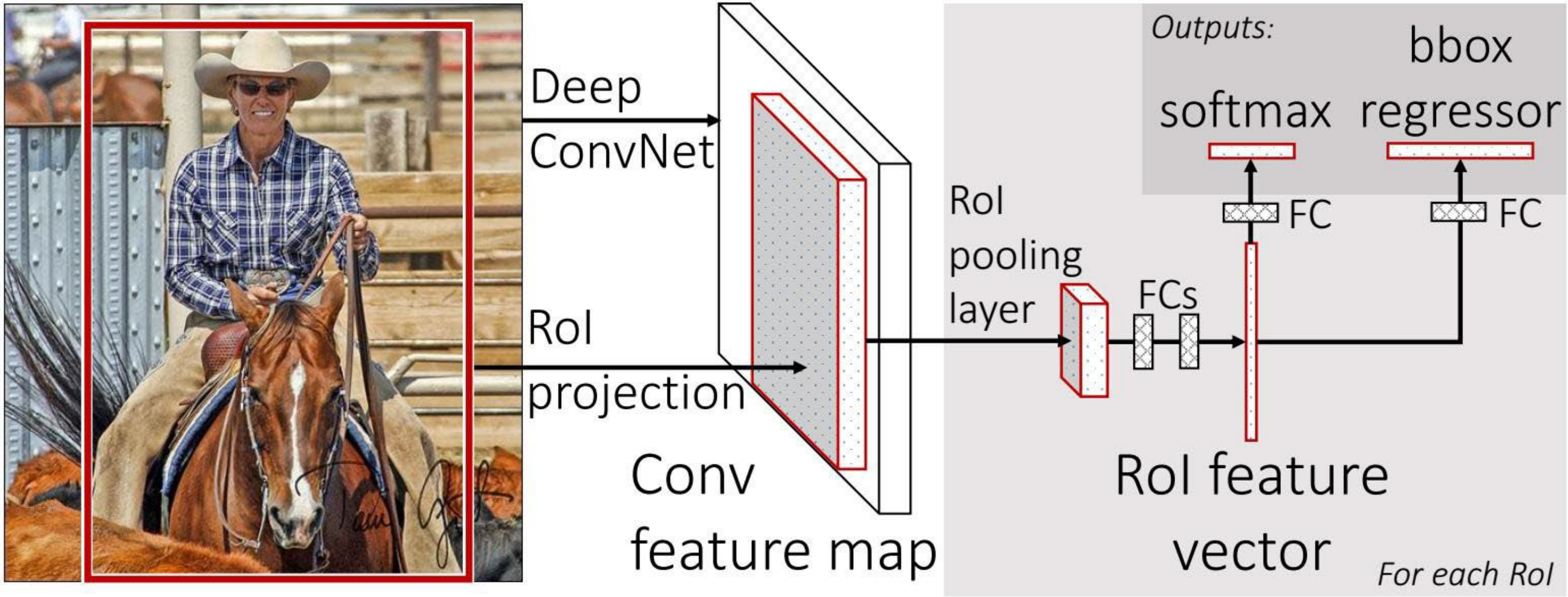
Object detection: The RCNN Family



Girshick et al. "Rich feature hierarchies for accurate object detection and semantic segmentation." CVPR2014. [Paper]

Fast R-CNN

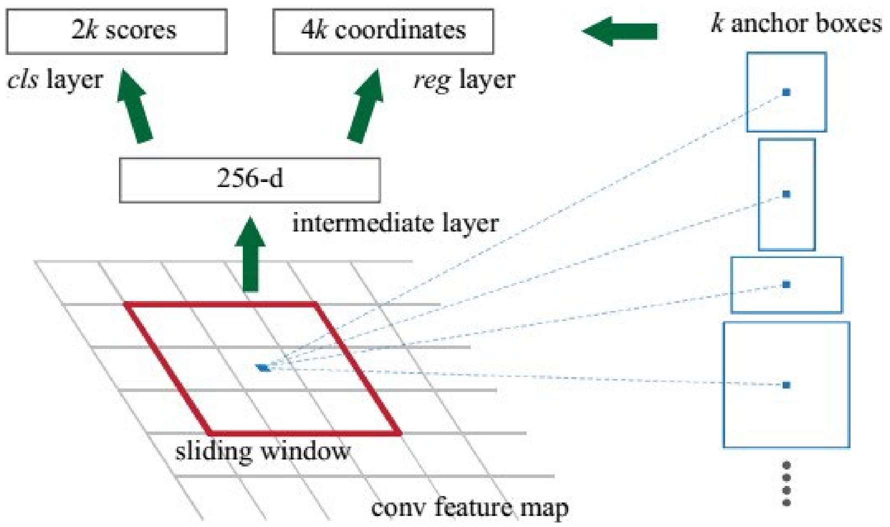
Predictions from sliding windows on feature maps



He et al. "Spatial pyramid pooling in deep convolutional networks for visual recognition." *IEEE TPAMI* 2015. [[Paper](#)]

Faster R-CNN

Generate also candidate locations



Region Proposal Network

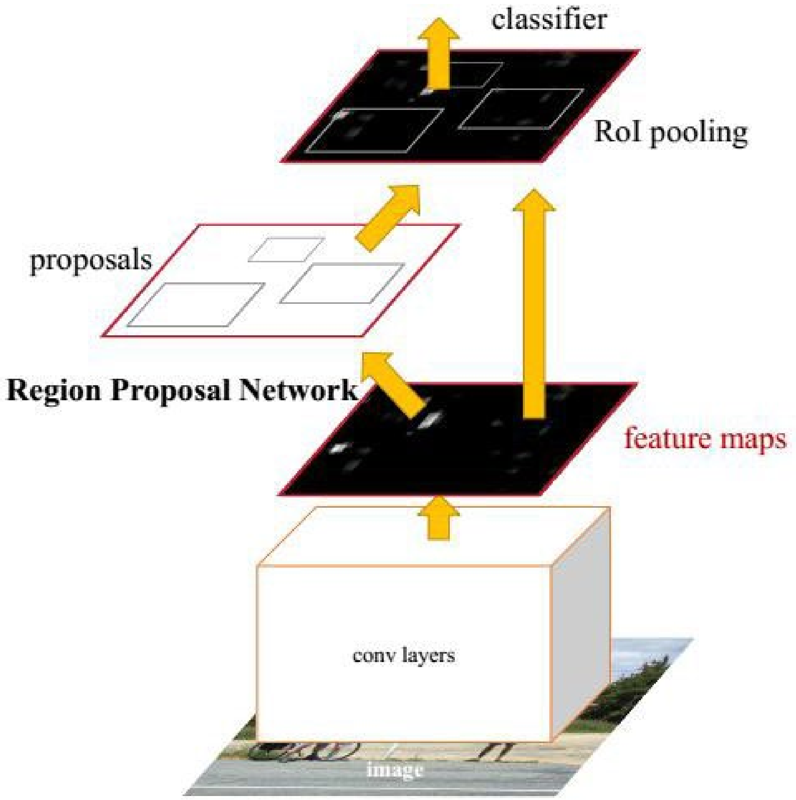
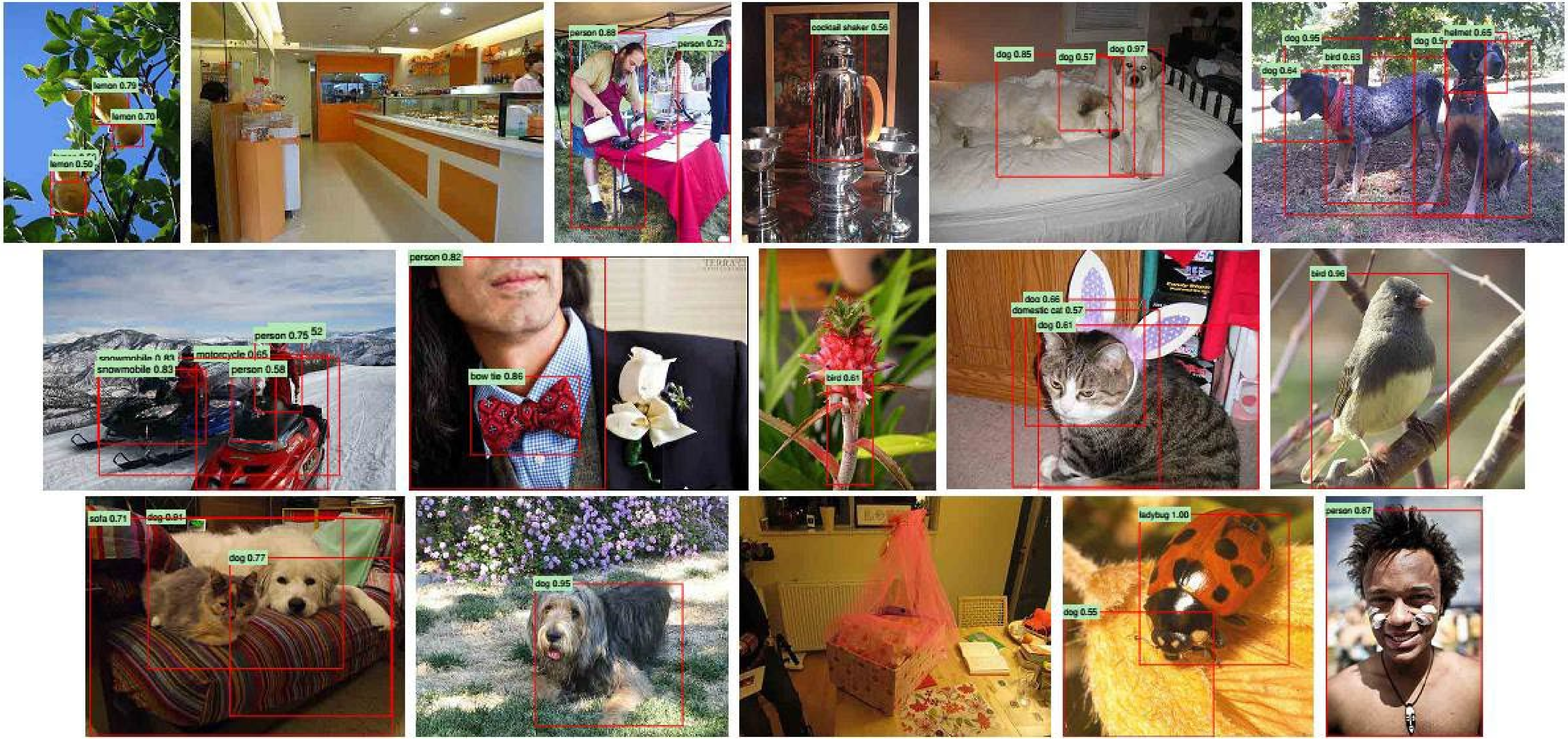


Figure 2: Faster R-CNN is a single, unified network for object detection. The RPN module serves as the 'attention' of this unified network.

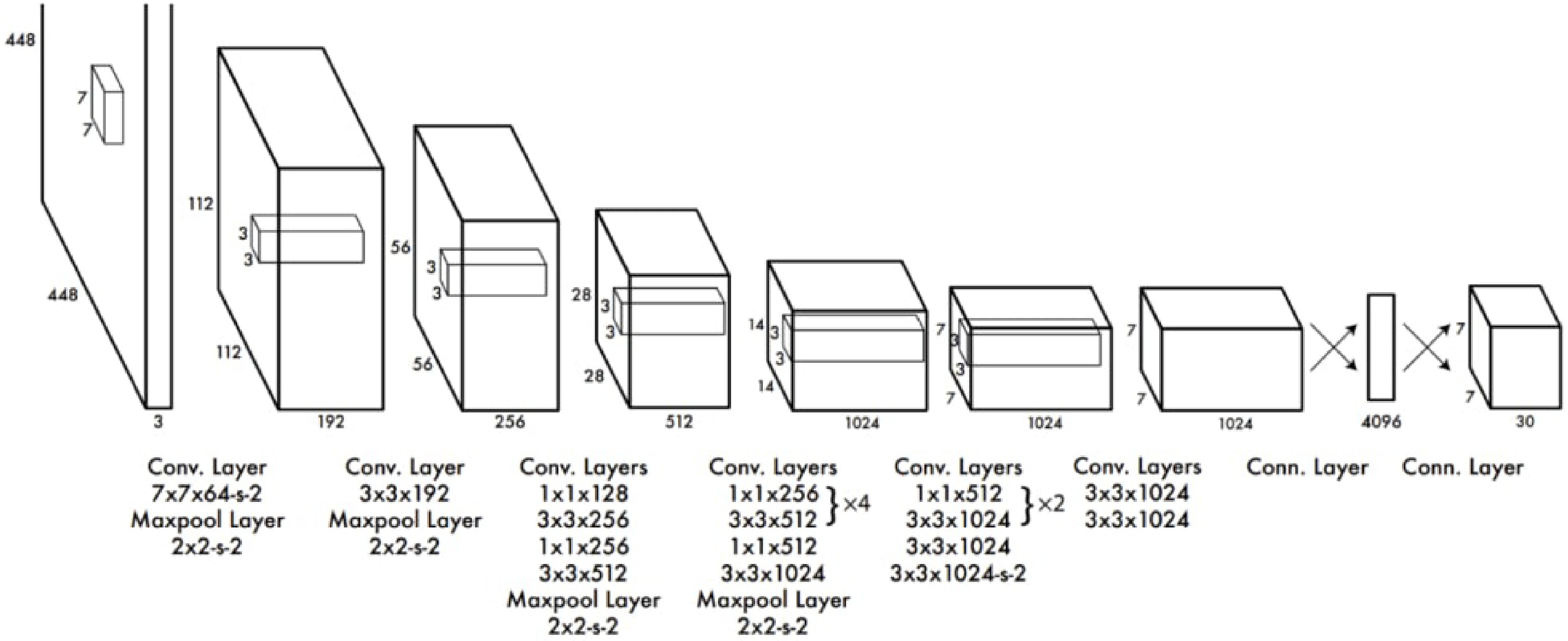
Ren et al. "Faster r-cnn: Towards real-time object detection with region proposal networks." *NeurIPS* 2015. [Paper]

Some results



You Only Lock Once: YOLO

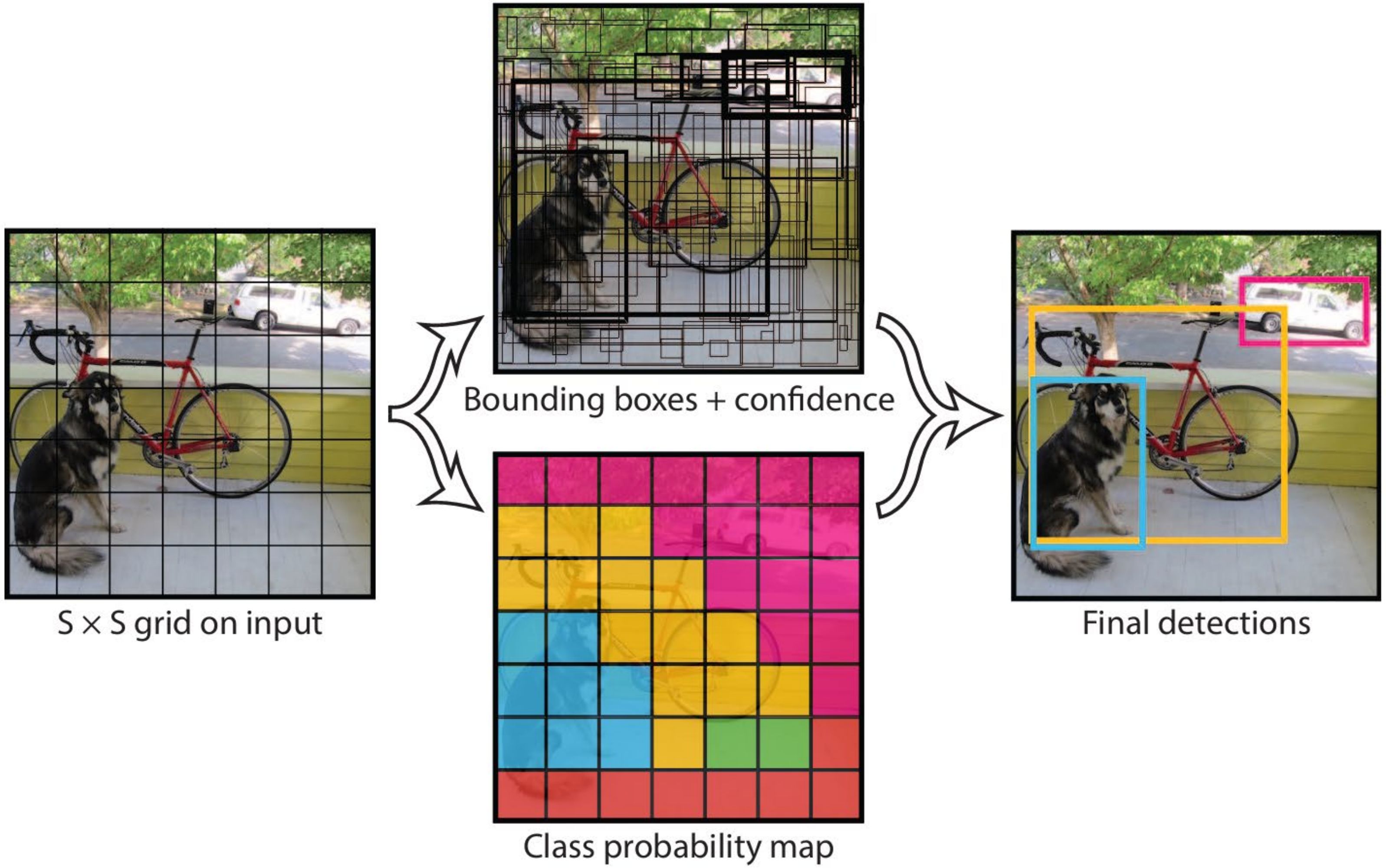
YOLO: from input image to tensor scores with one single convolutional network



The Architecture. Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1×1 convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224×224 input image) and then double the resolution for detection.

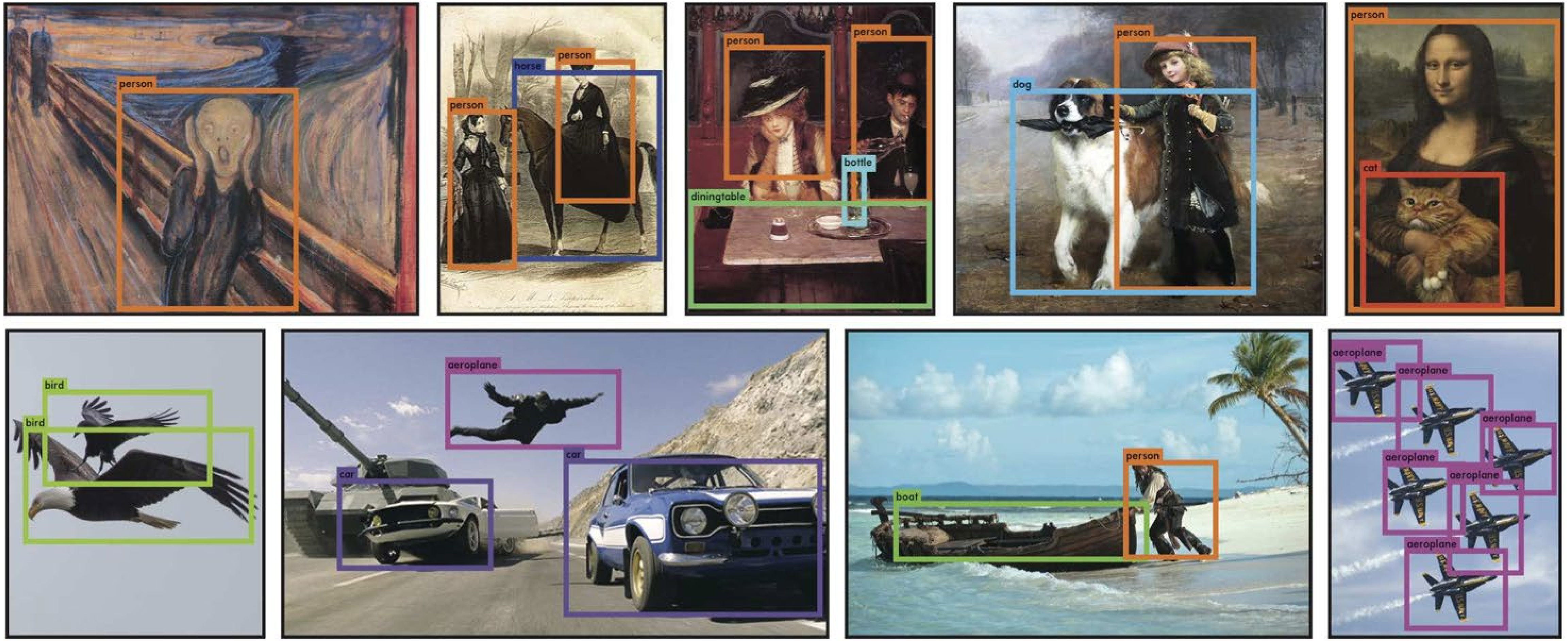
Redmon et al. "You only look once: Unified, real-time object detection." CVPR 2016. [Paper]

You Only Lock Once: YOLO



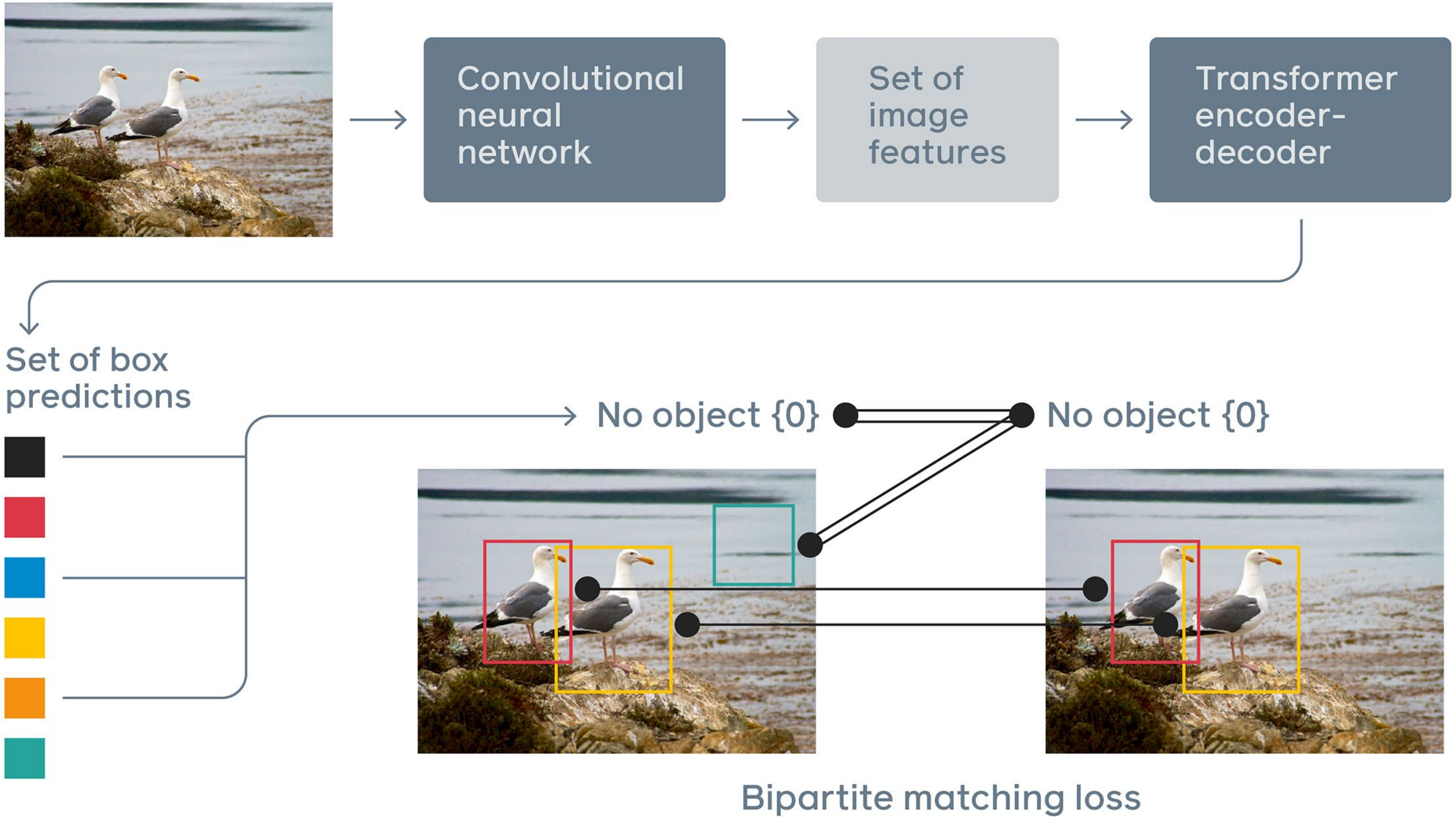
Redmon et al. "You only look once: Unified, real-time object detection." CVPR 2016. [Paper]

You Only Lock Once: YOLO



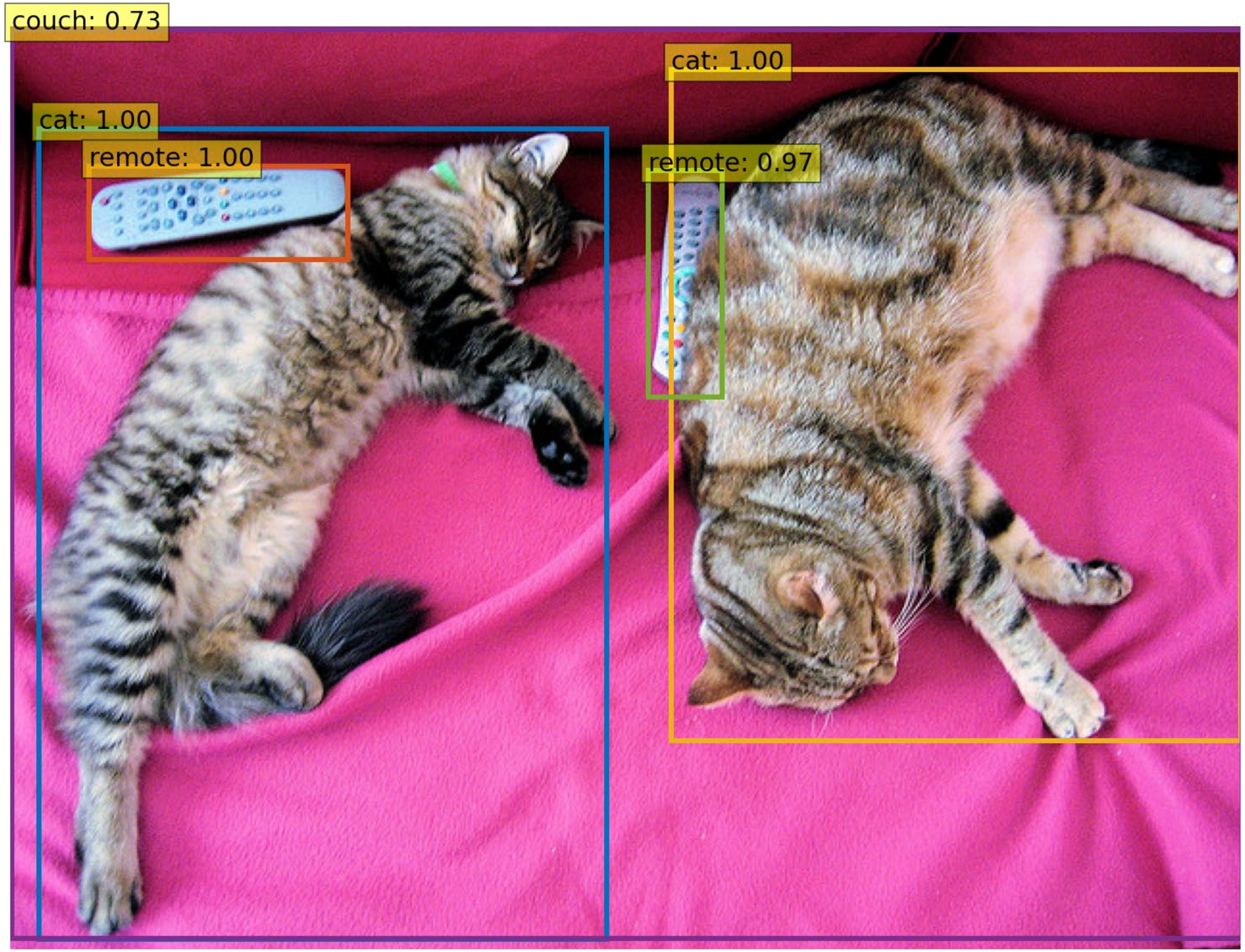
Redmon et al. "You only look once: Unified, real-time object detection." CVPR 2016. [Paper]

DETR



Carion et al. "End-to-end object detection with transformers." ECCV 2020. [Paper][GitHub]

DETR



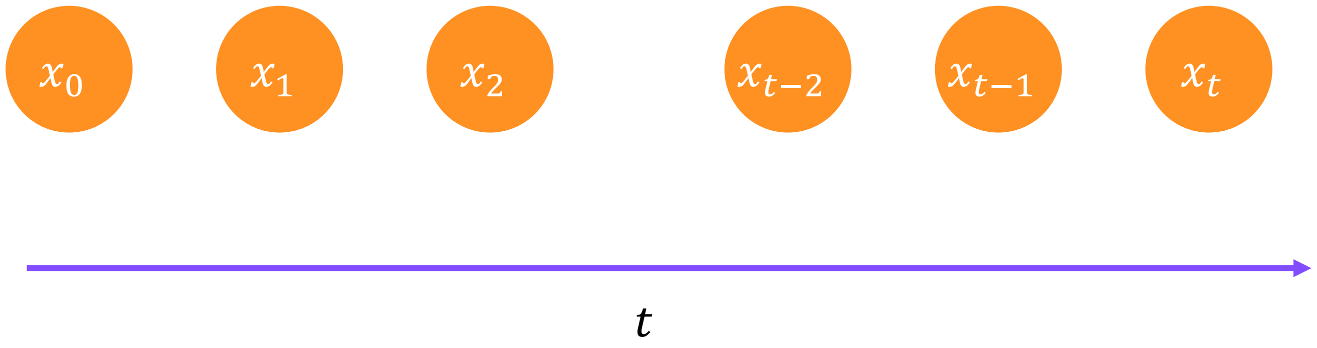
Carion et al. "End-to-end object detection with transformers." ECCV 2020. [Paper][GitHub]

TRANSFORMERS

Goal of sequence modeling

RNNs use recurrence to model sequence dependencies

Sequence of inputs

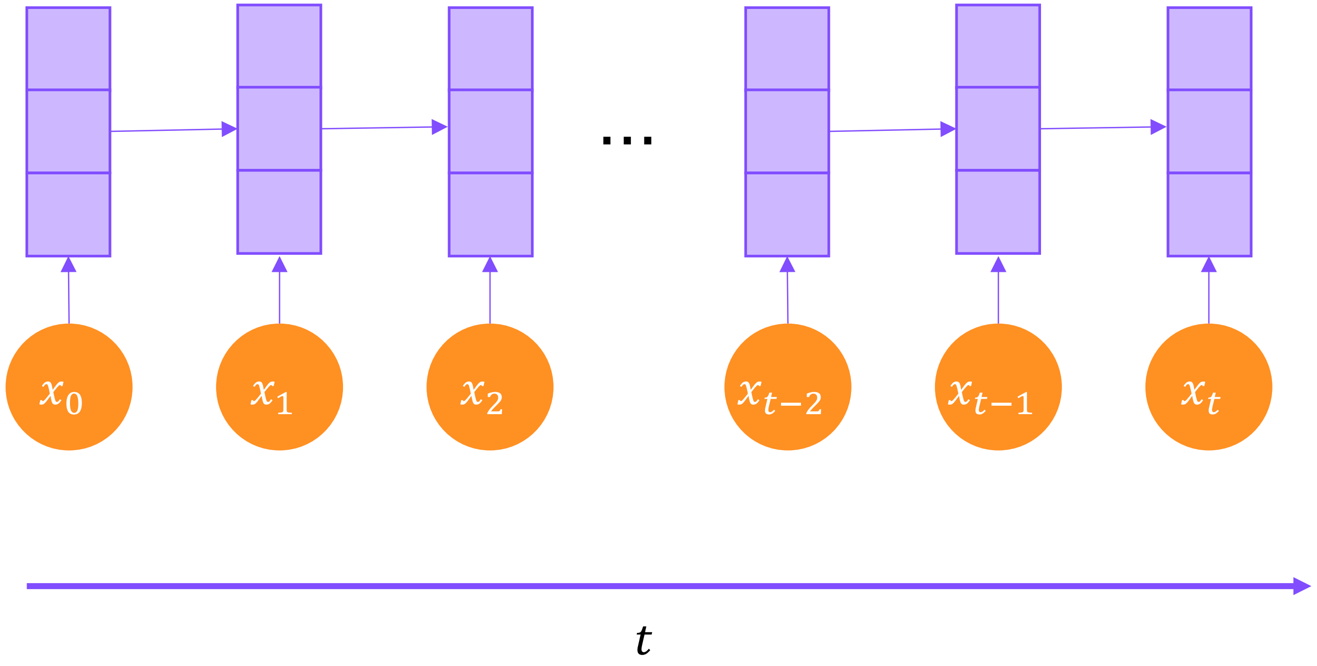


Goal of sequence modeling

RNNs use recurrence to model sequence dependencies

Sequence of features

Sequence of inputs



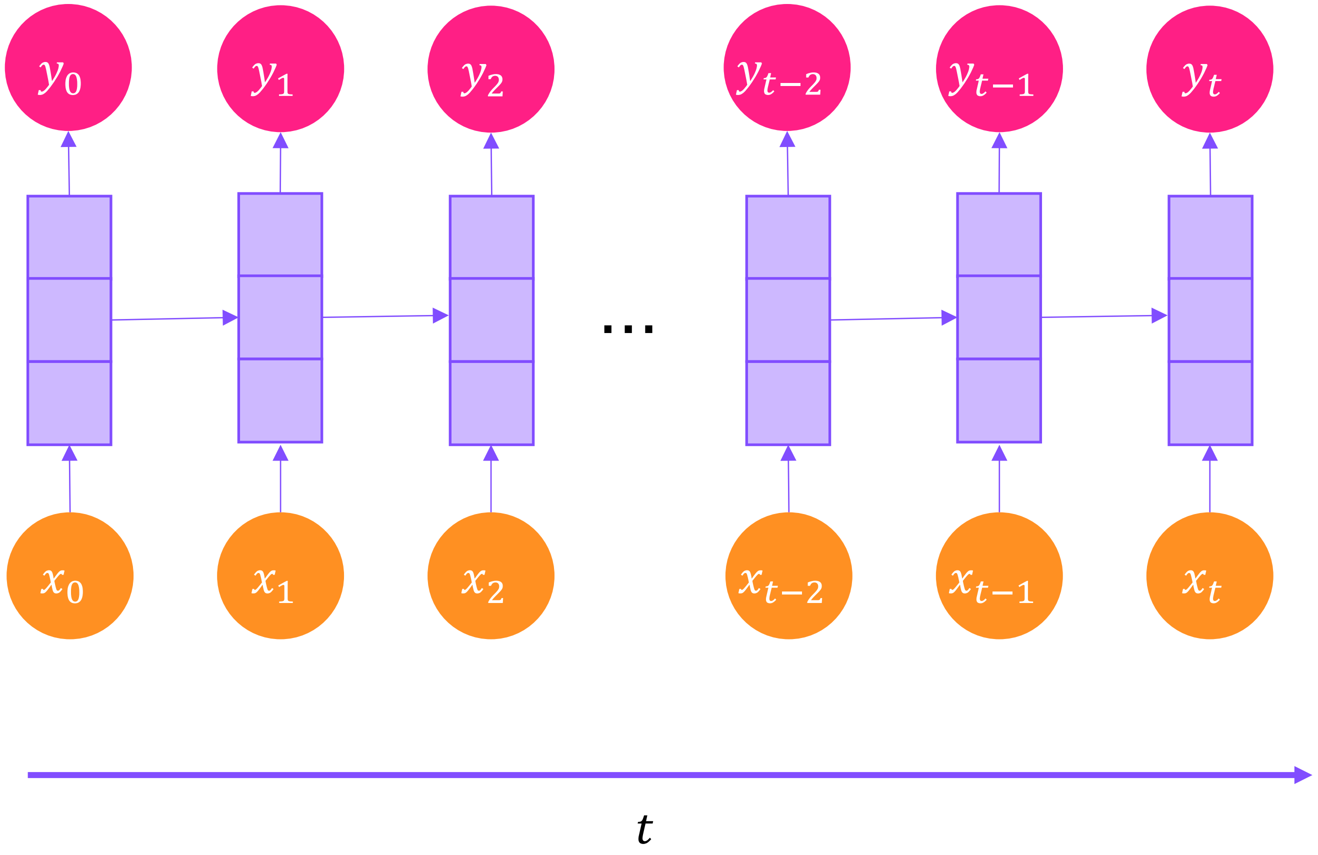
Goal of sequence modeling

RNNs use recurrence to model sequence dependencies

Sequence of outputs

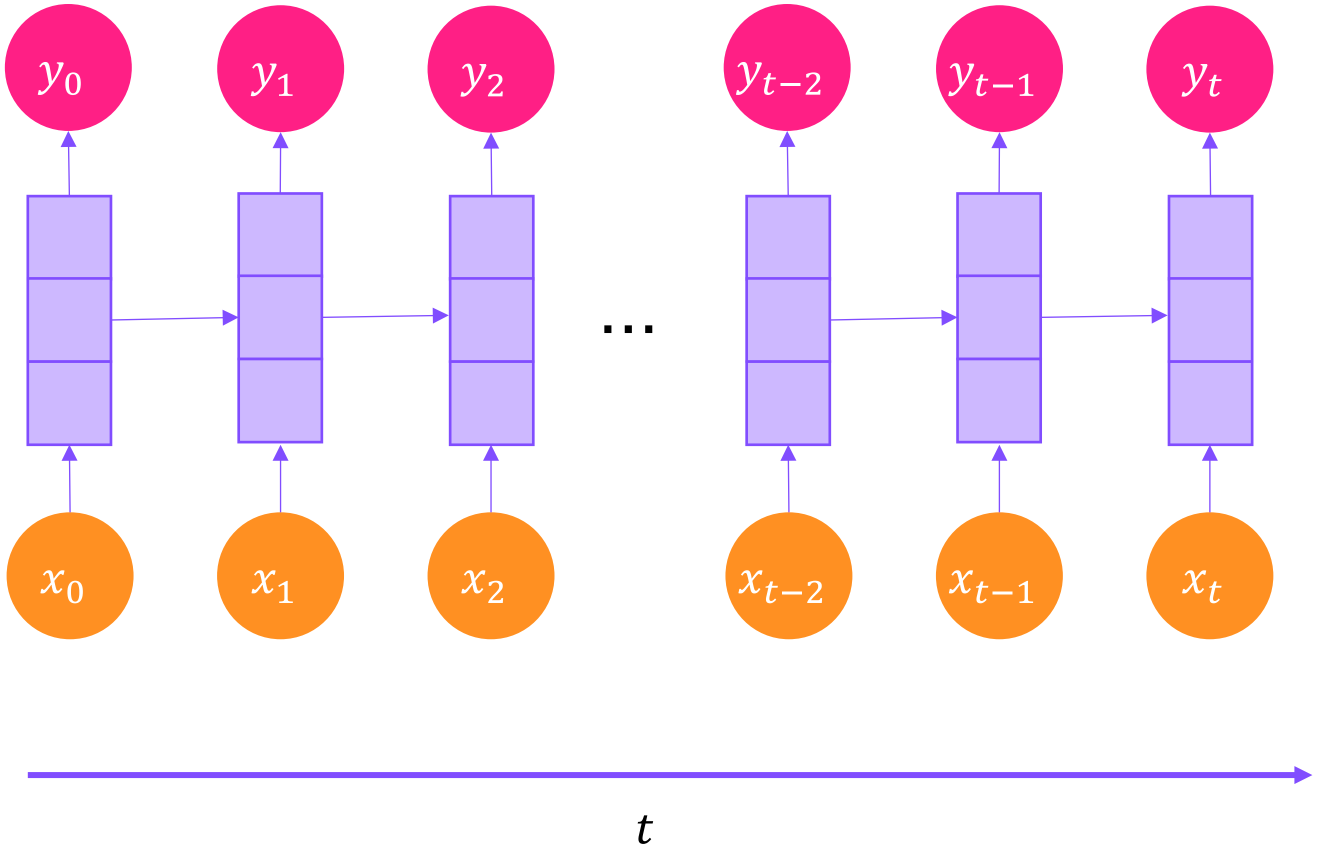
Sequence of features

Sequence of inputs



Goal of sequence modeling

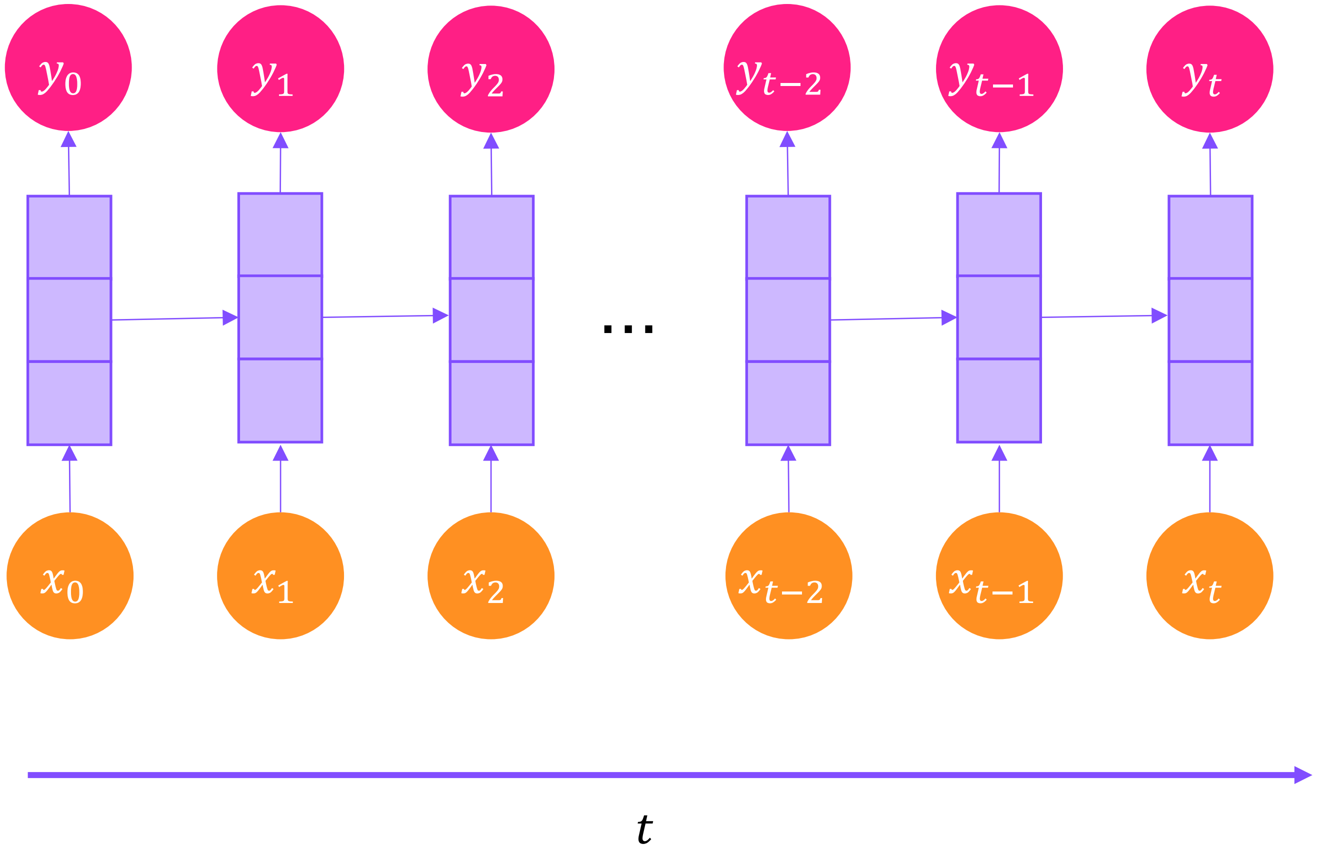
- Encoding bottleneck
- Slow, no parallelization
- Not long memory



Goal of sequence modeling

- Continous stream
- Parallelize
- Long memory

How can we eliminate the need for recurrence?

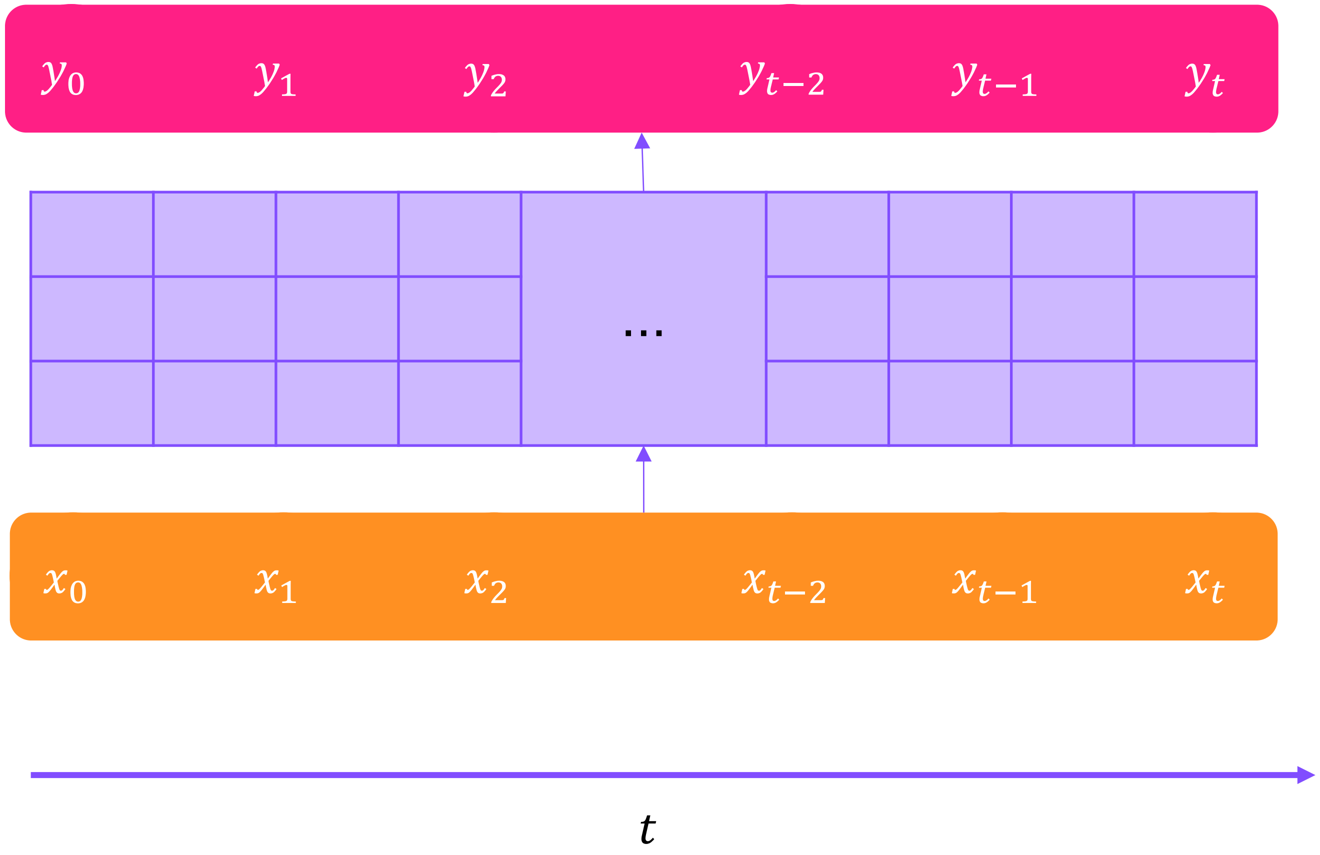


Goal of sequence modeling

How can we eliminate the need for recurrence?

Idea: feed everything as dense networks

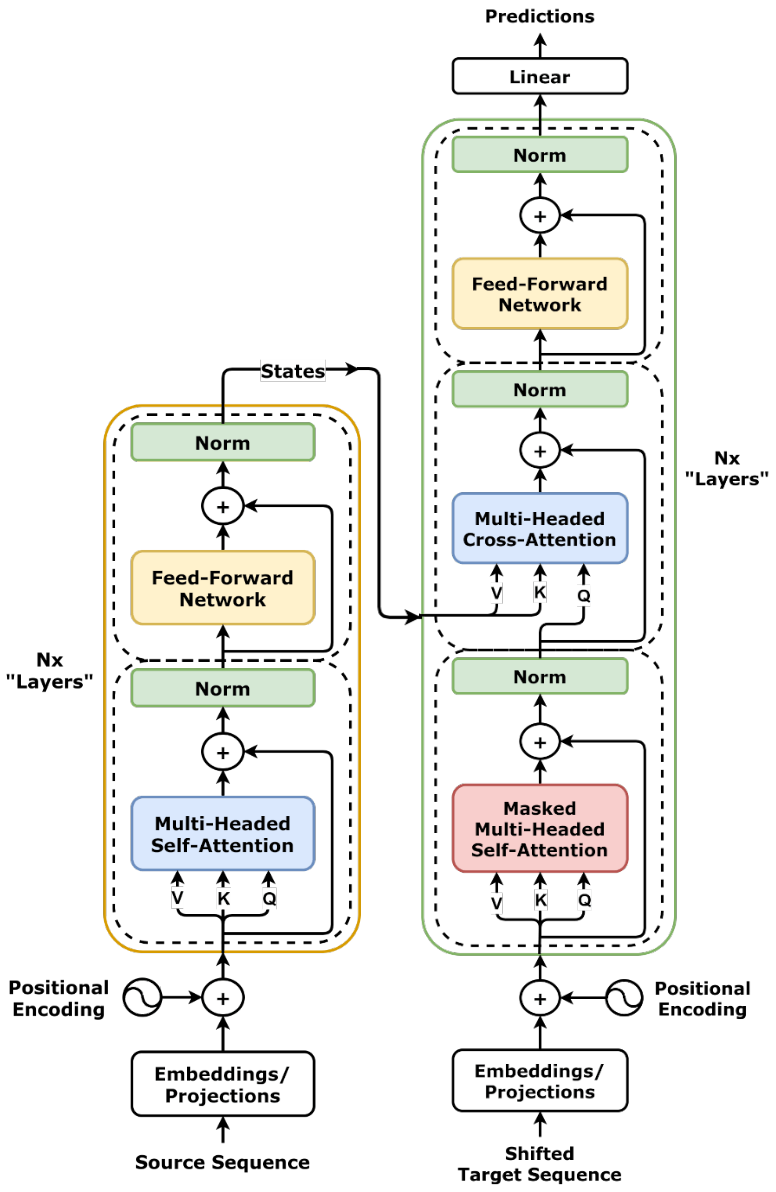
- ✓ No recurrence
- ✗ Not scalable
- ✗ No order
- ✗ No long memory



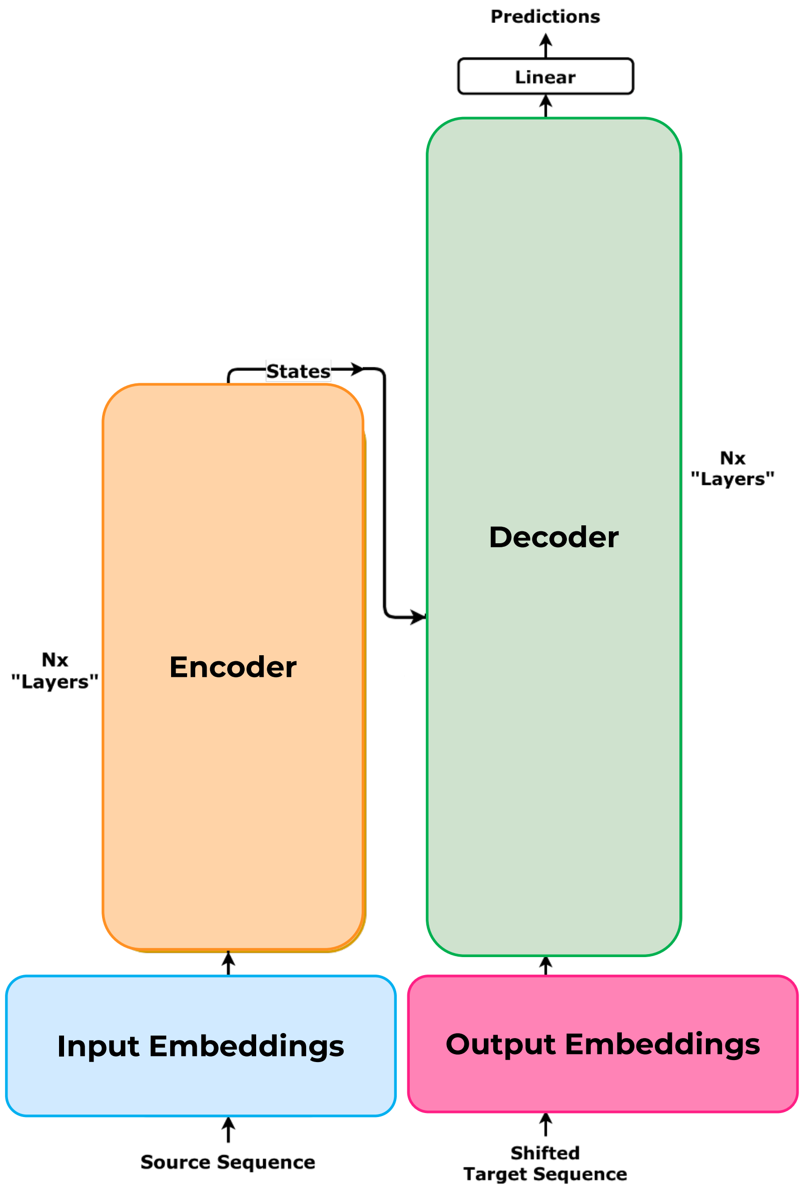
TRANSFORMERS ARCHITECTURE

The Encoder and Decoder stacks

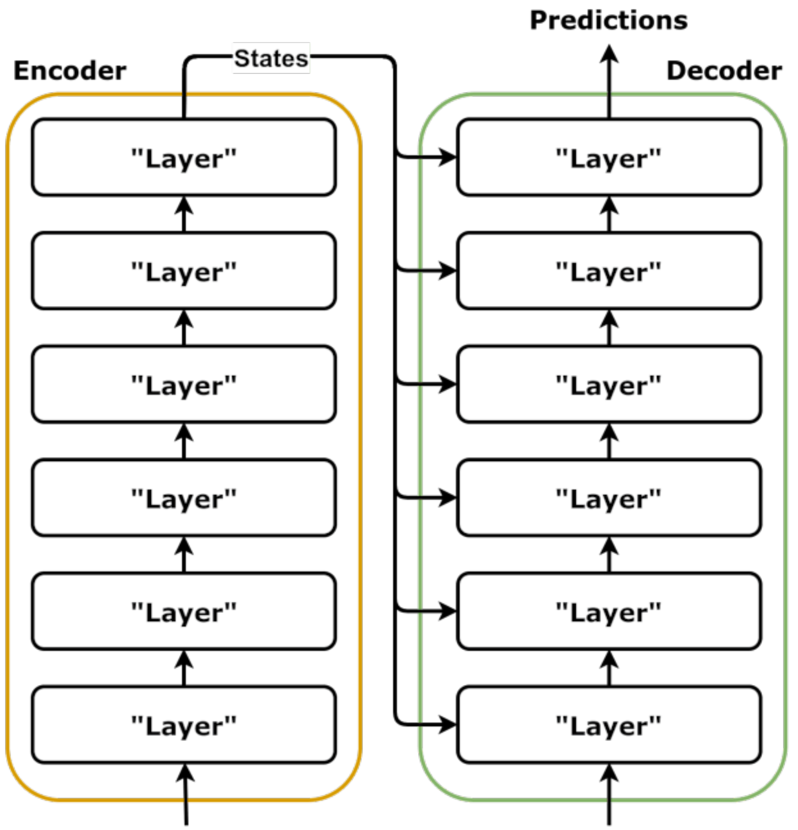
Architecture

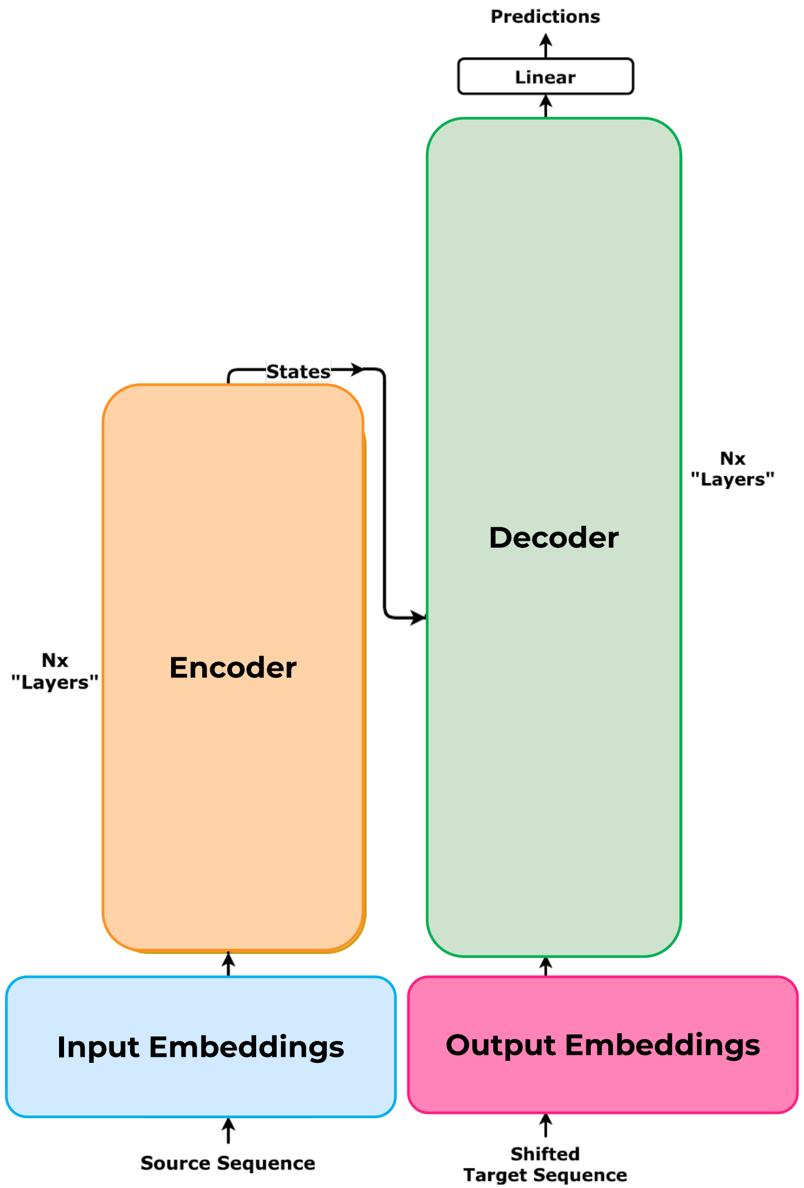


Architecture

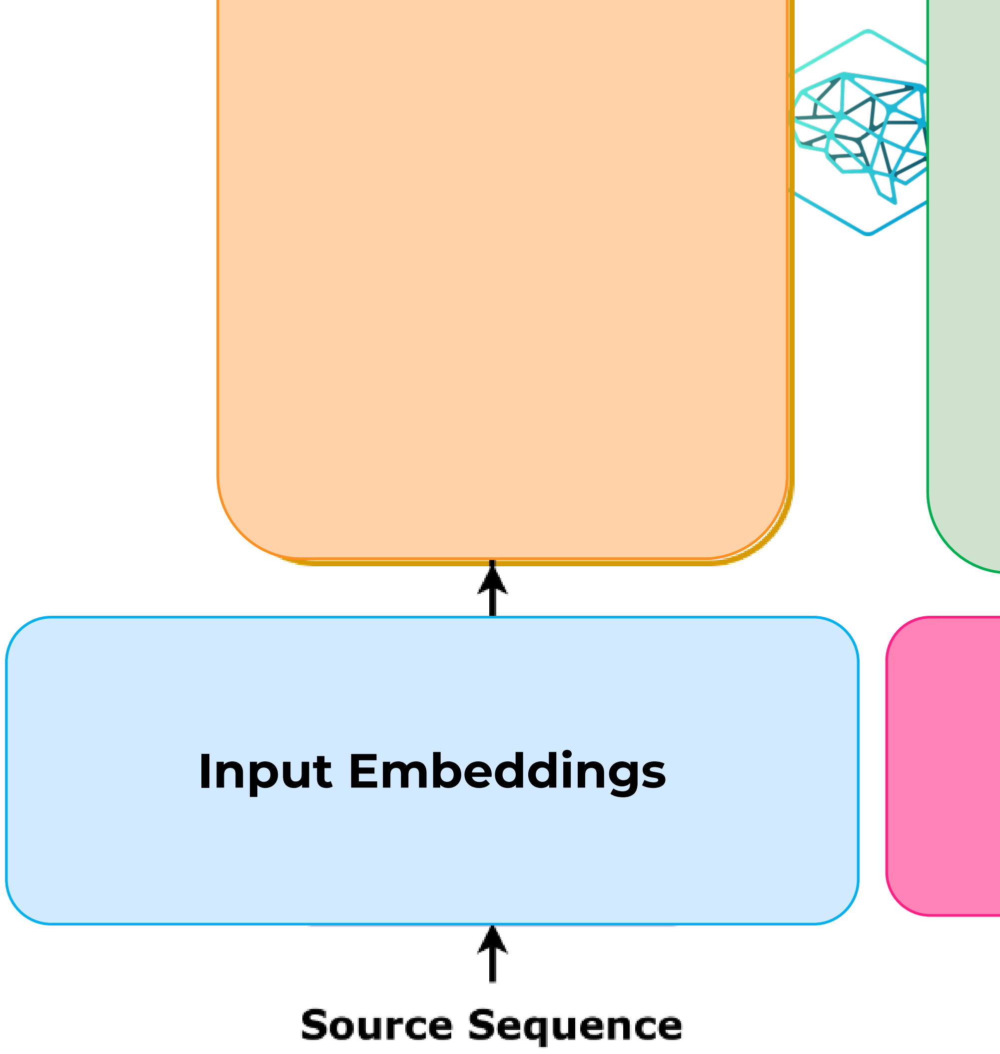


Architecture



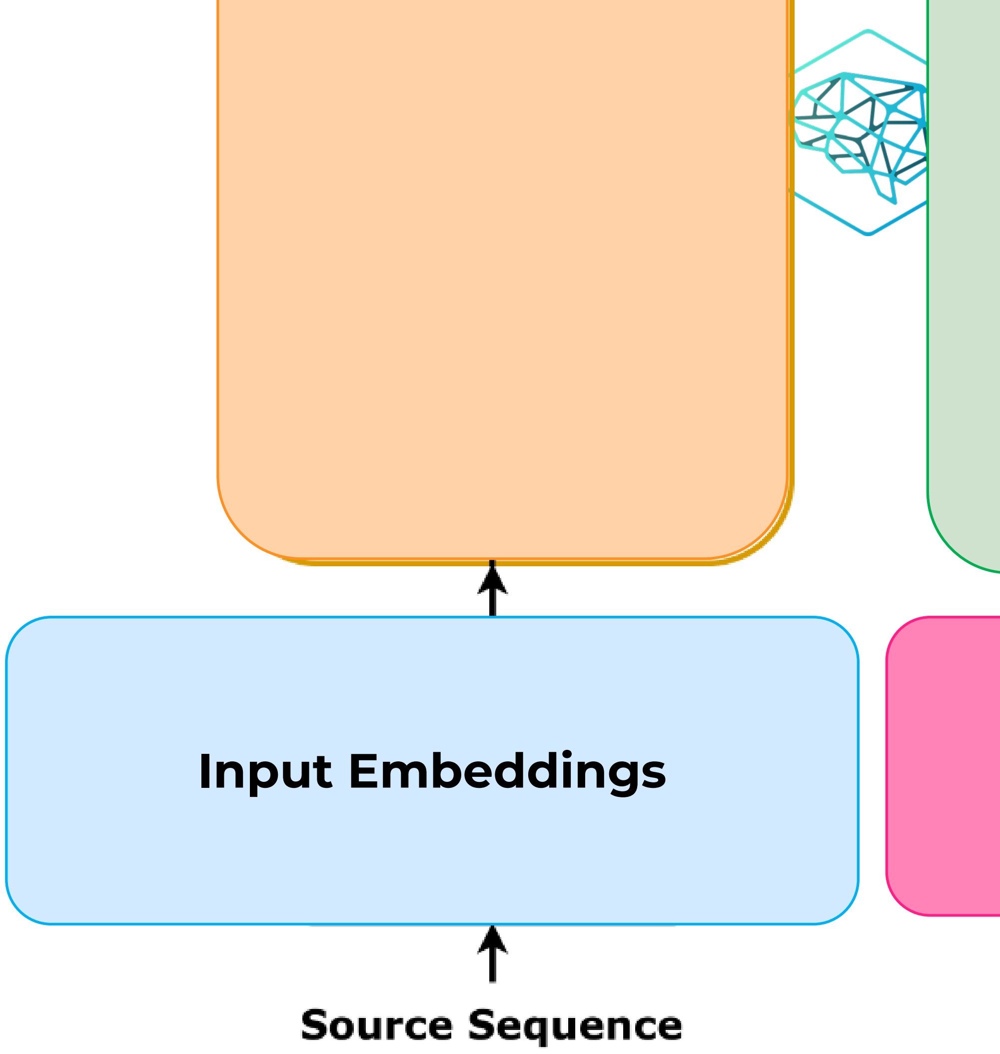


Architecture



Architecture

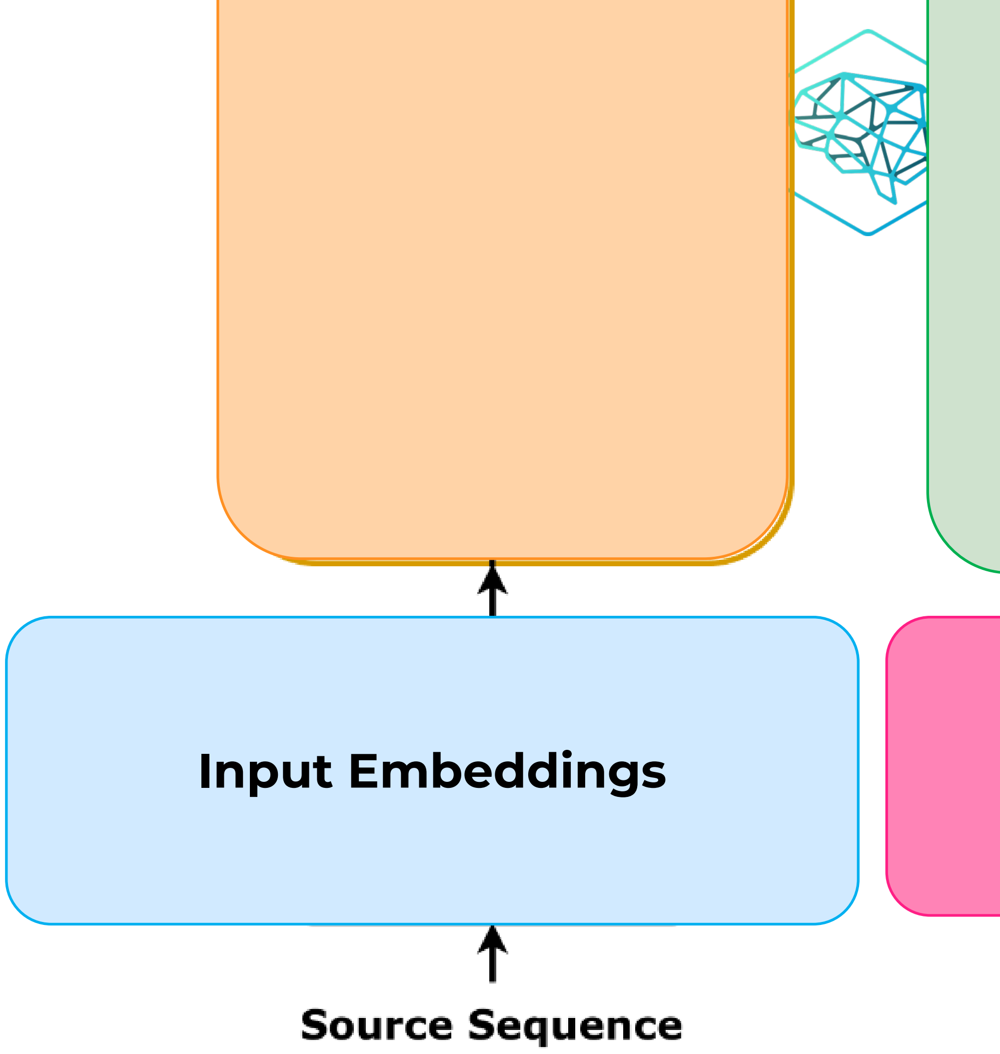
What is the source sequence?



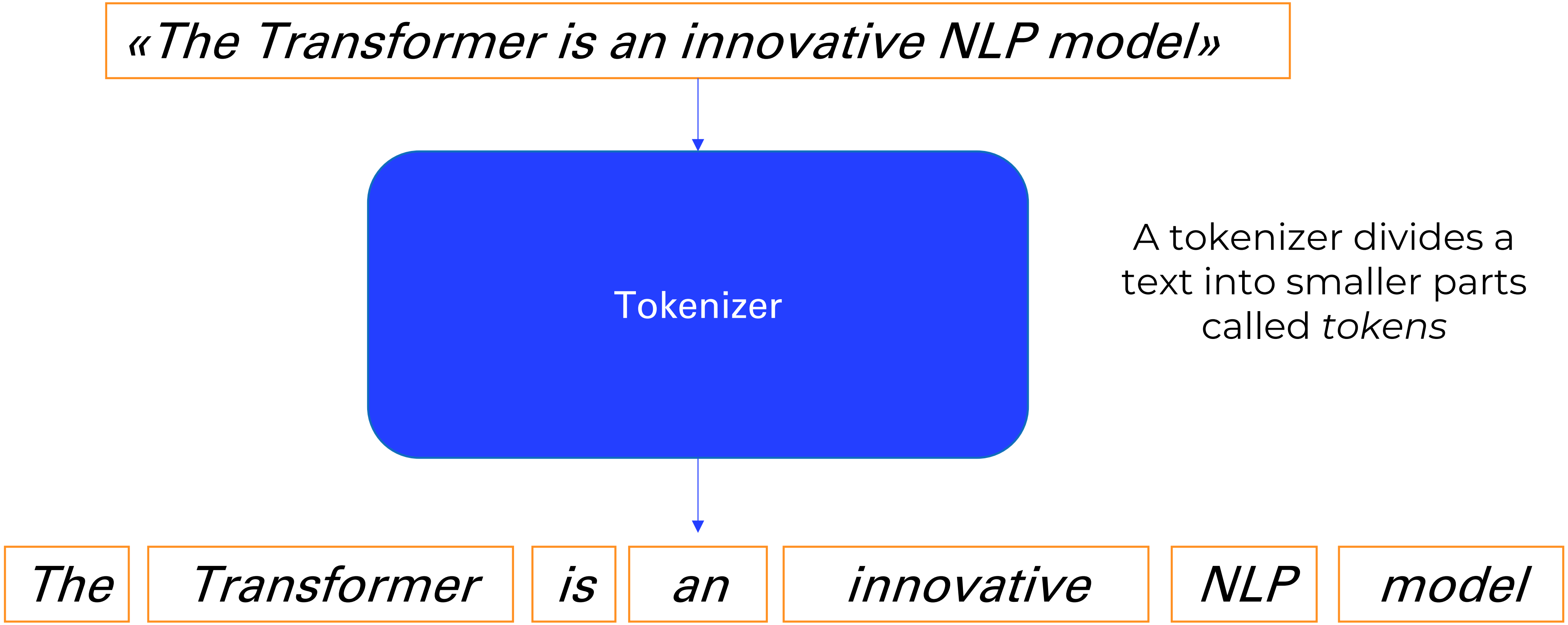
Architecture

What is the source sequence?

«The Transformer is an innovative NLP model»



How can we model text?



A tokenizer divides a text into smaller parts called *tokens*

How can we model text?

«The Transformer is an innovative NLP model»

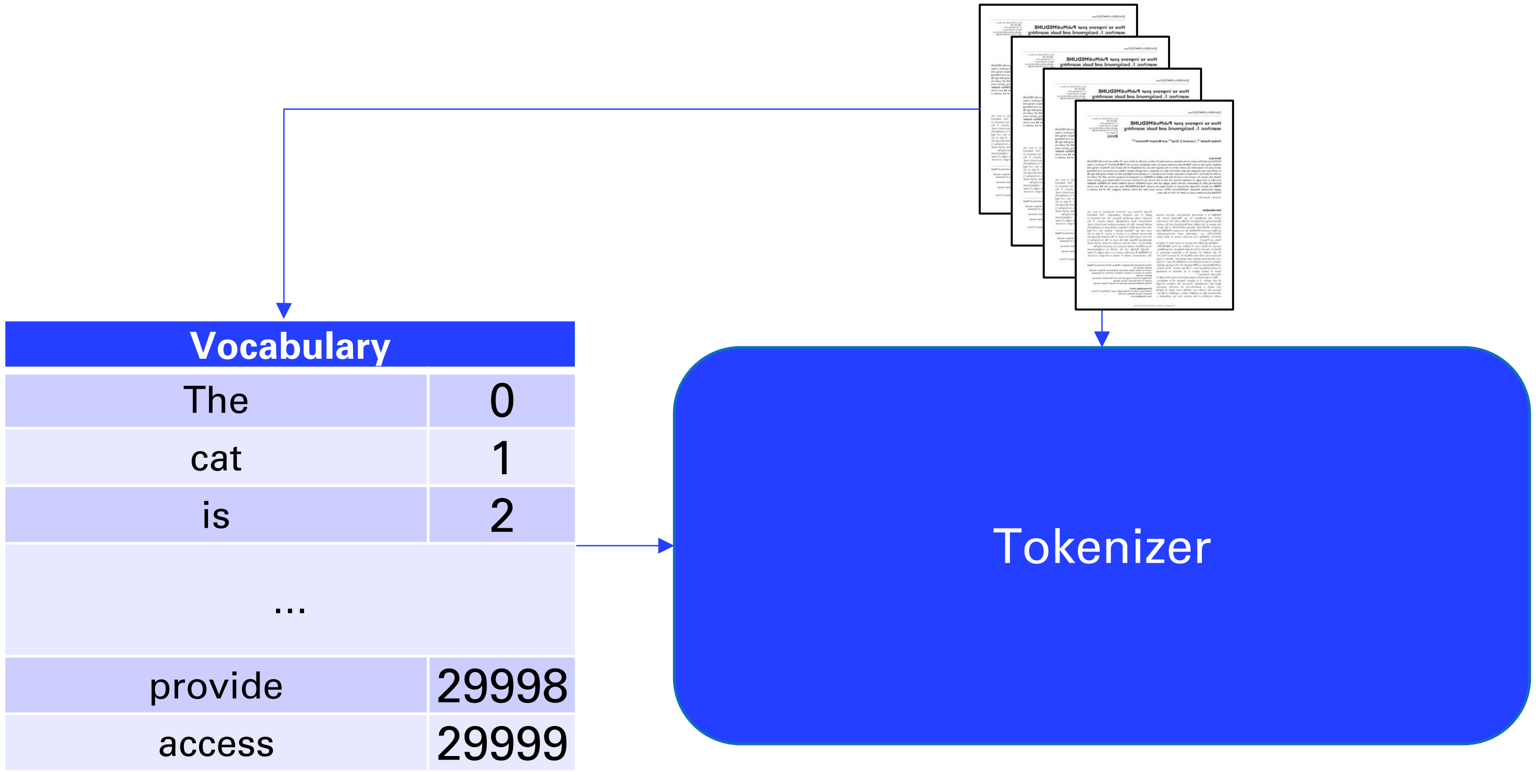
Vocabulary	
The	0
Transformer	1
is	2
an	3
innovative	4
NLP	5
model	6

Tokenizer

It uses a vocabulary to convert *tokens* into *token ids*

0	1	2	3	4	5	6

How can we model text?



We don't train a model with a single sentence…
…we use *corpora*

How can we model text?

«*The Transformer is an innovative NLP model*»

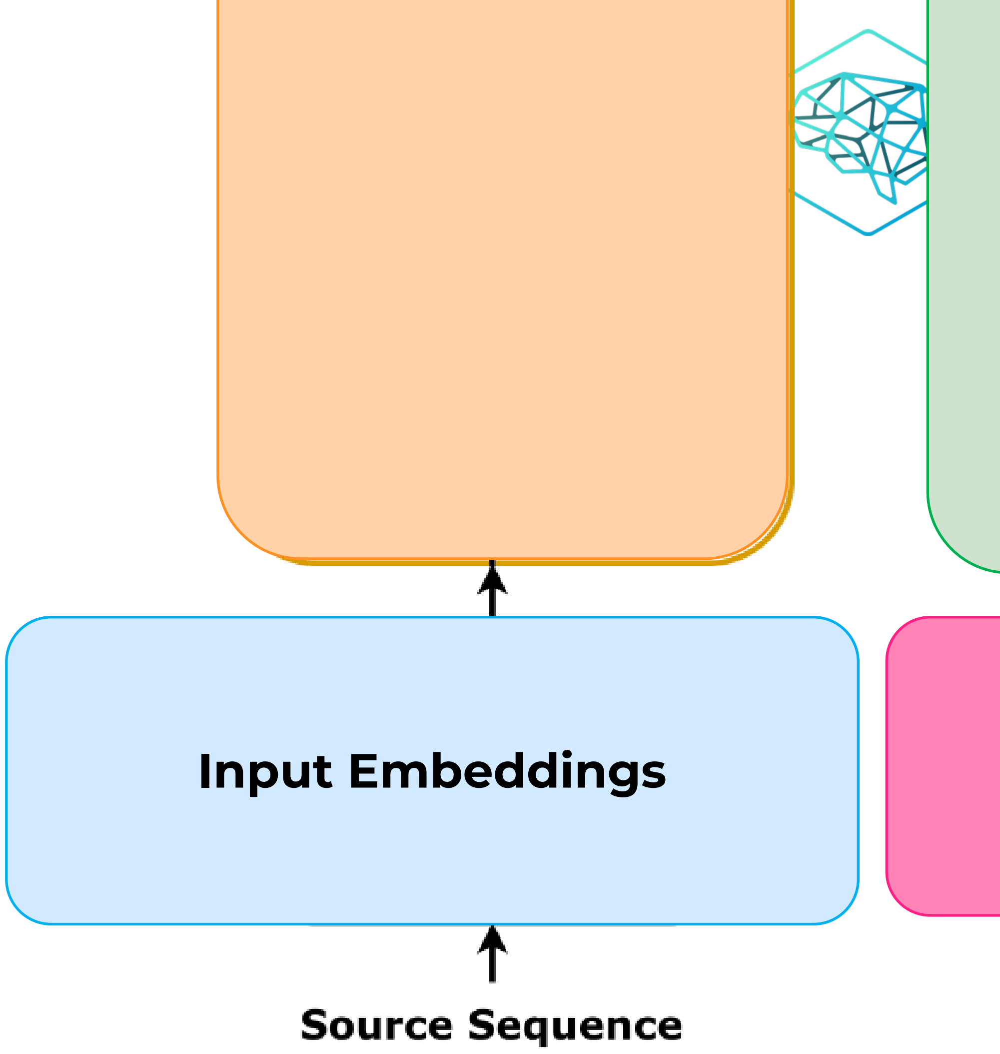
Vocabulary	
The	0
cat	1
is	2
...	
provide	29998
access	29999

Tokenizer

In a vocabulary we usually have thousands of *tokens*

0	1489	2	67	13679	946	103

Architecture



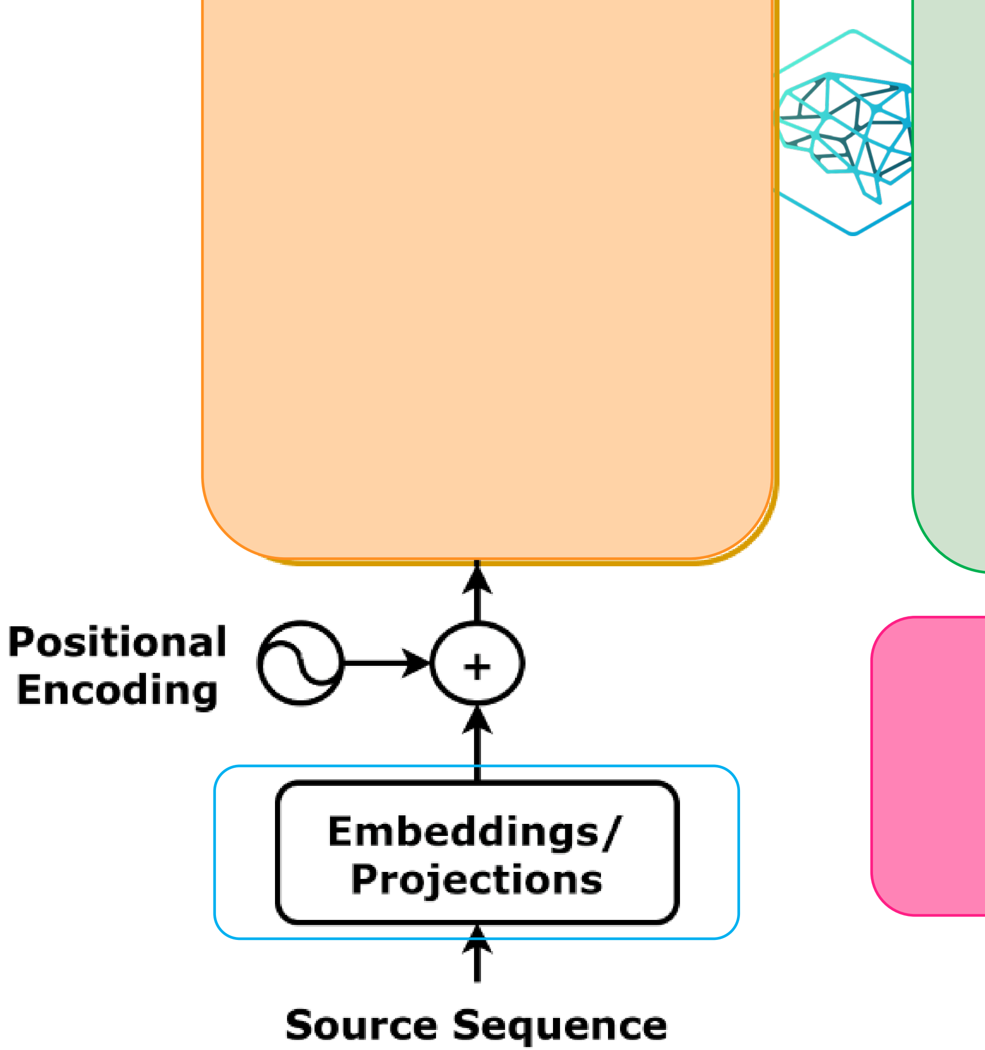
What is the source sequence?

x_0 x_1 x_2 … x_{t-1} x_t

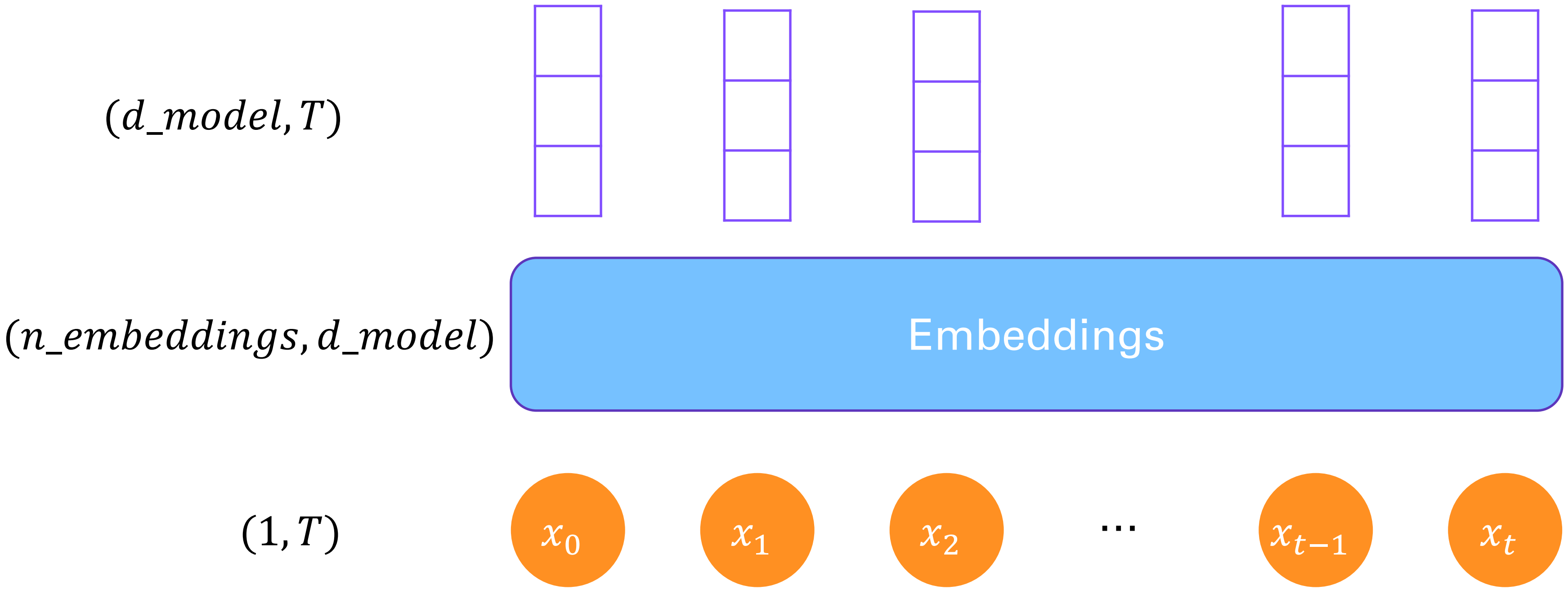
A sequence of tokens
$(1, T)$

T:
- time dimension
- sequence length
- number of tokens

Architecture

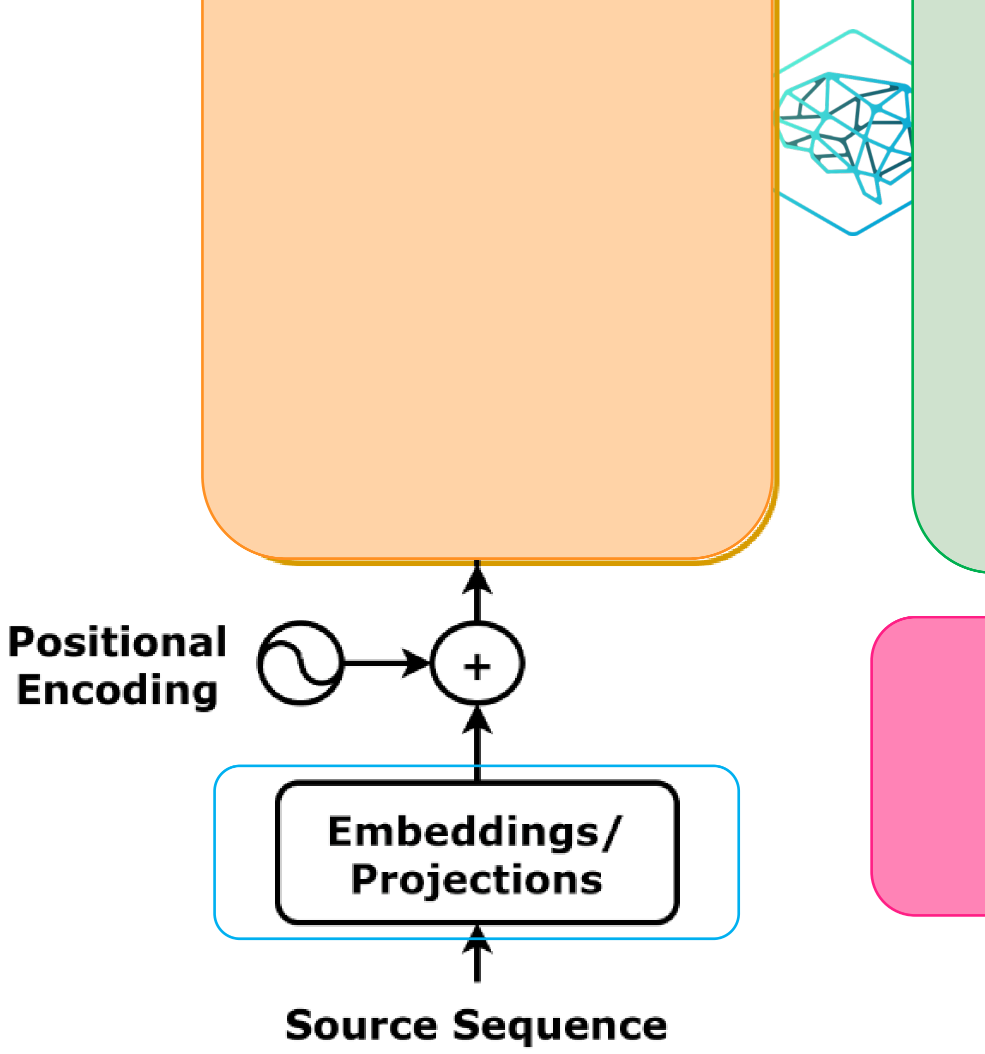


Embeddings



$$out = Wx$$

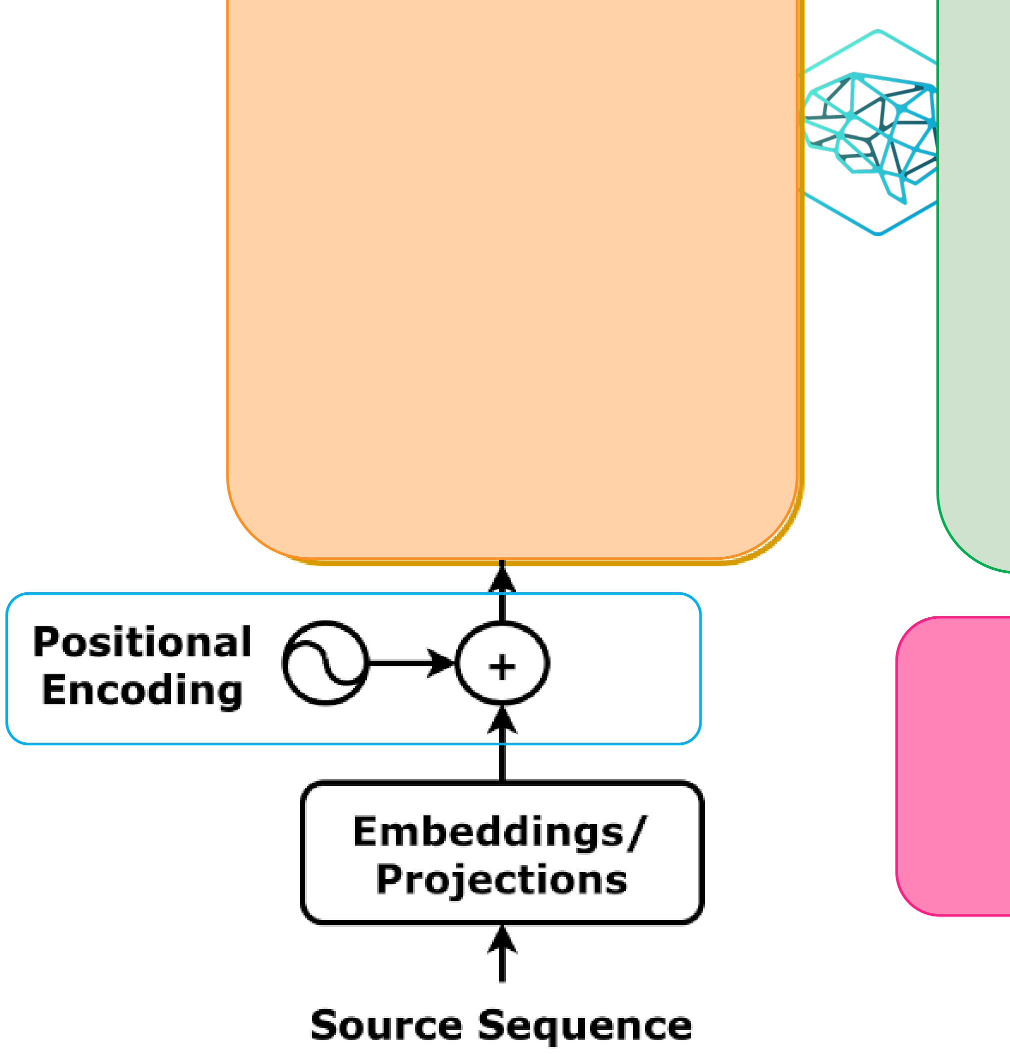
Architecture



Architecture

We have no recurrence

Transformer has no idea of the order of tokens

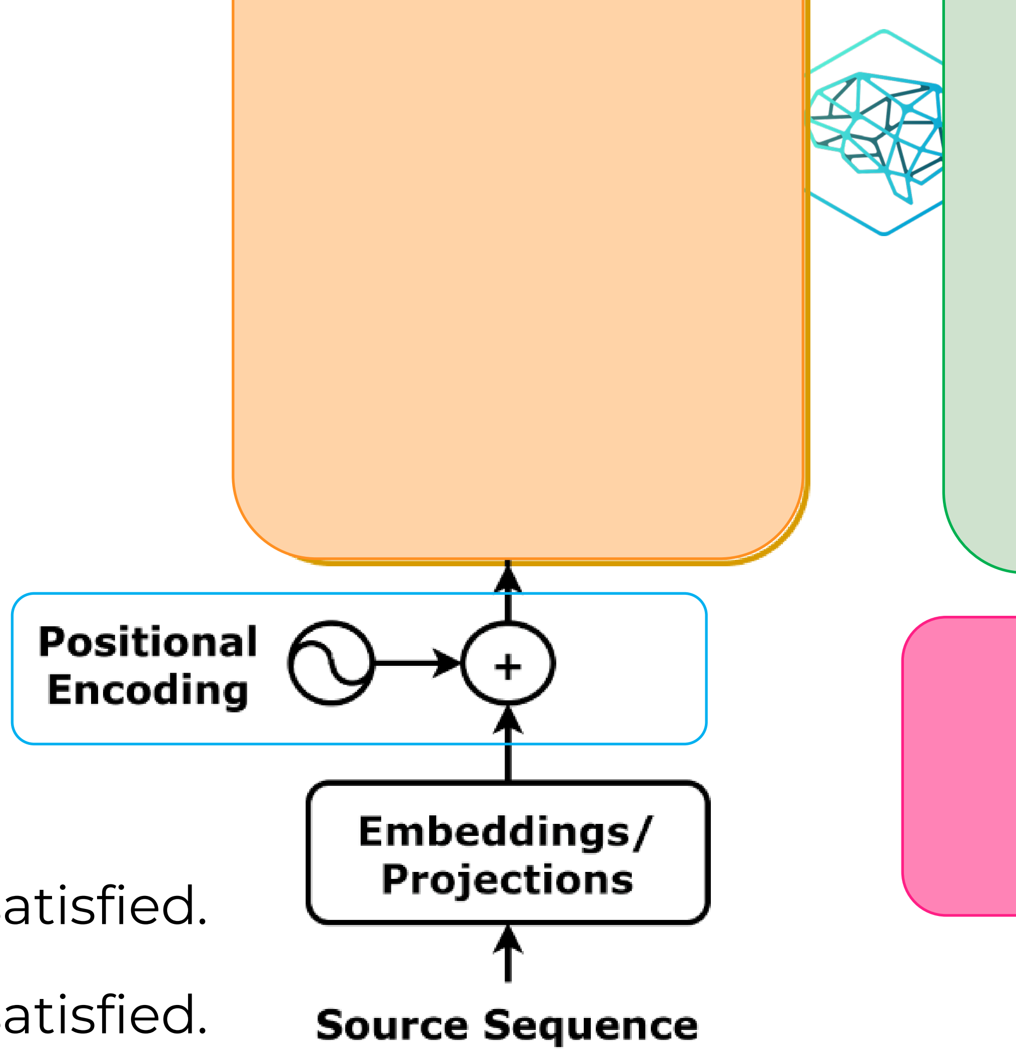


Architecture

Why the order matters?

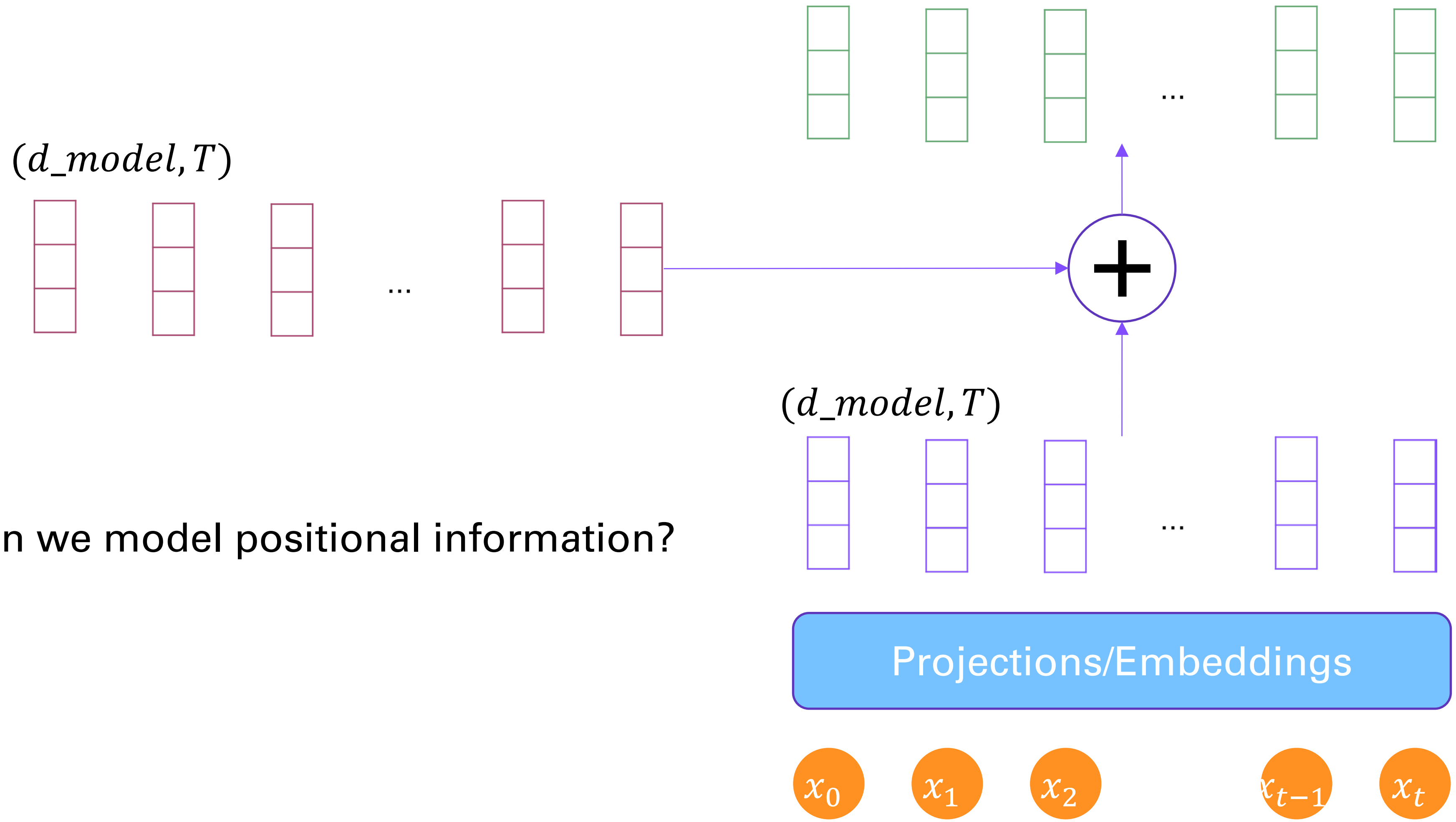
Even though she did not win the award, she was satisfied.

Even though she did win the award, she was not satisfied.



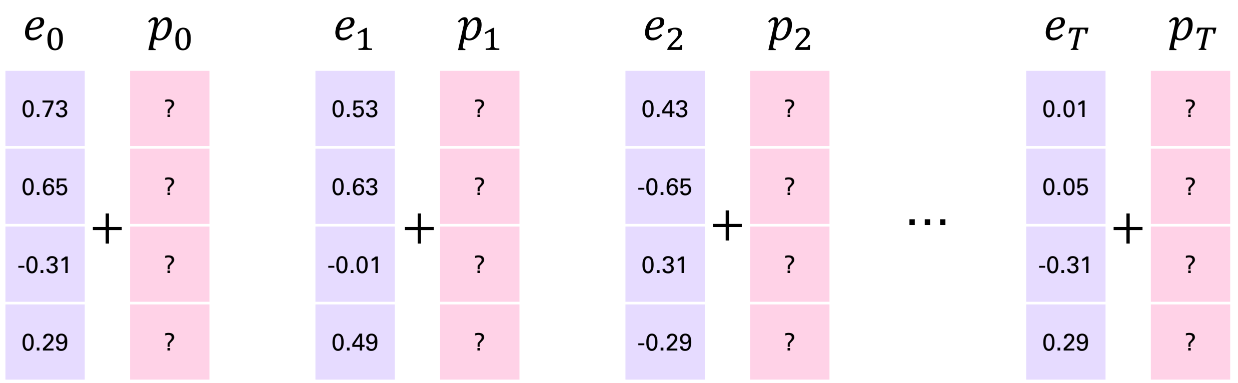
Positional Encoding

How can we model positional information?

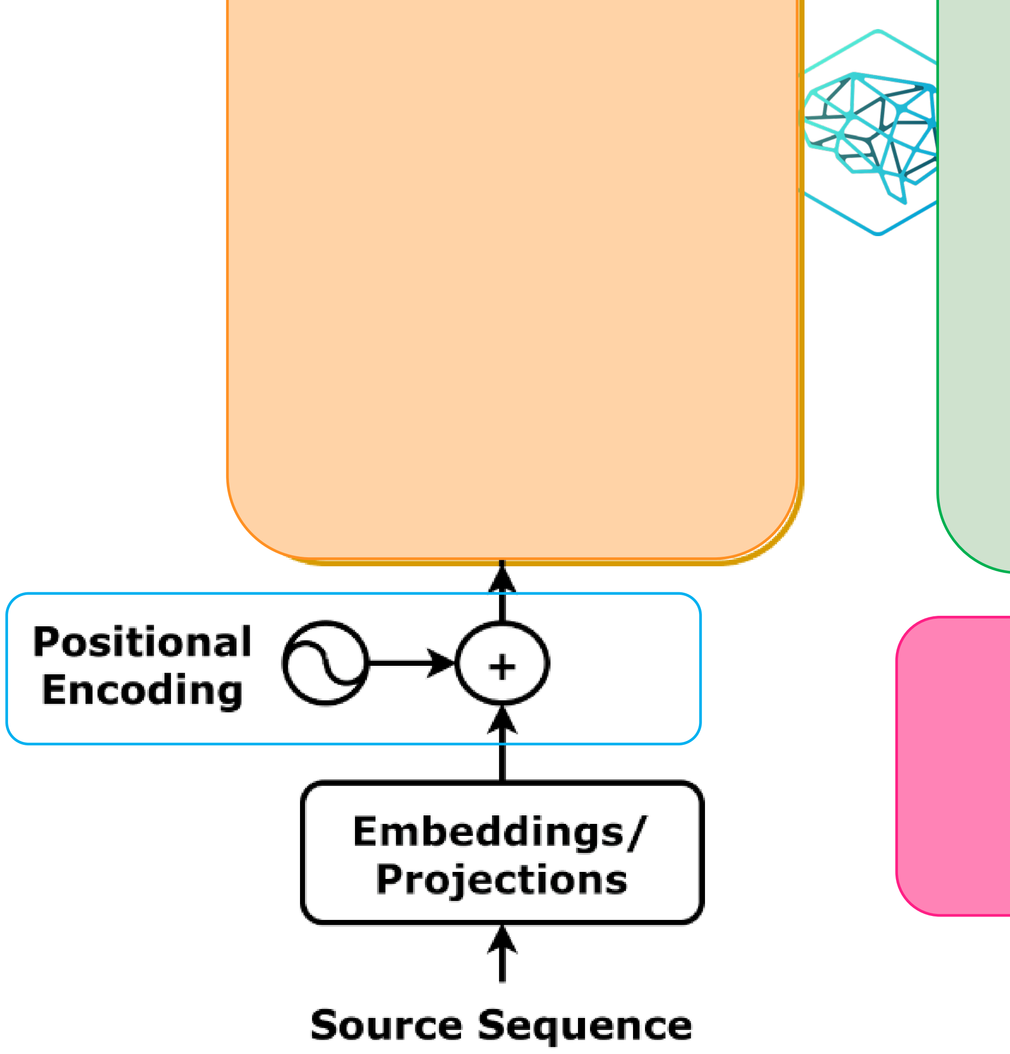


How can we model positional information?

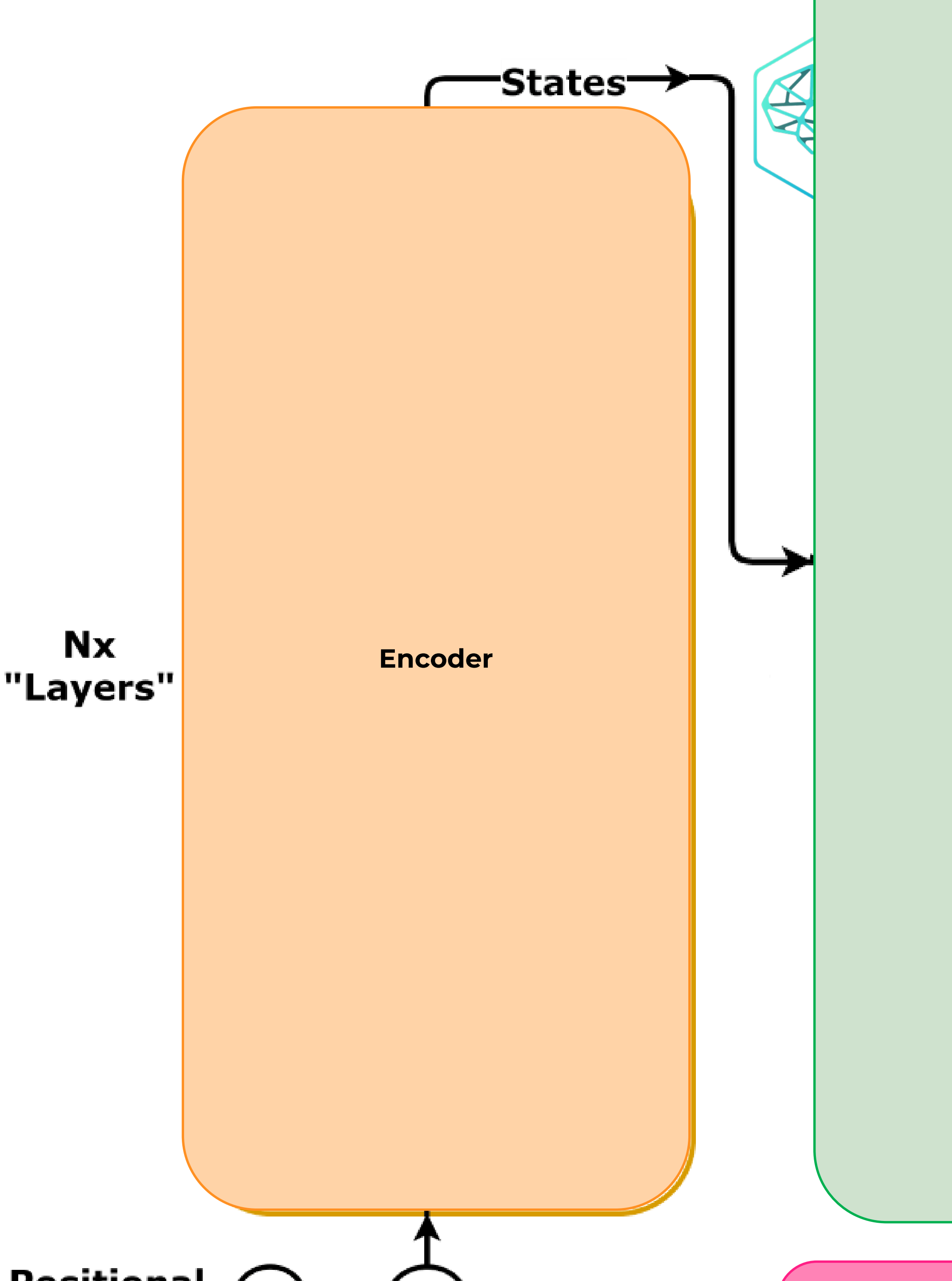
- Absolute index
- Normalized index
- *sin* and *cos*
- Learnable parameters



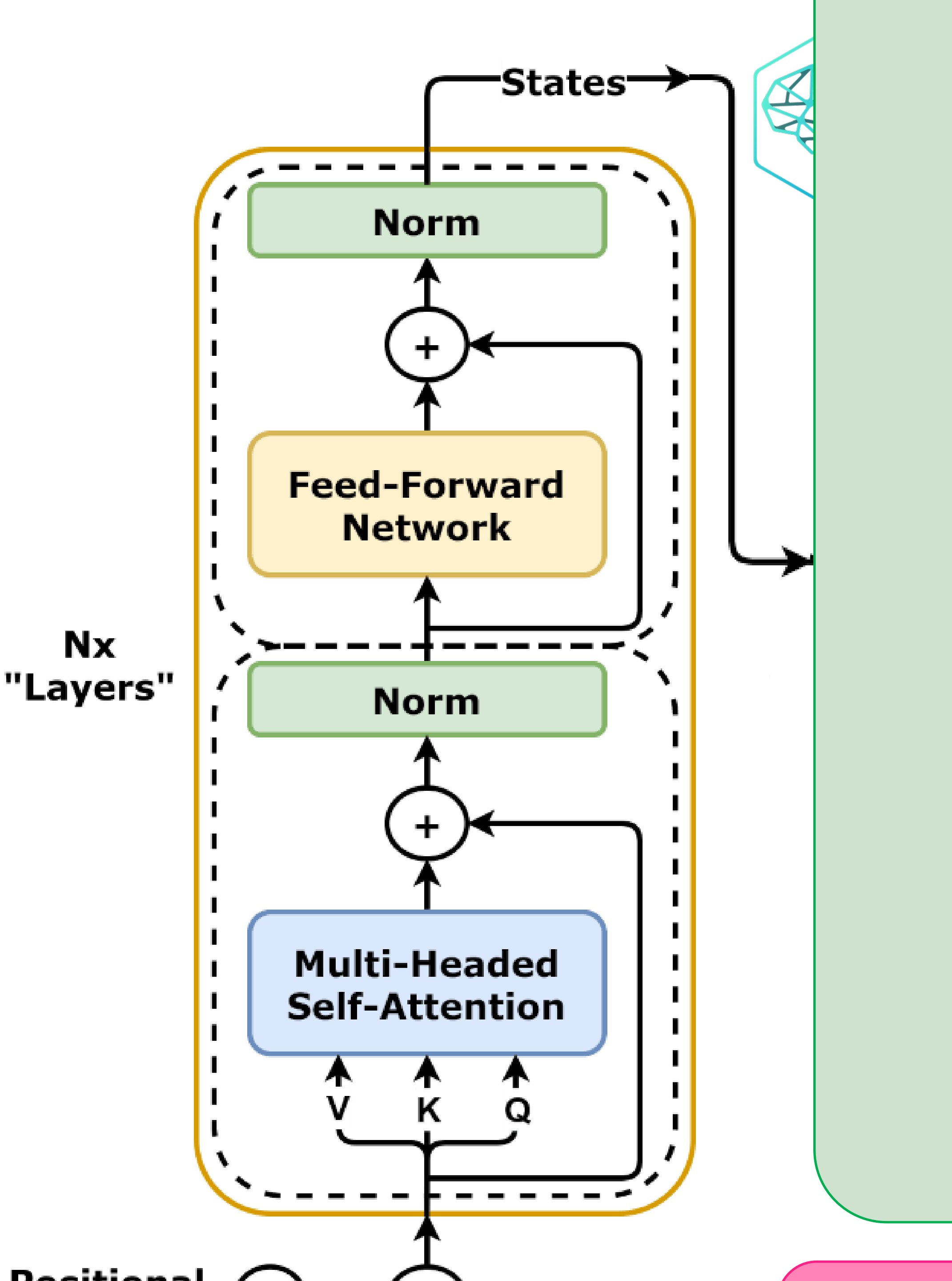
Architecture



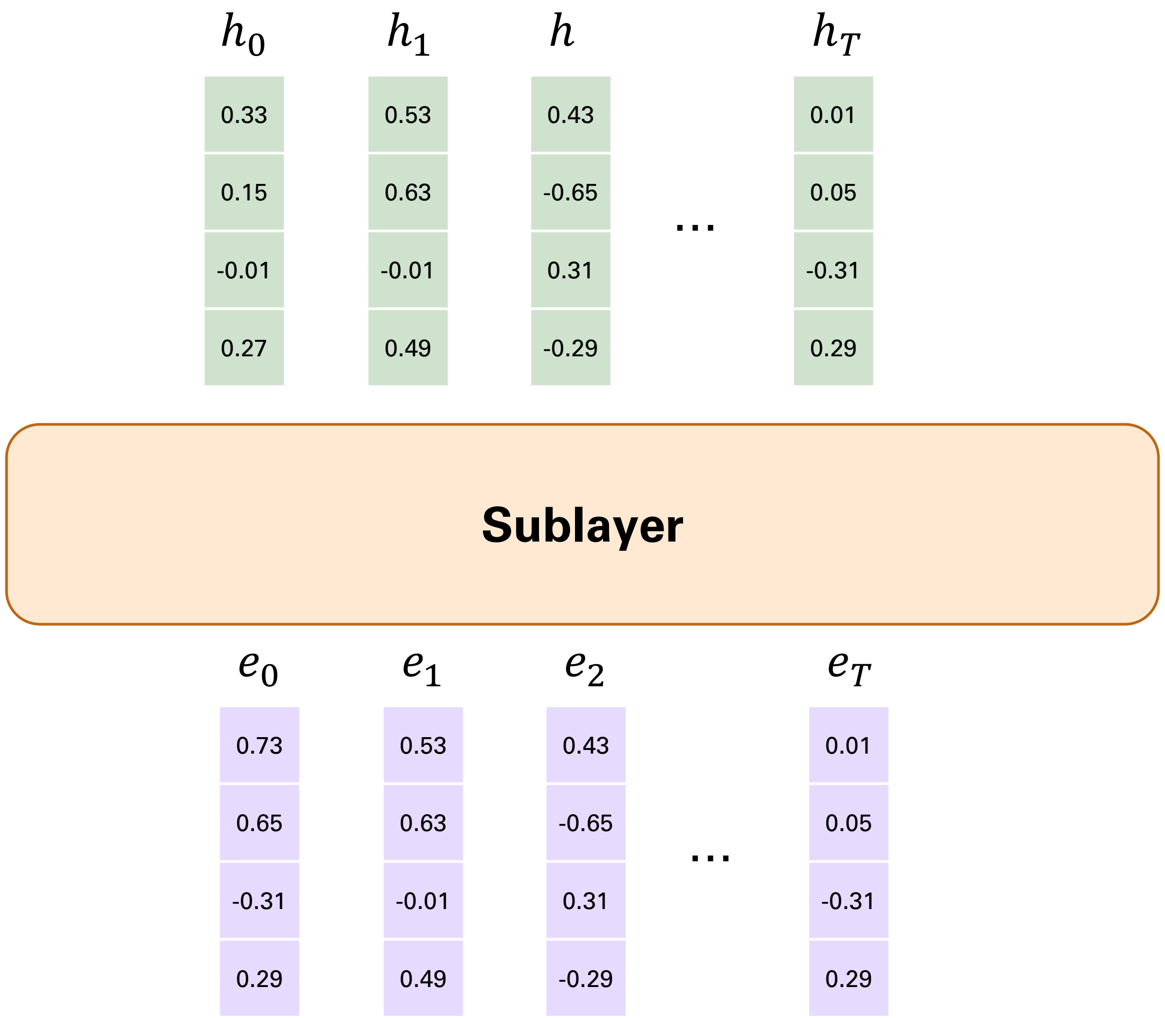
Architecture

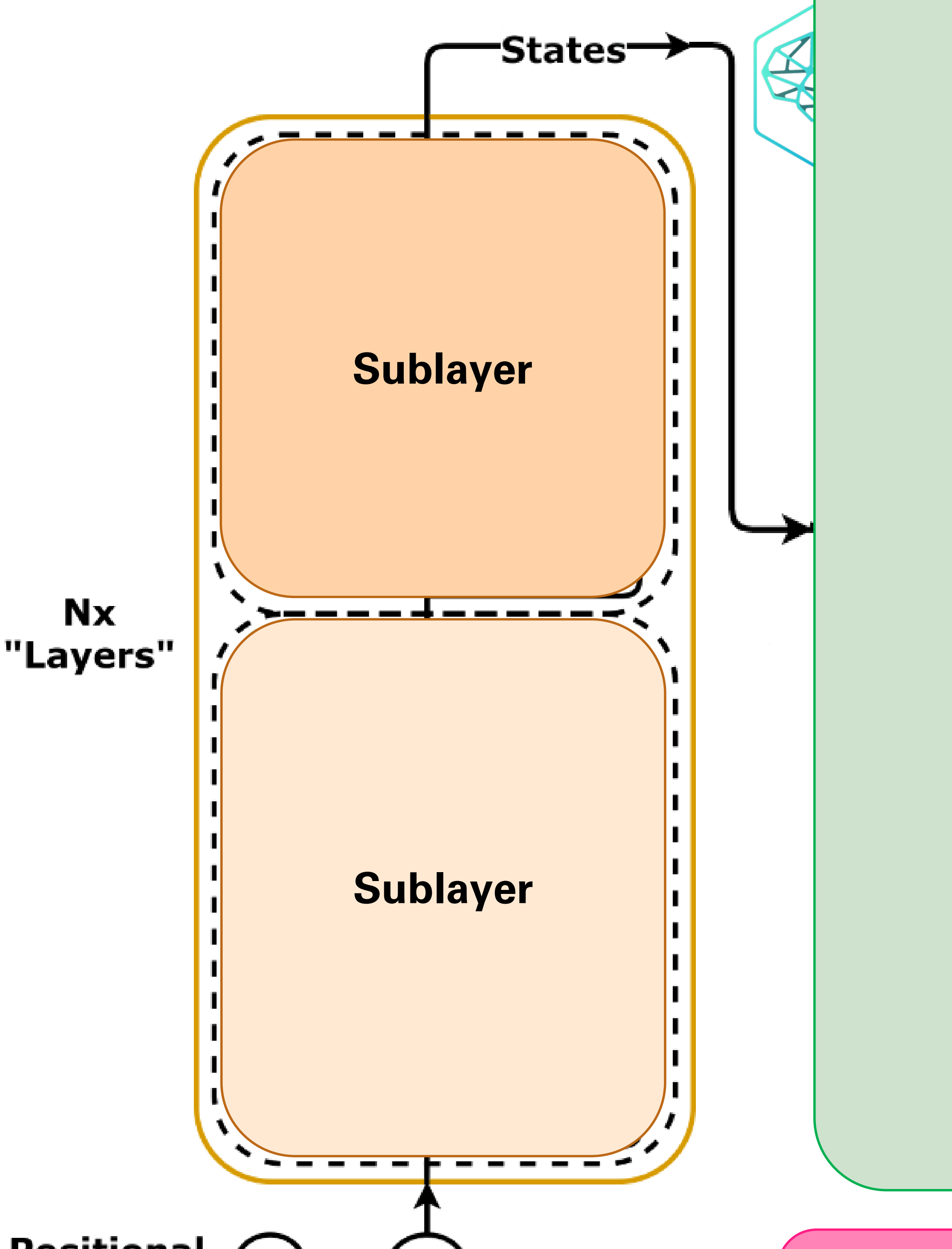


Architecture

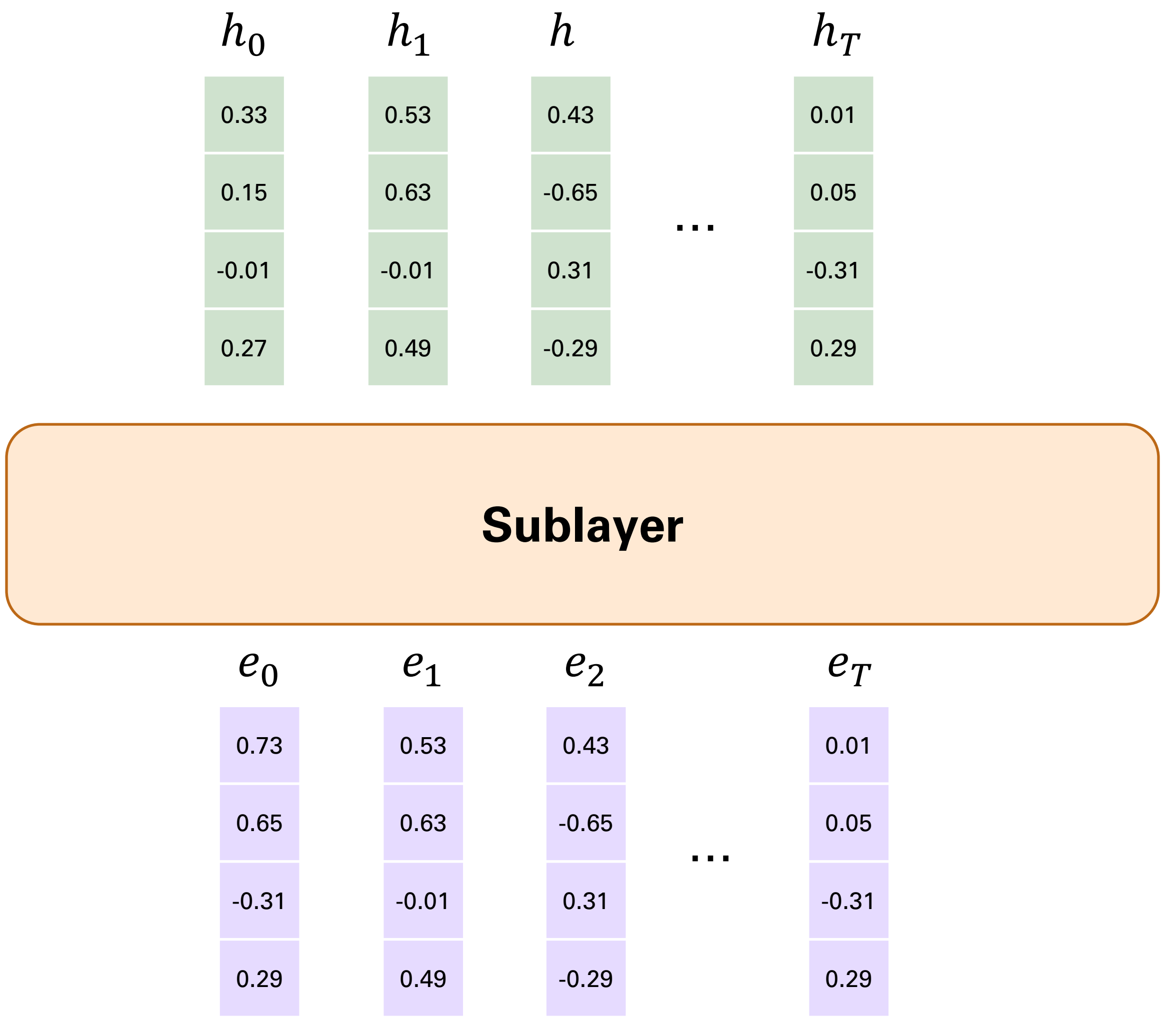


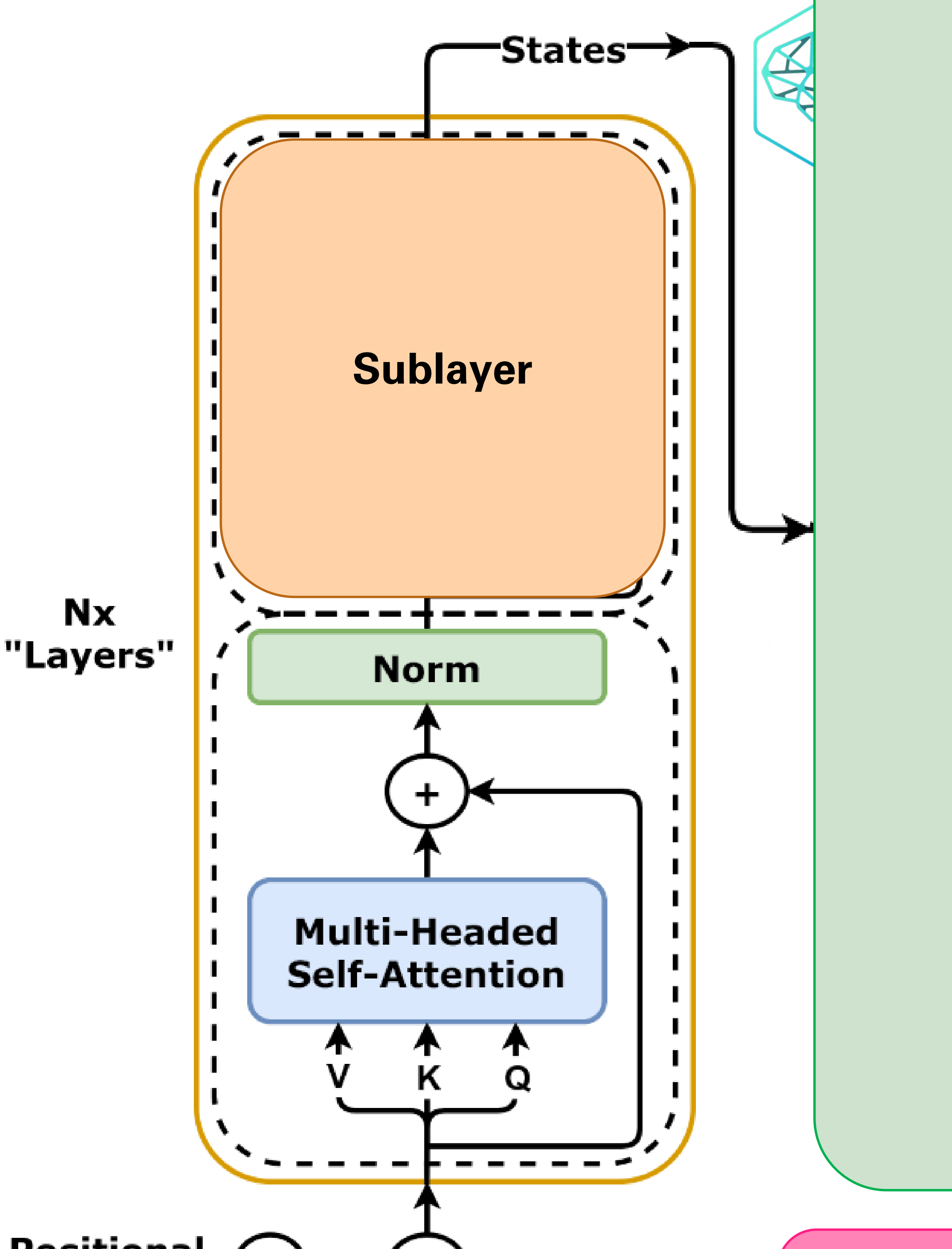
Architecture



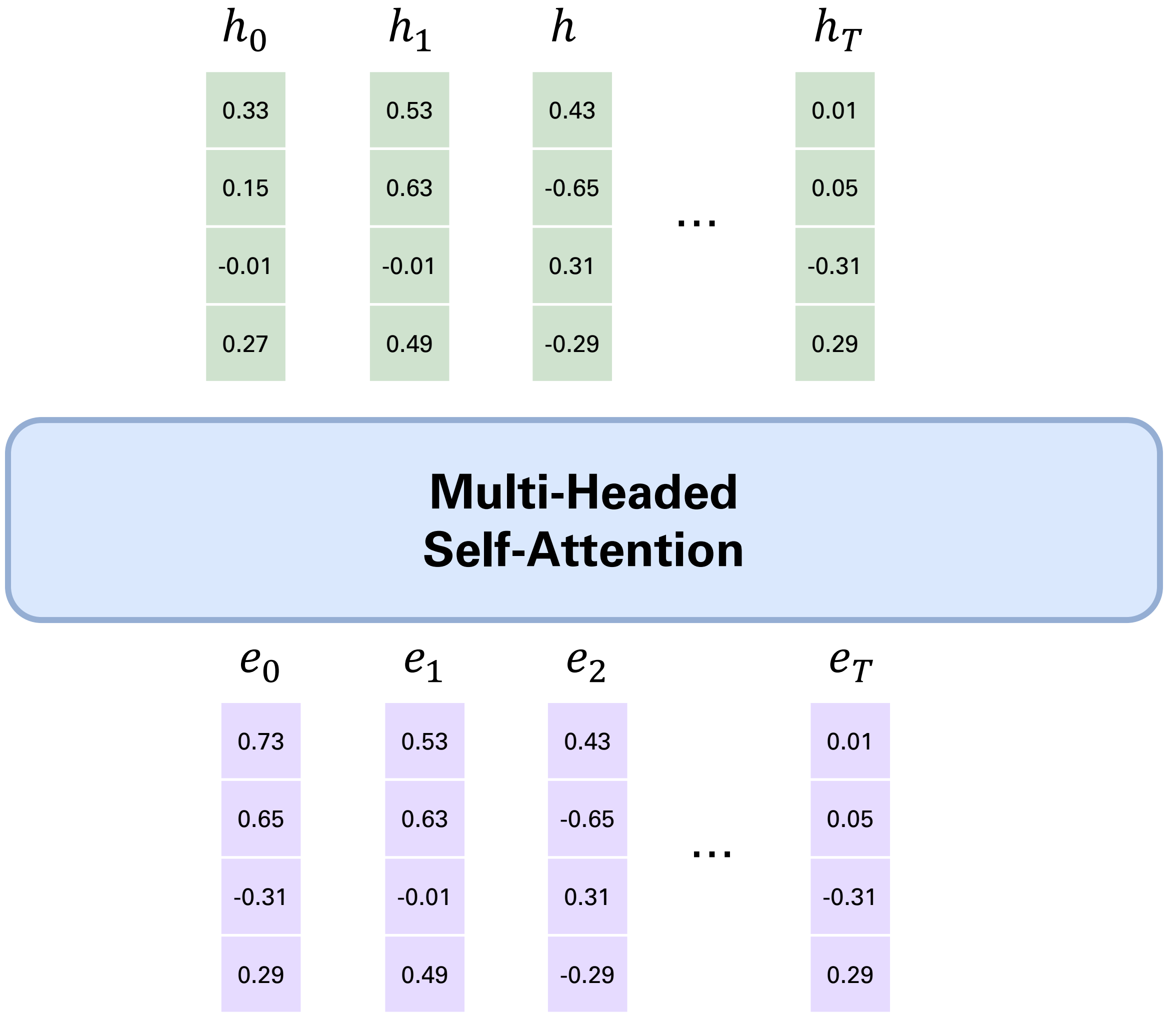


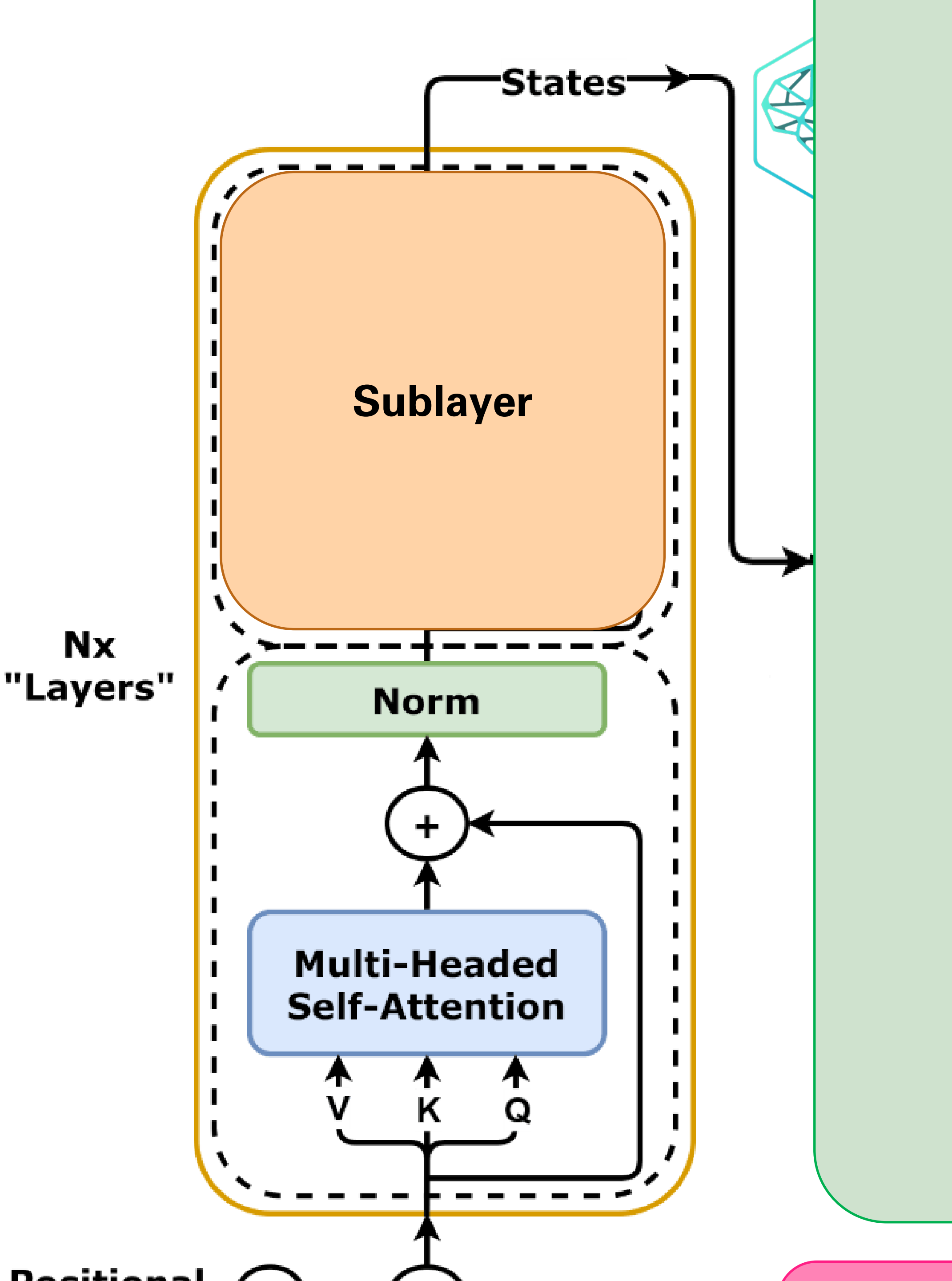
Architecture





Architecture





Self-attention

He went to the bank and learned of his empty account,
after which he went to the river bank and cried.

Self-attention

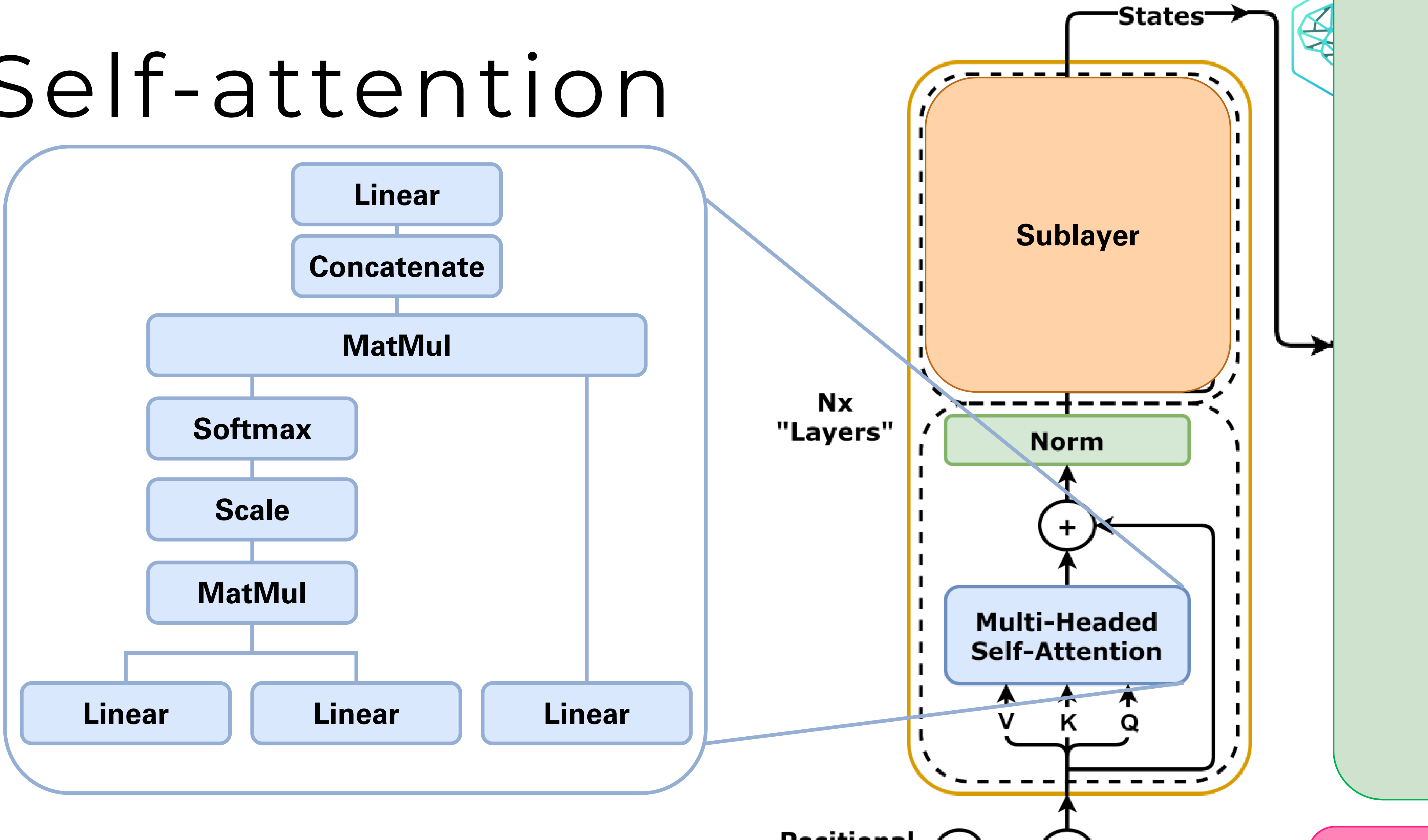
He went to the bank and learned of his empty account,
after which he went to the river bank and cried.

Self-attention

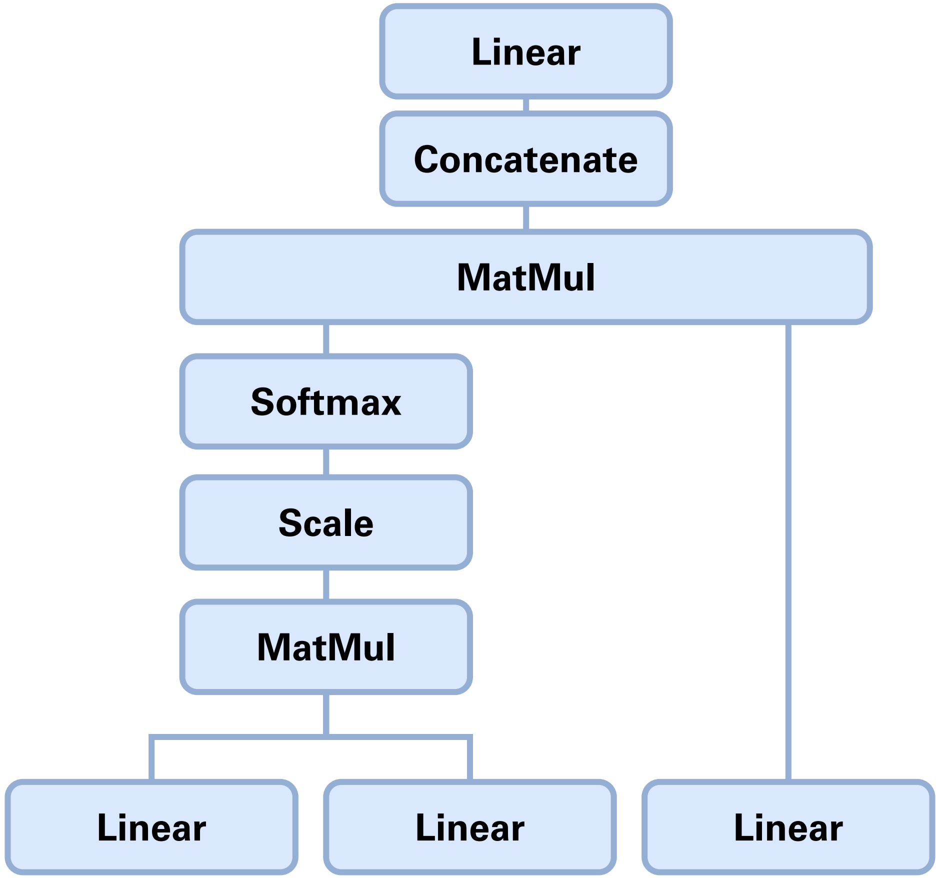
He went to the bank and learned of his empty account,
after which he went to the river bank and cried.

The meaning of every word can be
regarded as the sum of the words it
pays the most attention to

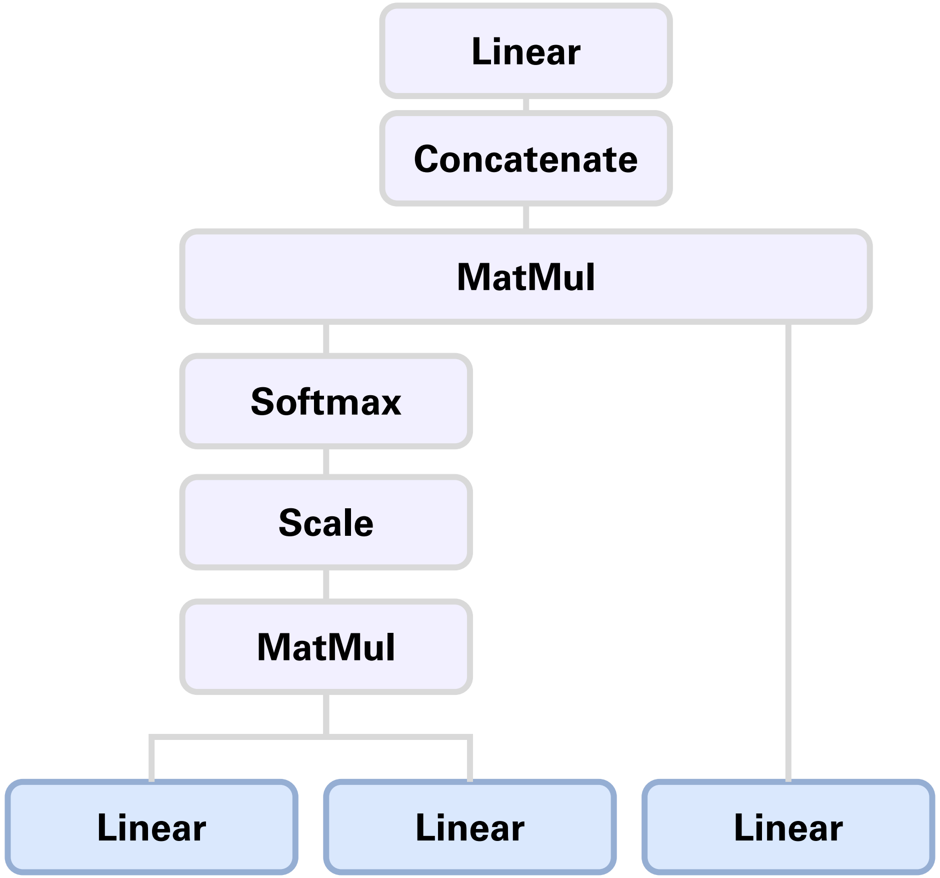
Self-attention



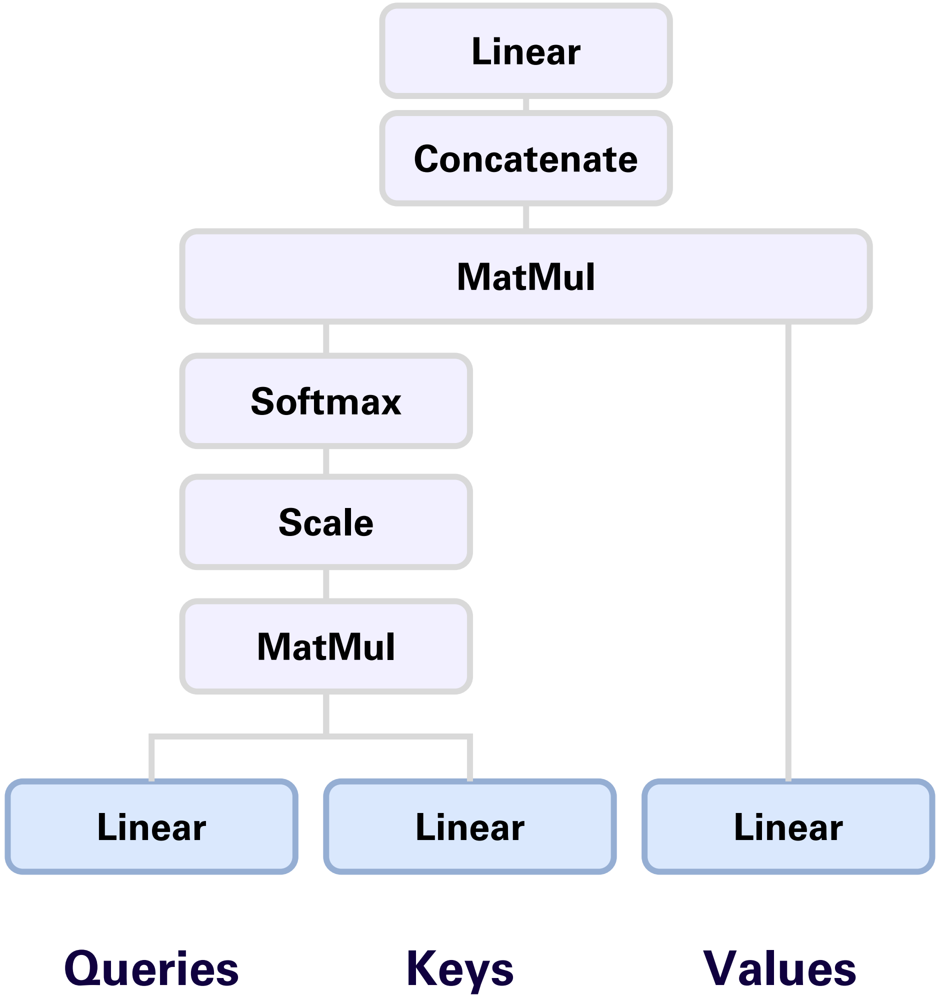
Self-attention



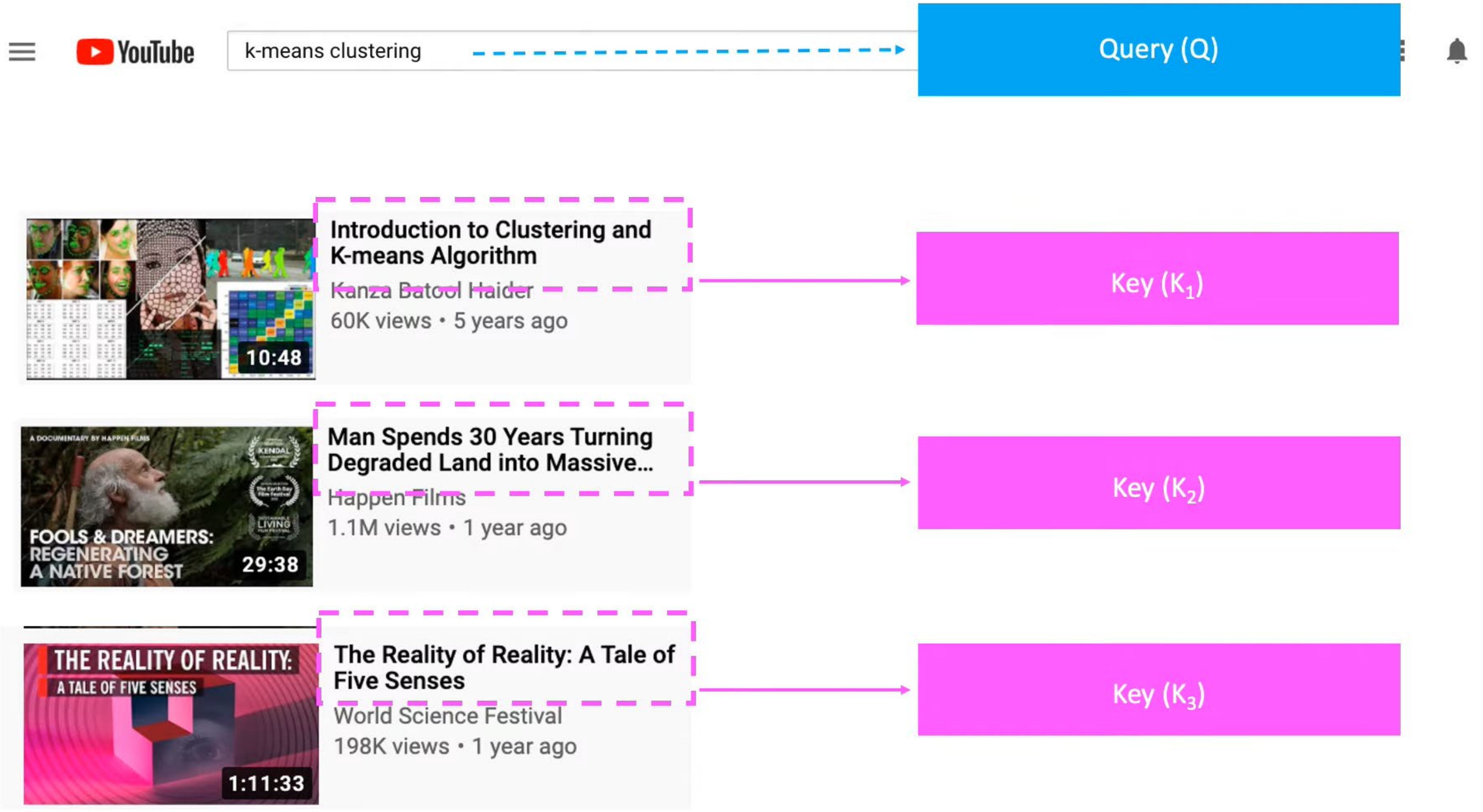
Self-attention

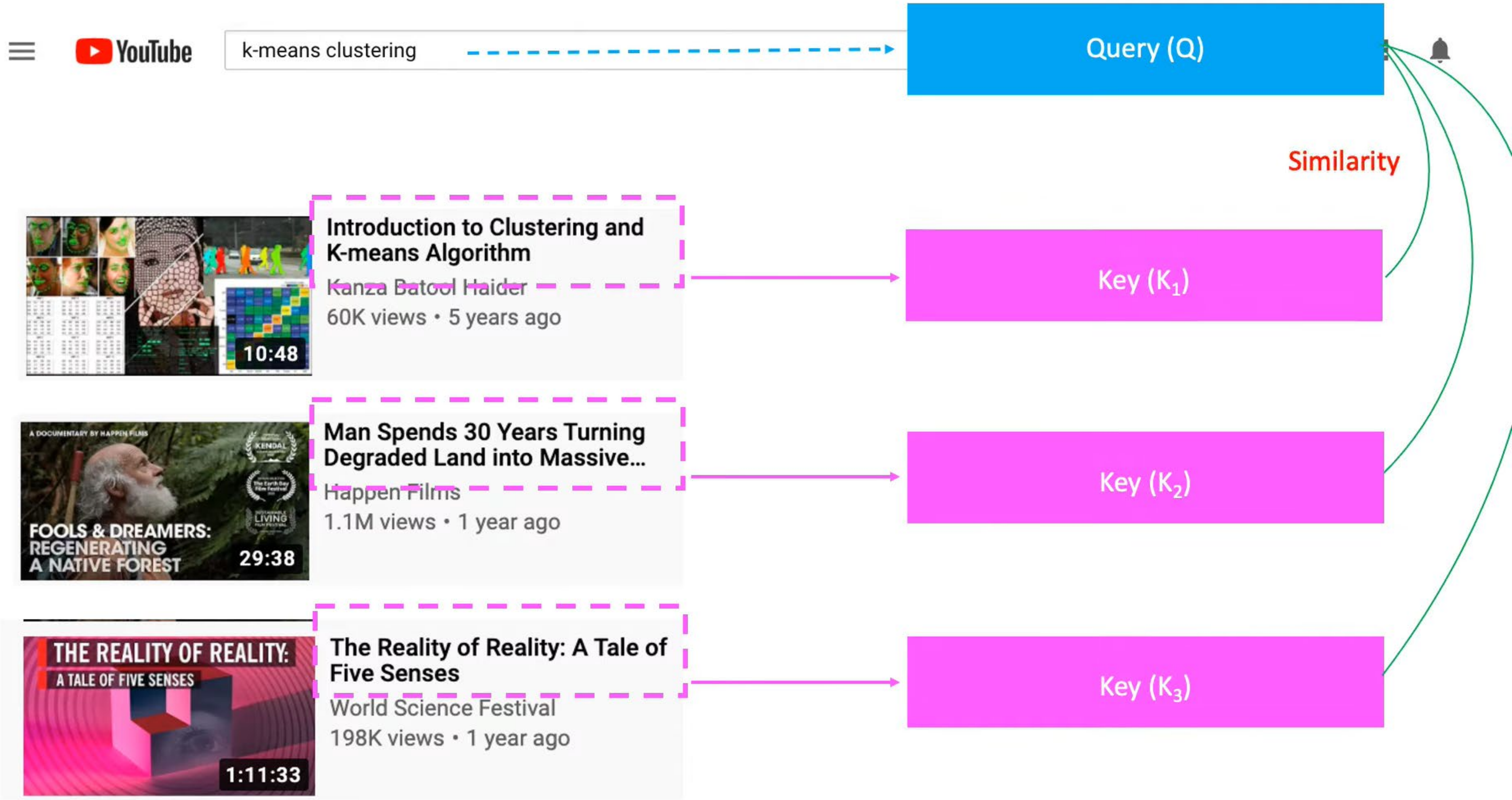


Self-attention

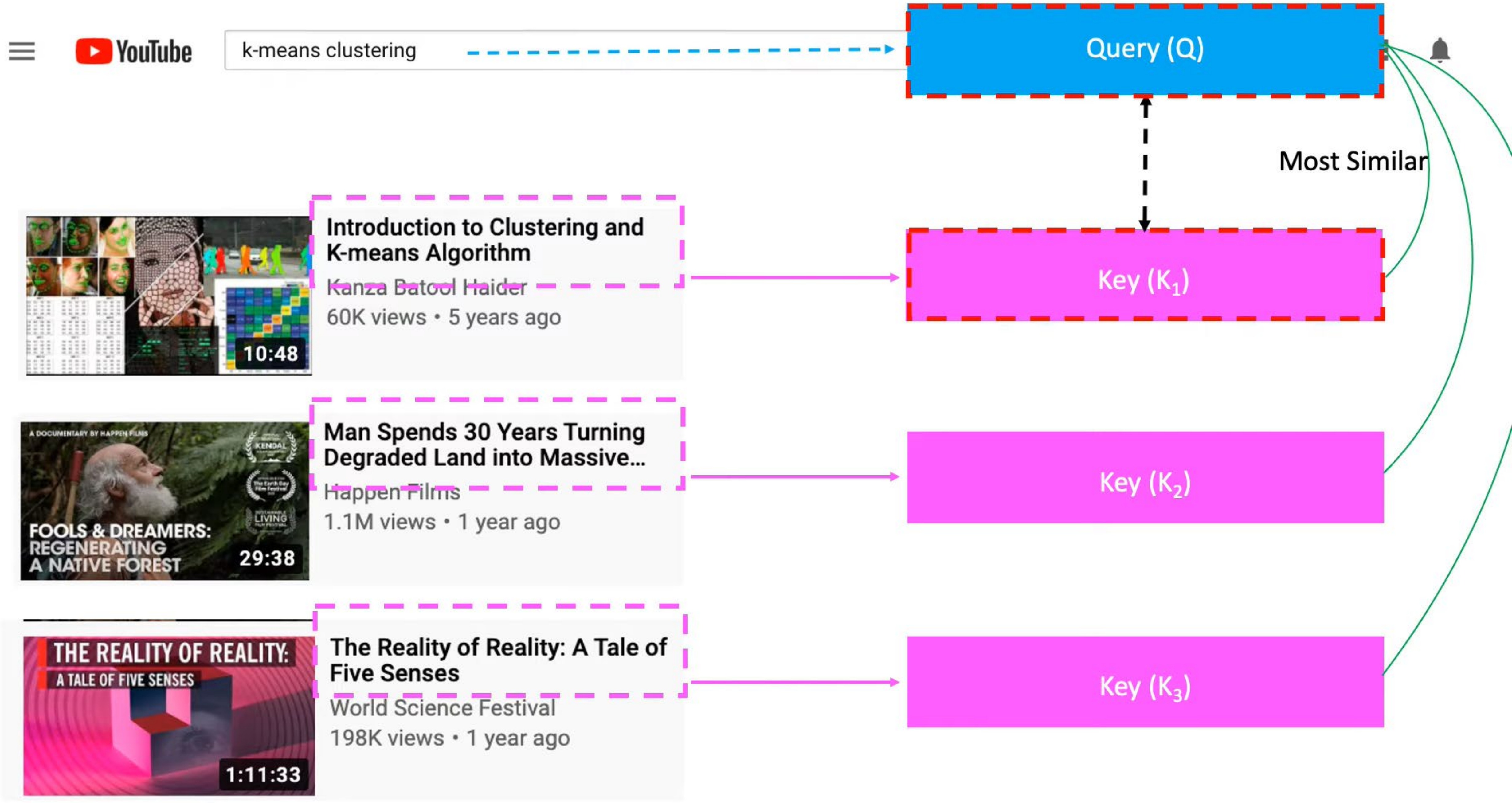



YouTube
k-means clustering
Query (Q)

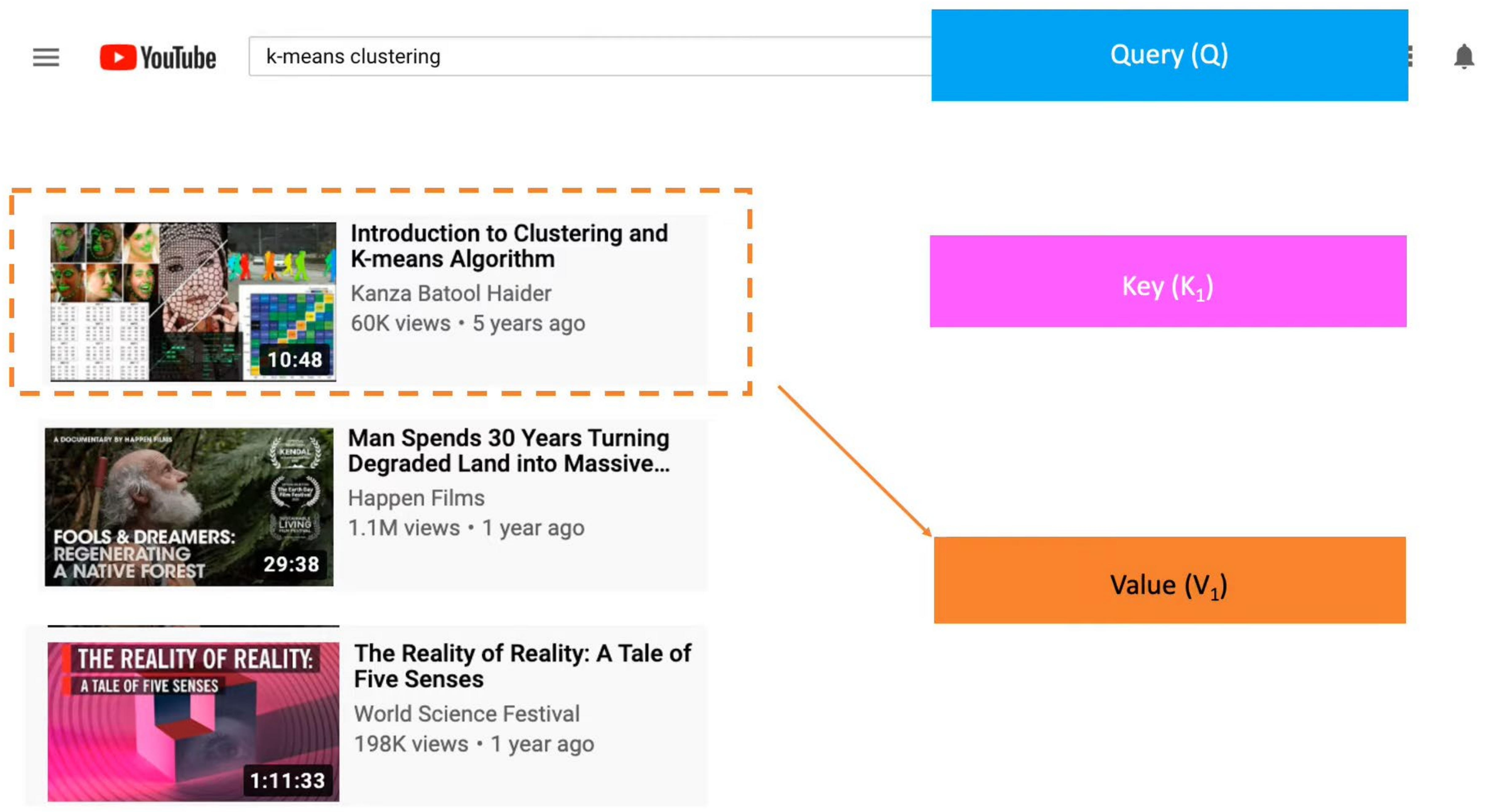

YouTube
k-means clustering
Query (Q)
Introduction to Clustering and K-means Algorithm
Kanza Batool Haider
60K views • 5 years ago
10:48
Key (K_1)
A DOCUMENTARY BY HAPPEN FILMS
KENDAL
LIVING
FOOLS & DREAMERS: REGENERATING A NATIVE FOREST
29:38
Man Spends 30 Years Turning Degraded Land into Massive...
Happen Films
1.1M views • 1 year ago
Key (K_2)
THE REALITY OF REALITY: A TALE OF FIVE SENSES
1:11:33
The Reality of Reality: A Tale of Five Senses
World Science Festival
198K views • 1 year ago
Key (K_3)



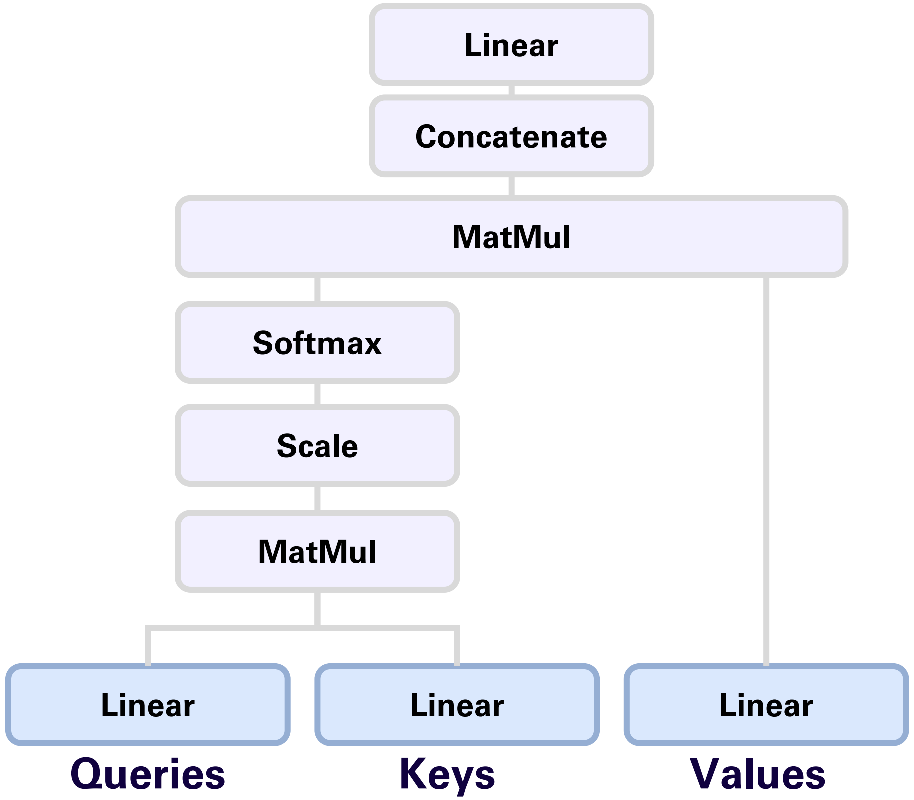
YouTube
k-means clustering
Query (Q)
Similarity
Introduction to Clustering and K-means Algorithm
Kanza Batool Haider
60K views • 5 years ago
10:48
Key (K_1)
Man Spends 30 Years Turning Degraded Land into Massive...
Happen Films
1.1M views • 1 year ago
FOOLS & DREAMERS: REGENERATING A NATIVE FOREST
29:38
Key (K_2)
THE REALITY OF REALITY: A TALE OF FIVE SENSES
The Reality of Reality: A Tale of Five Senses
World Science Festival
198K views • 1 year ago
1:11:33
Key (K_3)

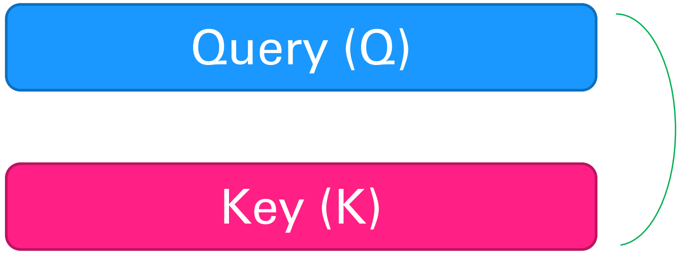


YouTube
k-means clustering
Query (Q)
Most Similar
Introduction to Clustering and K-means Algorithm
Kanza Batool Haider
60K views • 5 years ago
10:48
Key (K_1)
A DOCUMENTARY BY HAPPEN FILMS
FOOLS & DREAMERS: REGENERATING A NATIVE FOREST
29:38
Man Spends 30 Years Turning Degraded Land into Massive...
Happen Films
1.1M views • 1 year ago
Key (K_2)
THE REALITY OF REALITY: A TALE OF FIVE SENSES
1:11:33
The Reality of Reality: A Tale of Five Senses
World Science Festival
198K views • 1 year ago
Key (K_3)


YouTube
k-means clustering
Query (Q)
Key (K_1)
Value (V_1)
Introduction to Clustering and K-means Algorithm
Kanza Batool Haider
60K views • 5 years ago
10:48
Man Spends 30 Years Turning Degraded Land into Massive...
Happen Films
1.1M views • 1 year ago
A DOCUMENTARY BY HAPPEN FILMS
FOOLS & DREAMERS: REGENERATING A NATIVE FOREST
29:38
The Reality of Reality: A Tale of Five Senses
World Science Festival
198K views • 1 year ago
THE REALITY OF REALITY: A TALE OF FIVE SENSES
1:11:33

Self-attention

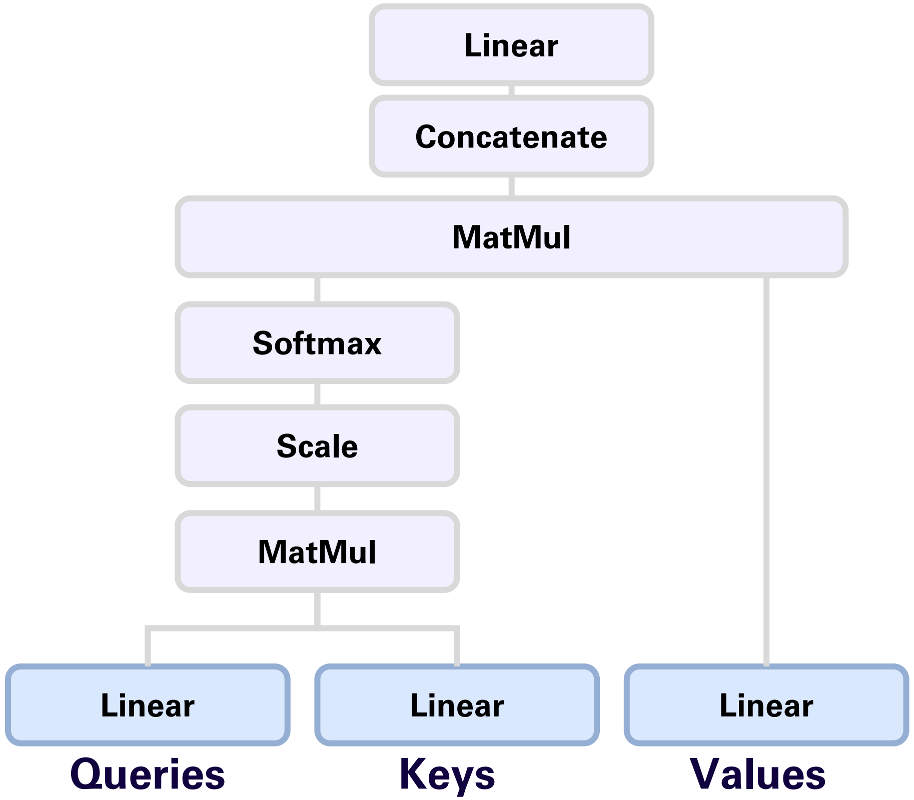


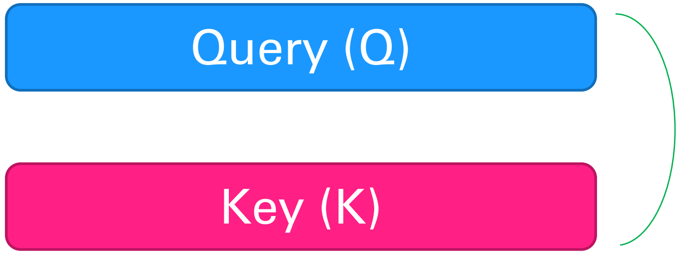


How to compute similarity?

$$\cos(a, b) = \frac{a \cdot b}{|a||b|}$$

Self-attention

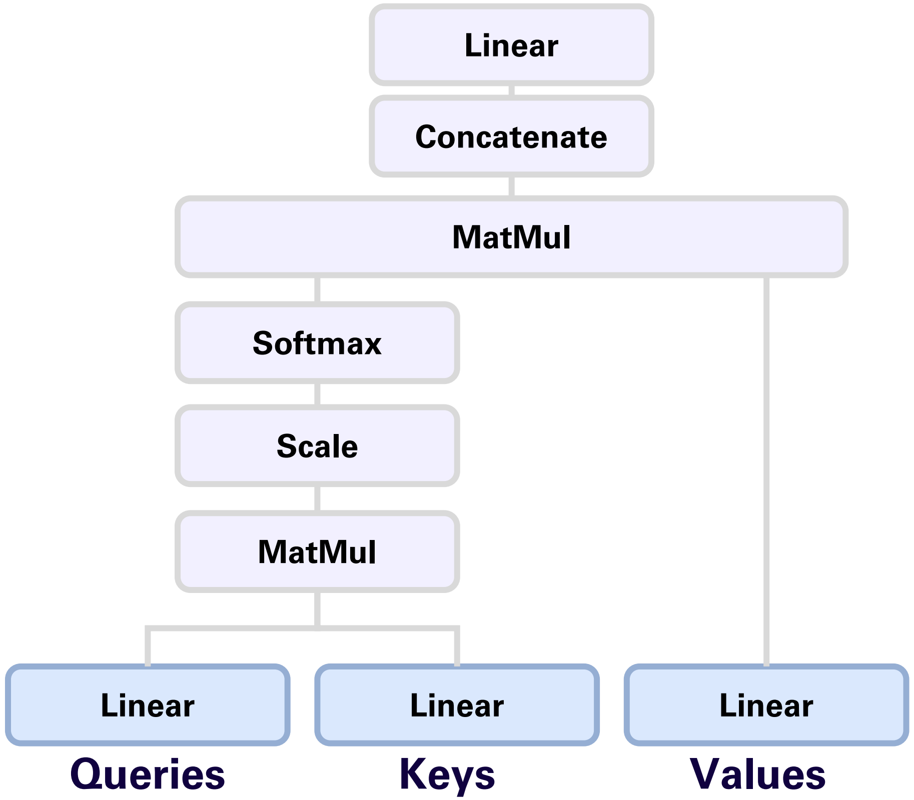




How to compute similarity?

$$\cos(A, B) = \frac{A \cdot B^T}{scaling}$$

Self-attention



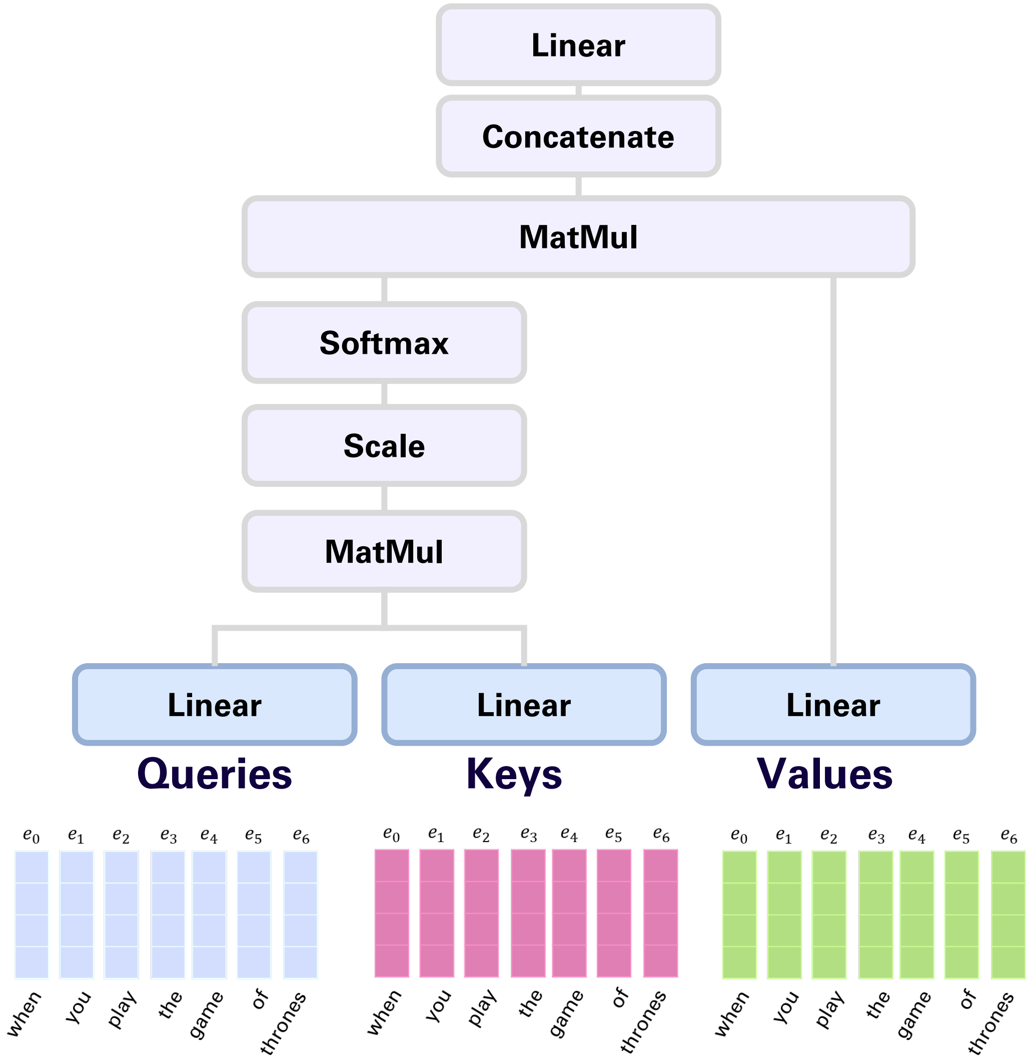
Query (Q)

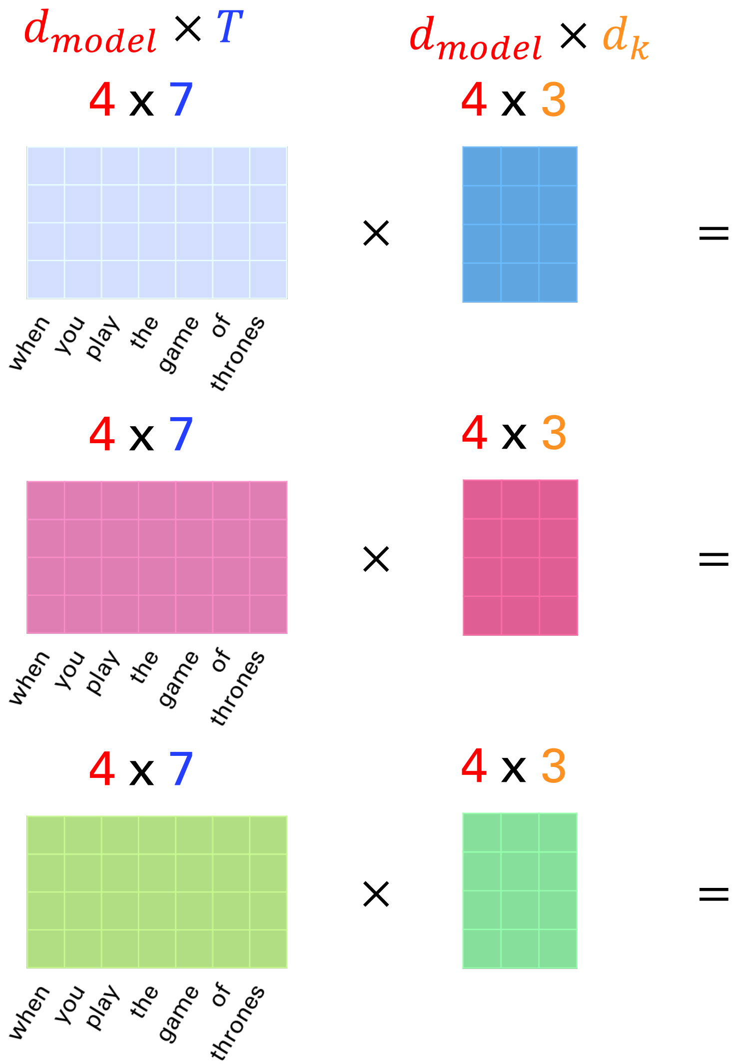
Key (K)

How to compute similarity?

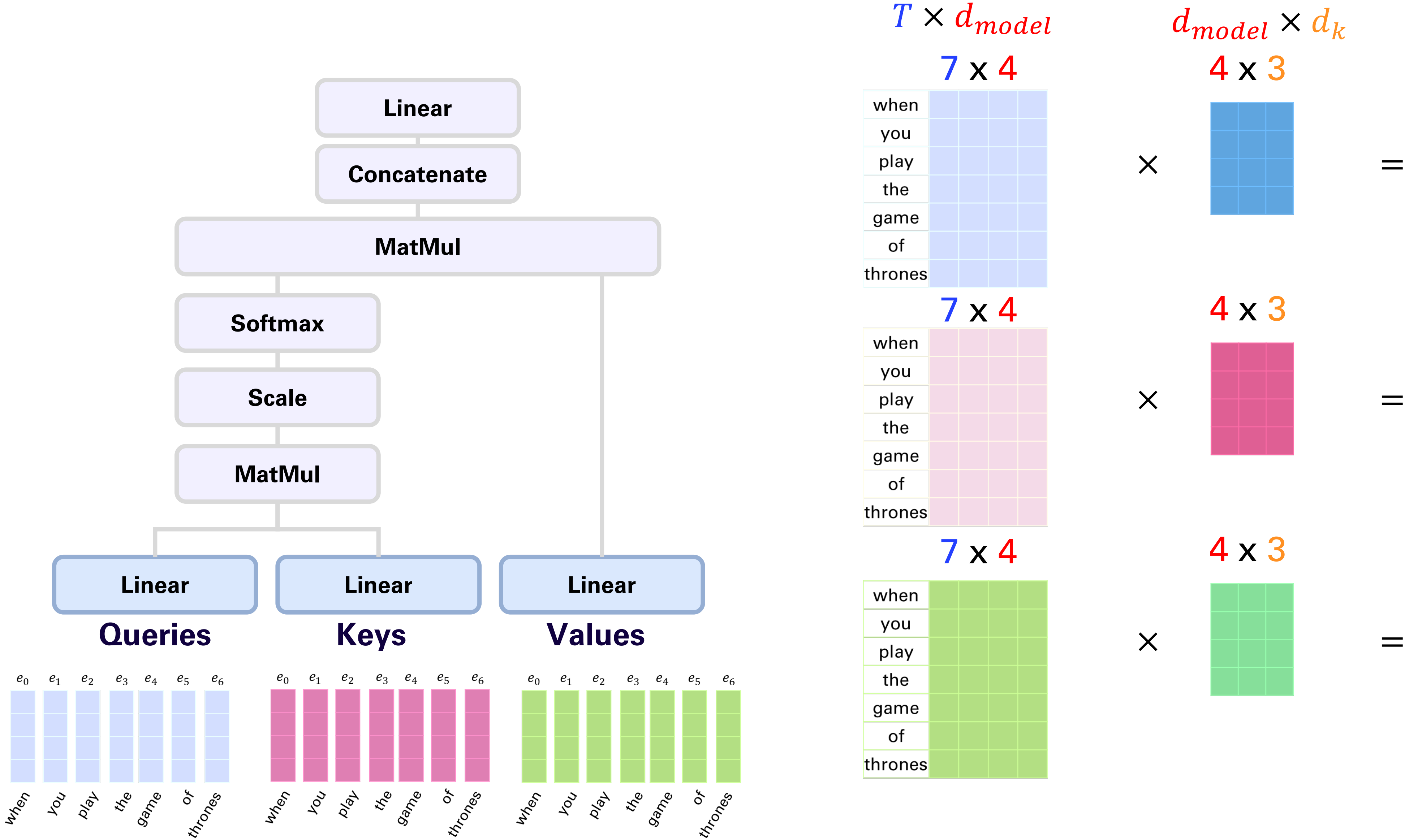
$$\cos(Q, K) = \frac{Q \cdot K^T}{scaling}$$

Self-attention

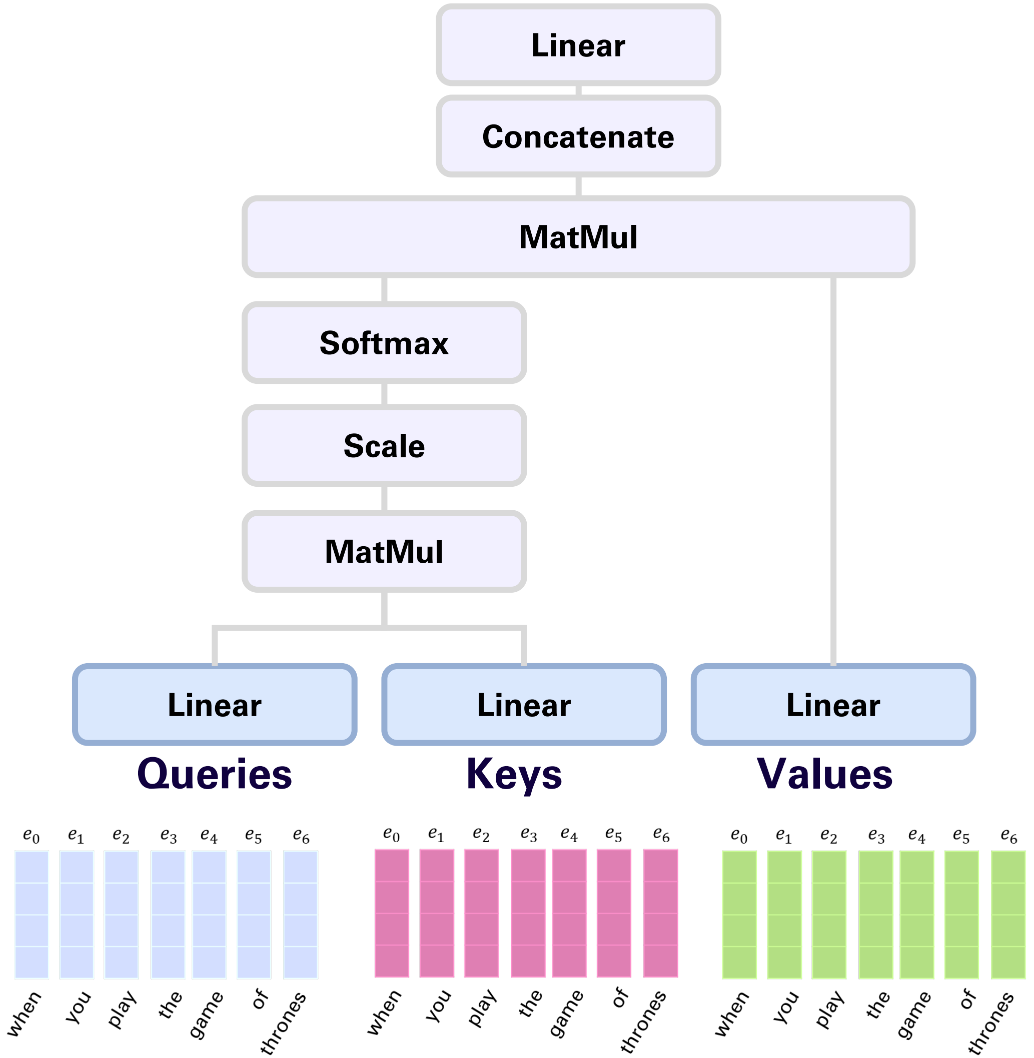


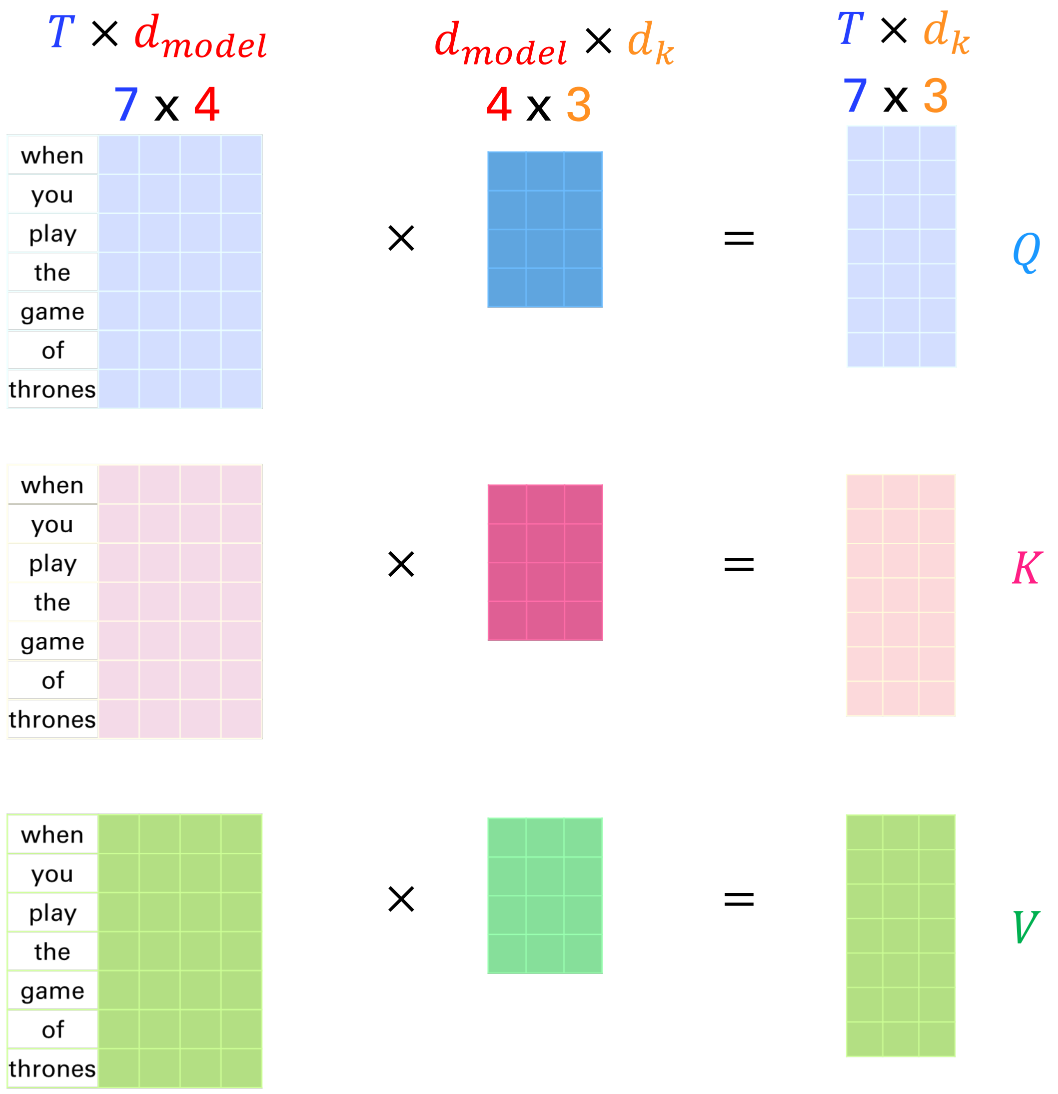


Self-attention

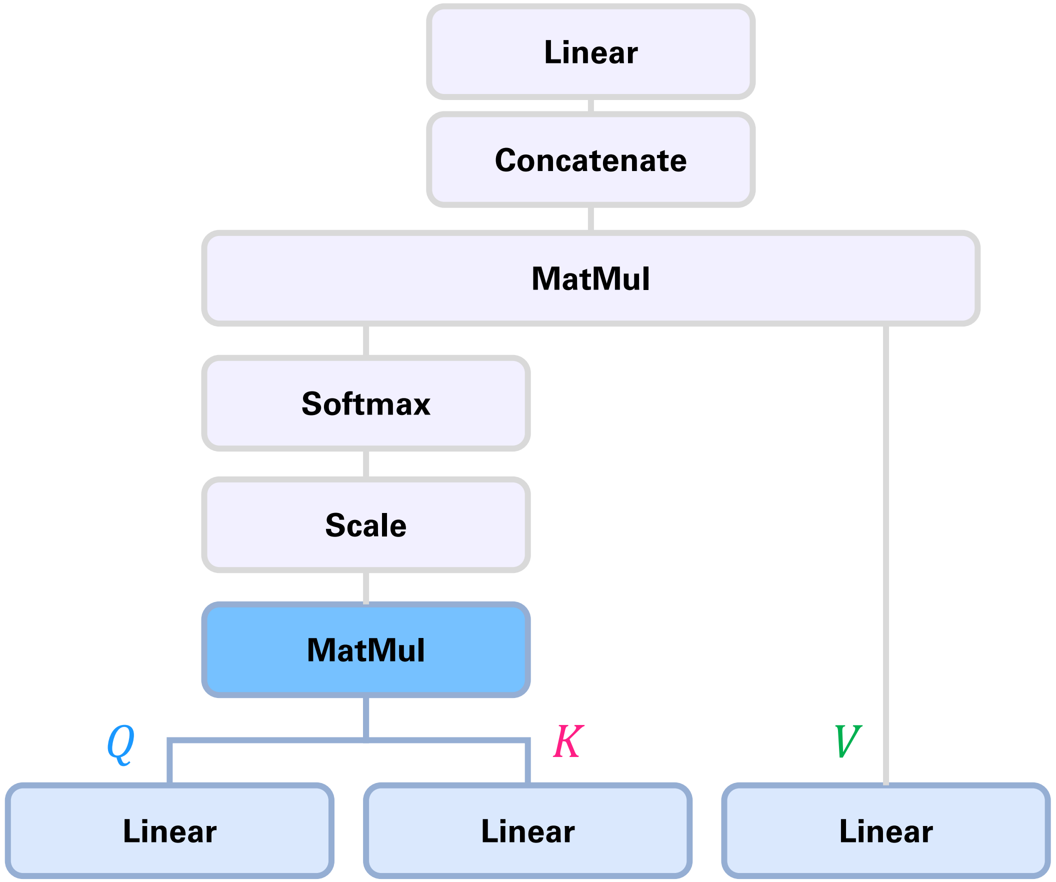


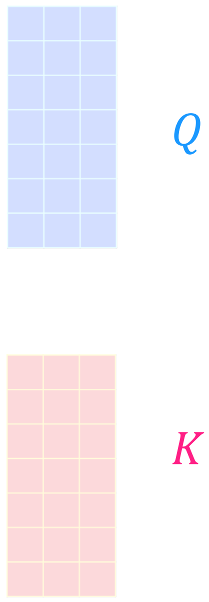
Self-attention



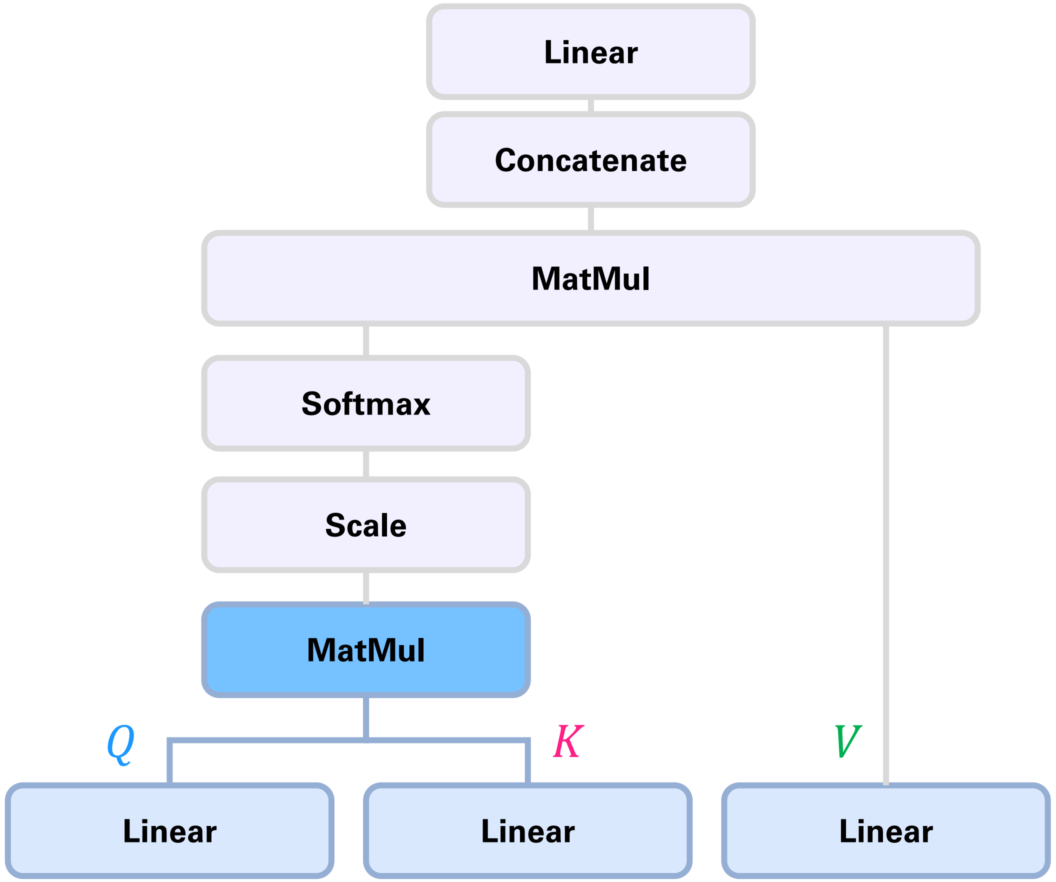


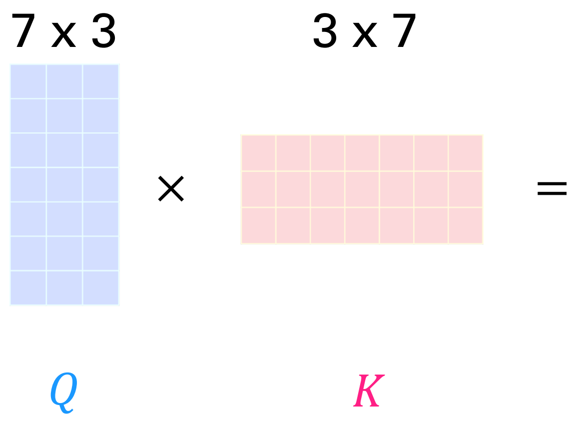
Self-attention





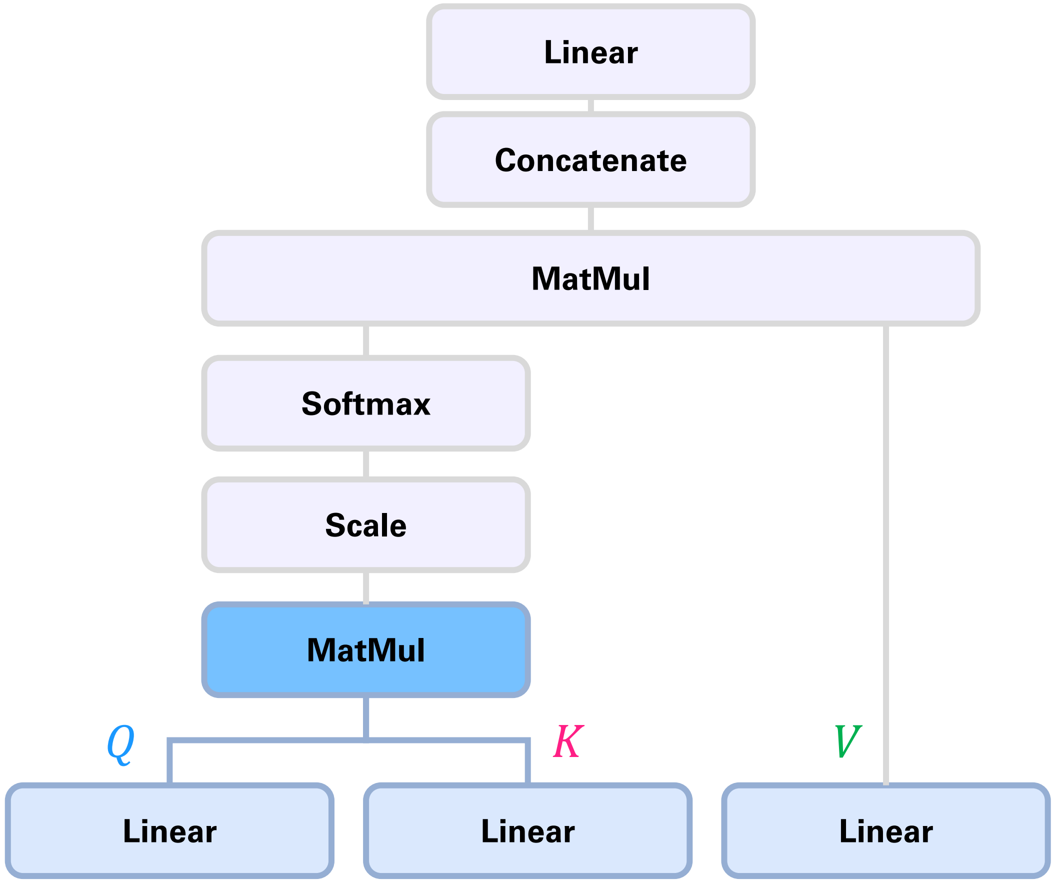
Self-attention

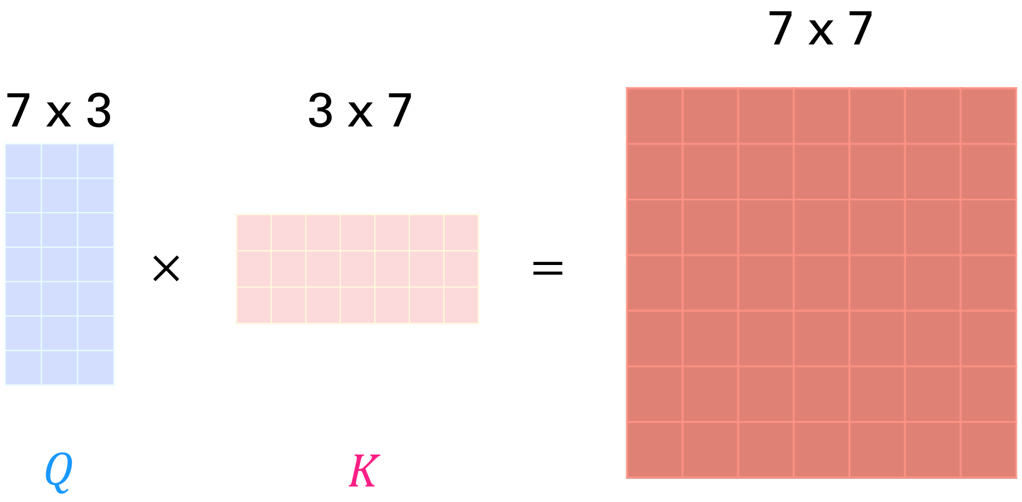




$$\frac{Q \cdot K^T}{scaling}$$

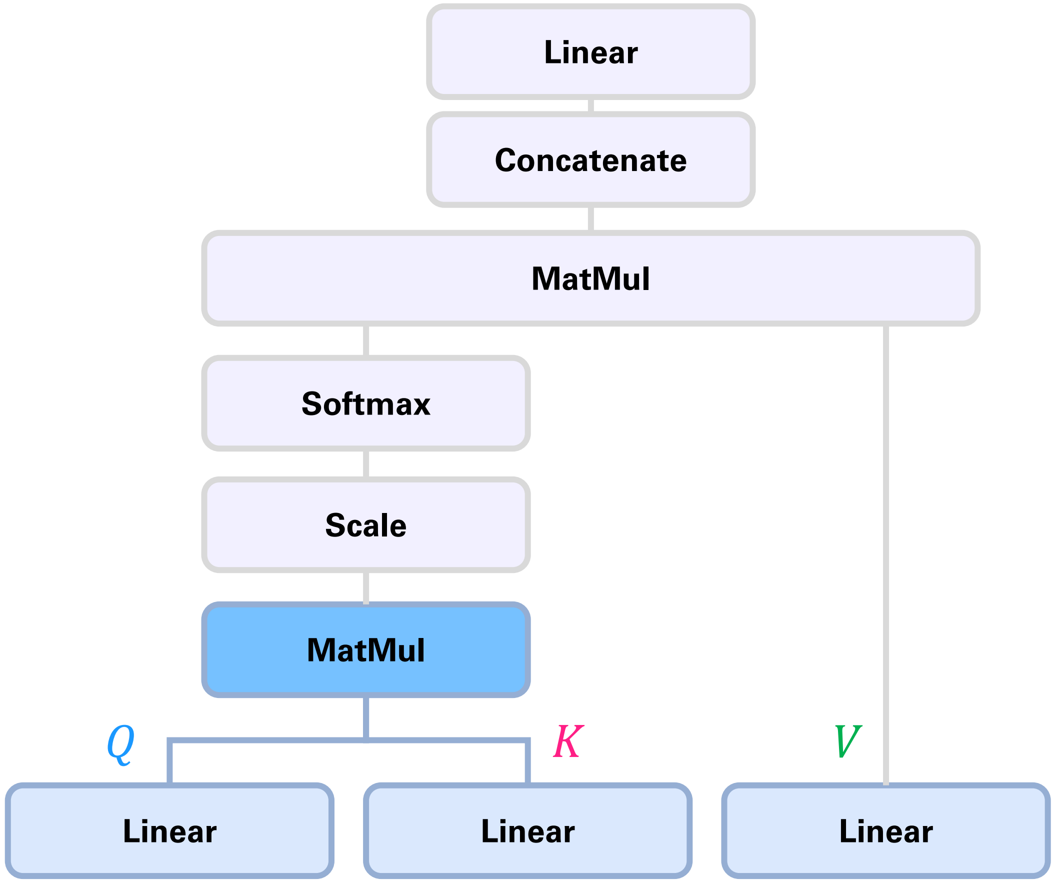
Self-attention





$$\frac{Q \cdot K^T}{scaling}$$

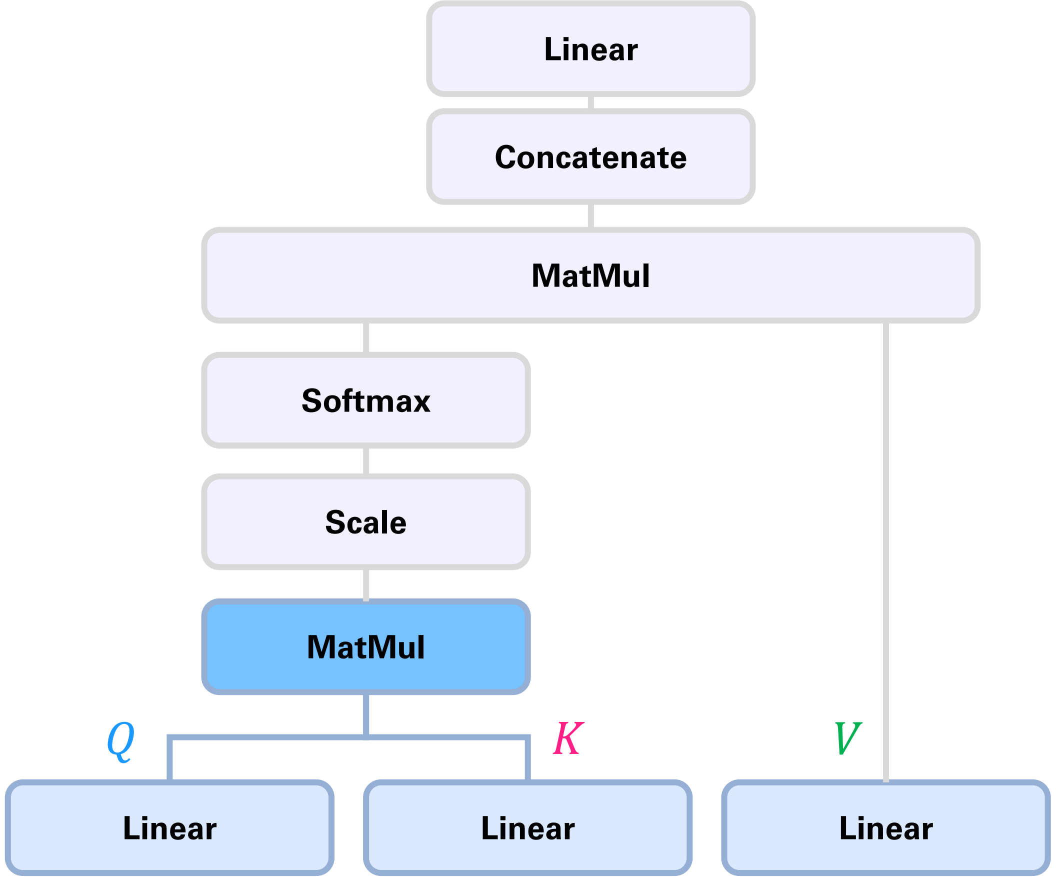
Self-attention



7 x 7

	When	you	play	the	game	of	thrones
When	89	20	41	10	55	78	59
you	90	98	81	22	87	15	32
play	29	81	95	10	90	30	92
the	10	22	67	12	88	40	89
game	22	70	90	56	98	44	80
of	10	15	30	40	44	44	59
thrones	59	72	92	90	13	59	99

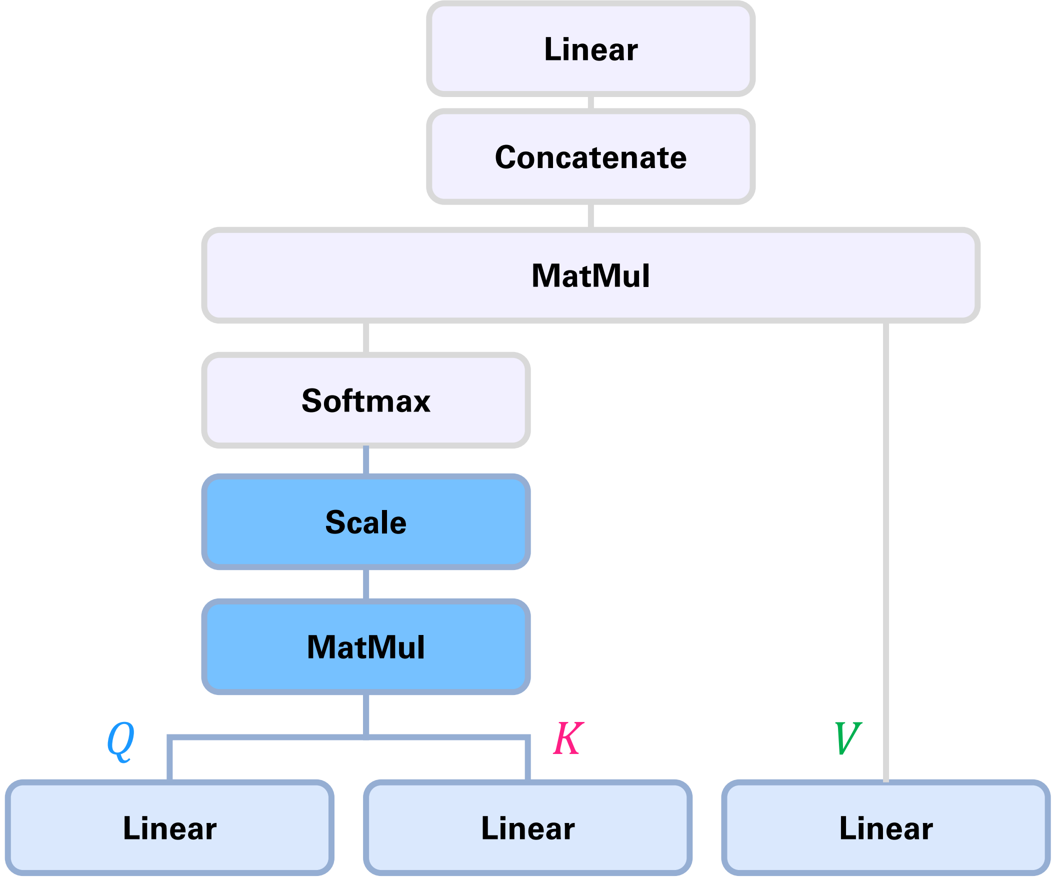
Self-attention



7 x 7

	When	you	play	the	game	of	thrones
When	89	20	41	10	55	78	59
you	90	98	81	22	87	15	32
play	29	81	95	10	90	30	92
the	10	22	67	12	88	40	89
game	22	70	90	56	98	44	80
of	10	15	30	40	44	44	59
thrones	59	72	92	90	13	59	99

Self-attention

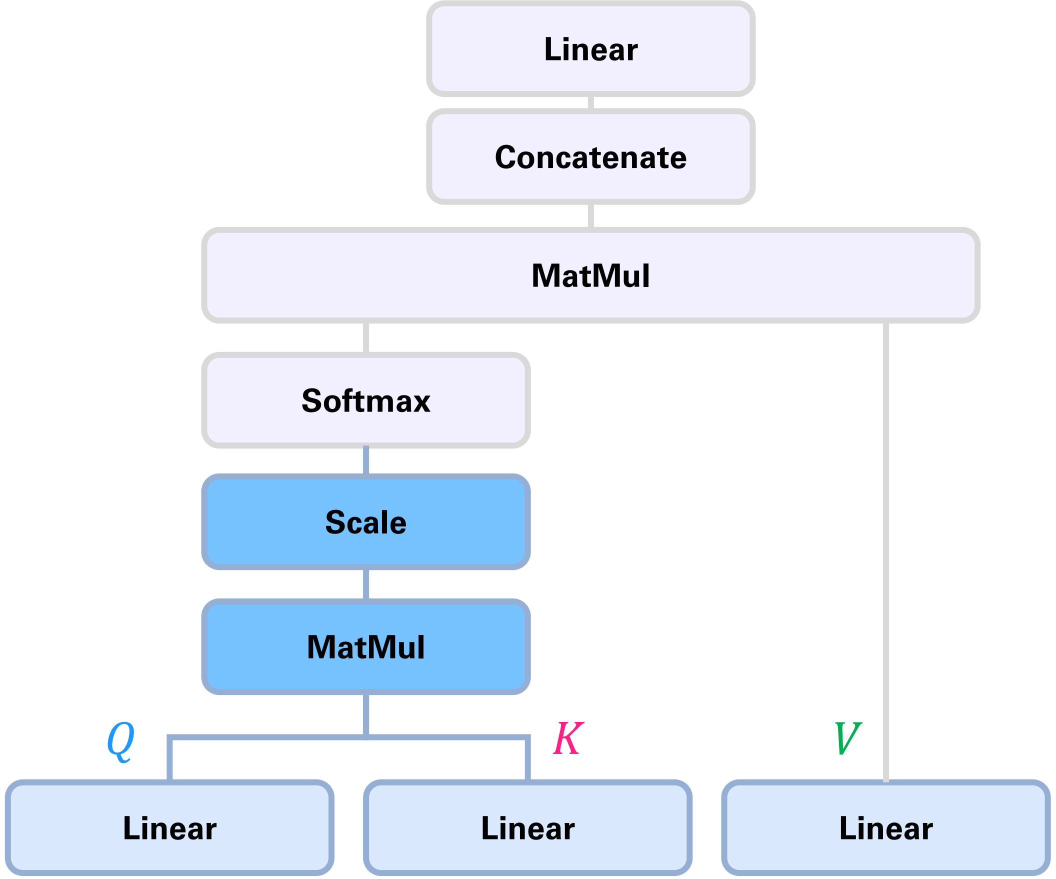


7 x 7

	When	you	play	the	game	of	thrones
When	89	20	41	10	55	78	59
you	90	98	81	22	87	15	32
play	29	81	95	10	90	30	92
the	10	22	67	12	88	40	89
game	22	70	90	56	98	44	80
of	10	15	30	40	44	44	59
thrones	59	72	92	90	13	59	99

$$\sqrt{d_k}$$

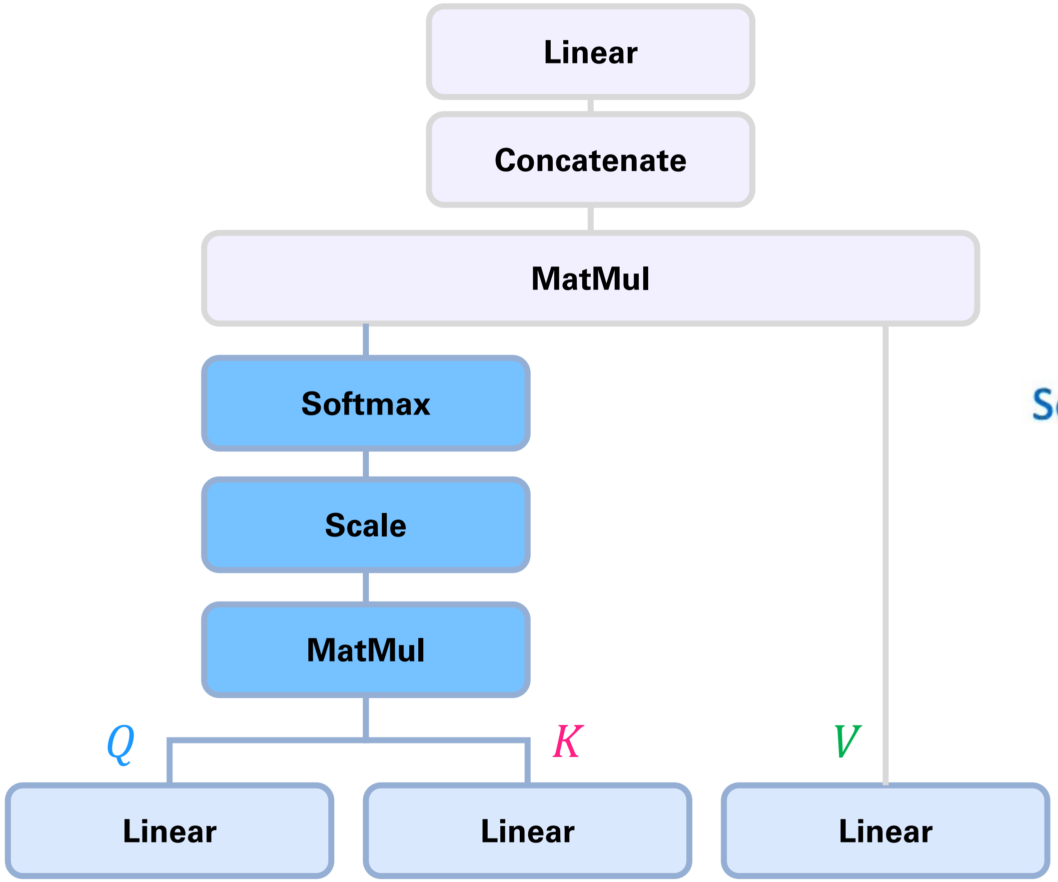
Self-attention



7 x 7

	When	you	play	the	game	of	thrones
When	33.6	7.6	15.5	3.8	20.8	3.8	22.3
you	7.6	34.0	30.6	8.3	26.5	5.7	27.2
play	15.5	30.6	35.9	3.8	34.0	11.3	34.8
the	3.8	8.3	3.8	34.8	33.3	15.1	33.6
game	20.8	26.5	34.0	33.3	37.0	16.6	35.9
of	3.8	5.7	11.3	15.1	16.6	32.1	22.3
thrones	22.3	27.2	34.8	34.0	35.9	22.3	37.4

Self-attention

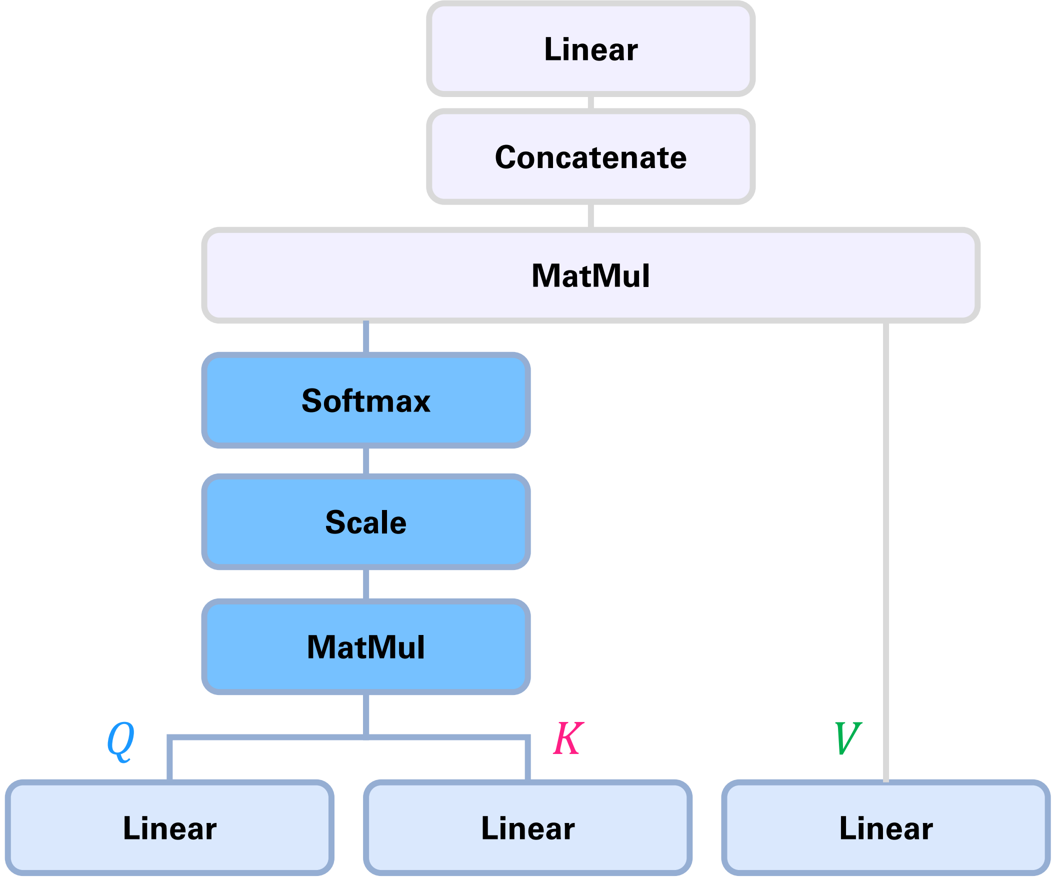


Softmax

7 x 7

	When	you	play	the	game	of	thrones
When	33.6	7.6	15.5	3.8	20.8	3.8	22.3
you	7.6	34.0	30.6	8.3	26.5	5.7	27.2
play	15.5	30.6	35.9	3.8	34.0	11.3	34.8
the	3.8	8.3	3.8	34.8	33.3	15.1	33.6
game	20.8	26.5	34.0	33.3	37.0	16.6	35.9
of	3.8	5.7	11.3	15.1	16.6	32.1	22.3
thrones	22.3	27.2	34.8	34.0	35.9	22.3	37.4

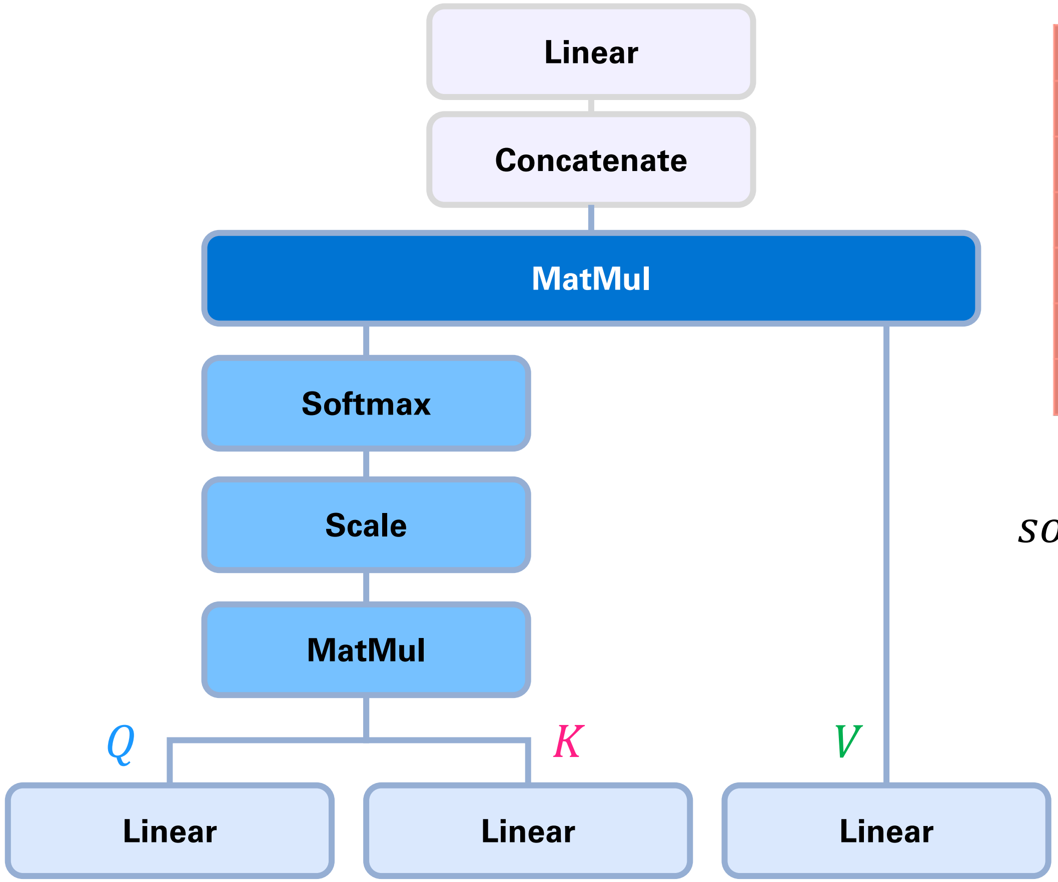
Self-attention

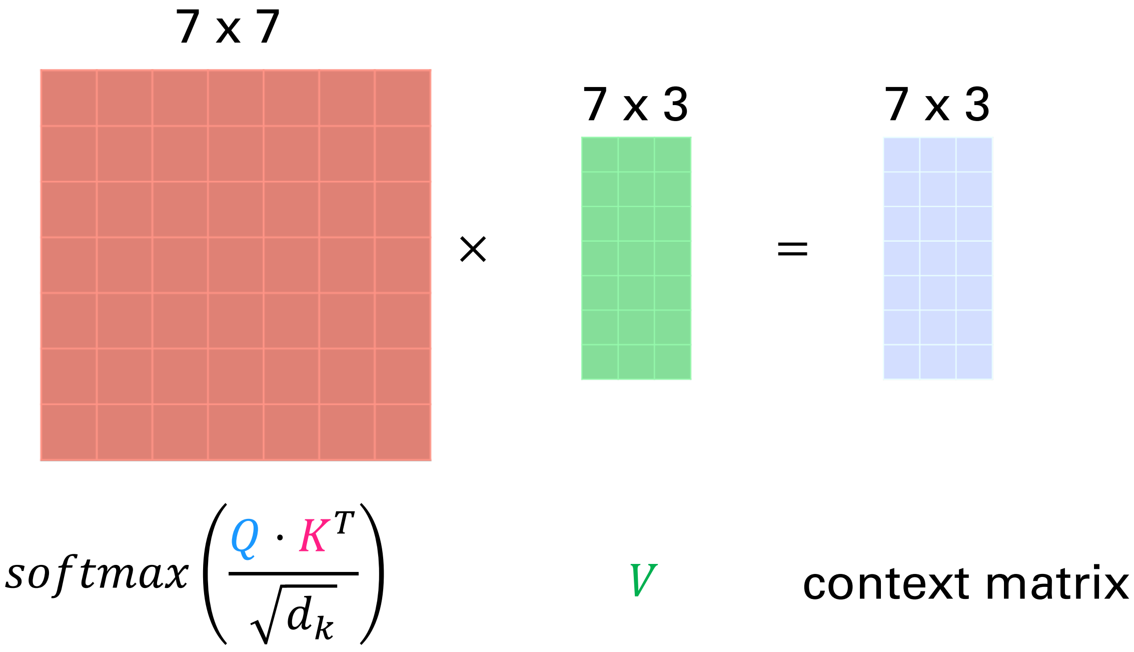


7 x 7

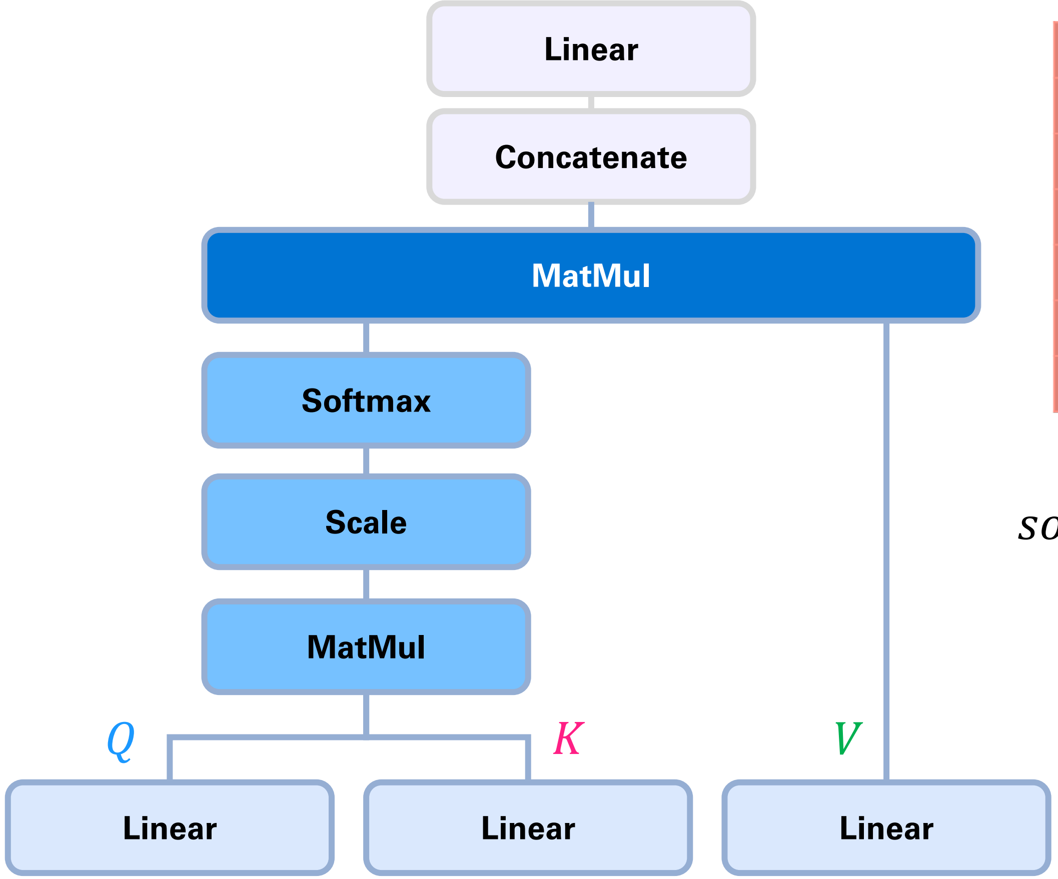
	When	you	play	the	game	of	thrones
When	1.00	0.00	0.00	0.00	0.00	0.00	0.00
you	0.00	0.97	0.03	0.00	0.00	0.00	0.00
play	0.00	0.00	0.68	0.00	0.10	0.00	0.22
the	0.00	0.00	0.00	0.65	0.14	0.00	0.21
game	0.00	0.00	0.03	0.02	0.72	0.00	0.23
of	0.00	0.00	0.00	0.00	0.00	1.00	0.00
thrones	0.00	0.00	0.05	0.03	0.17	0.00	0.75

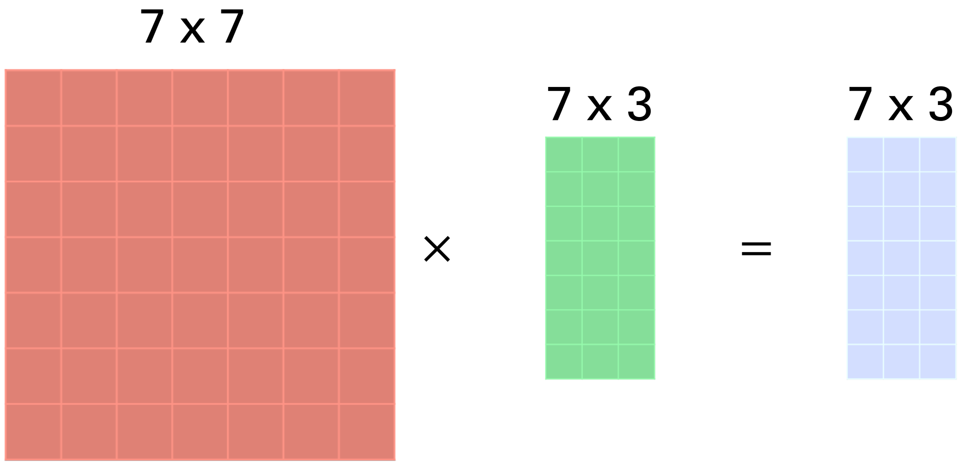
Self-attention





Self-attention

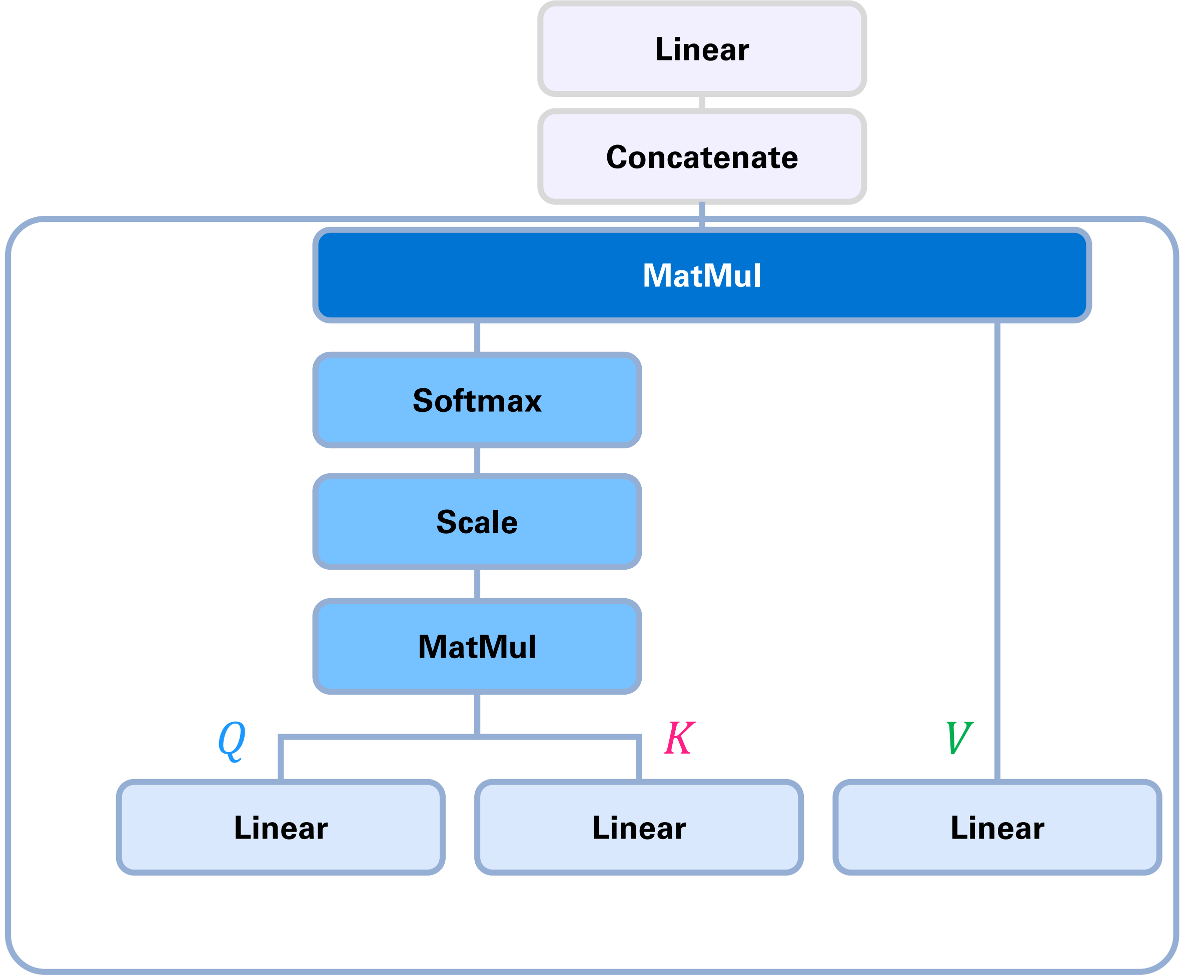




$$softmax\left(\frac{Q \cdot K^T}{\sqrt{d_k}}\right) \qquad V \qquad \text{context matrix}$$

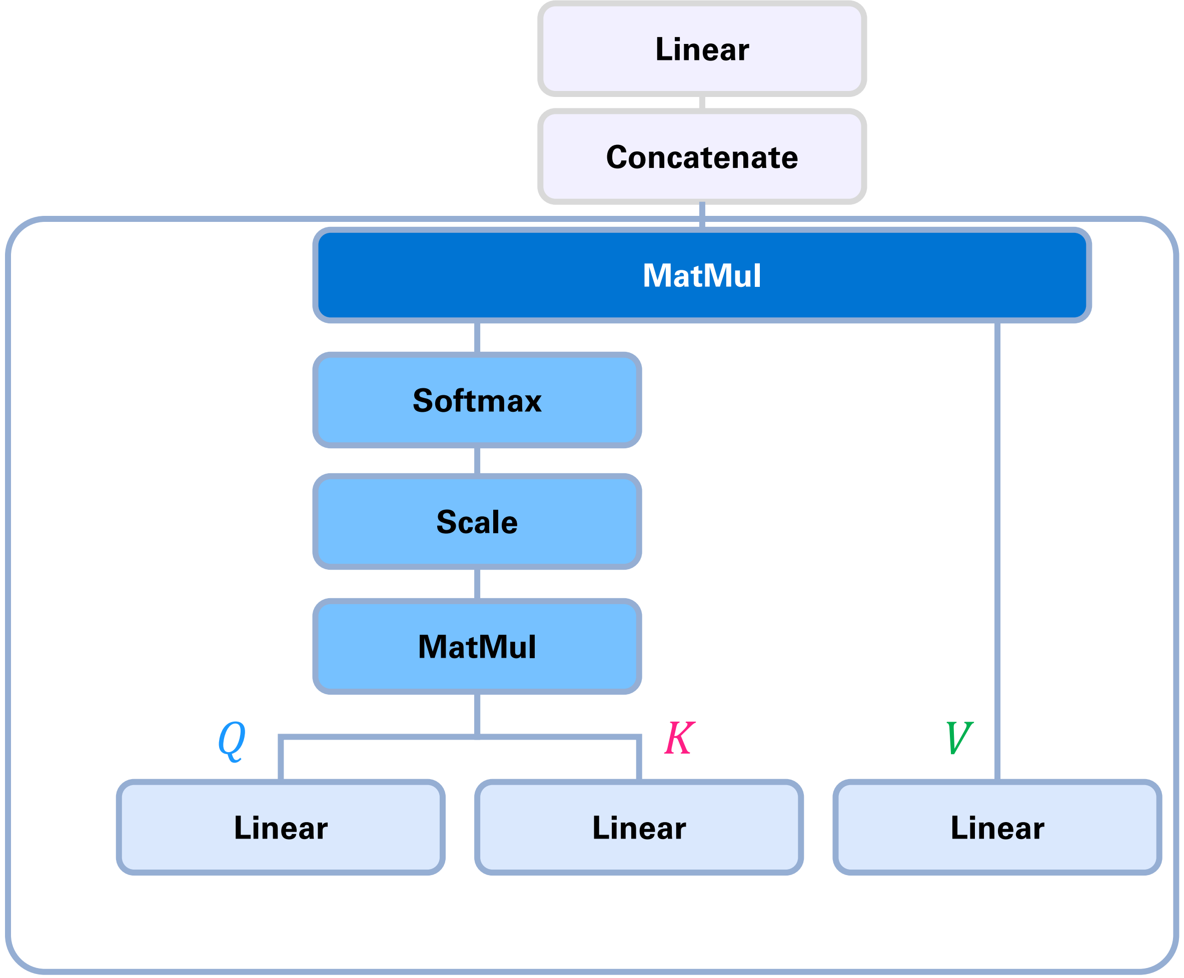
Attention weighting   Value   Output

Multi-headed Self-Attention

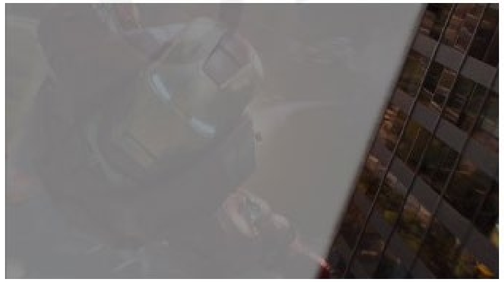


All this work is done by a single head...
... but we have multiple heads

Multi-headed Self-Attention



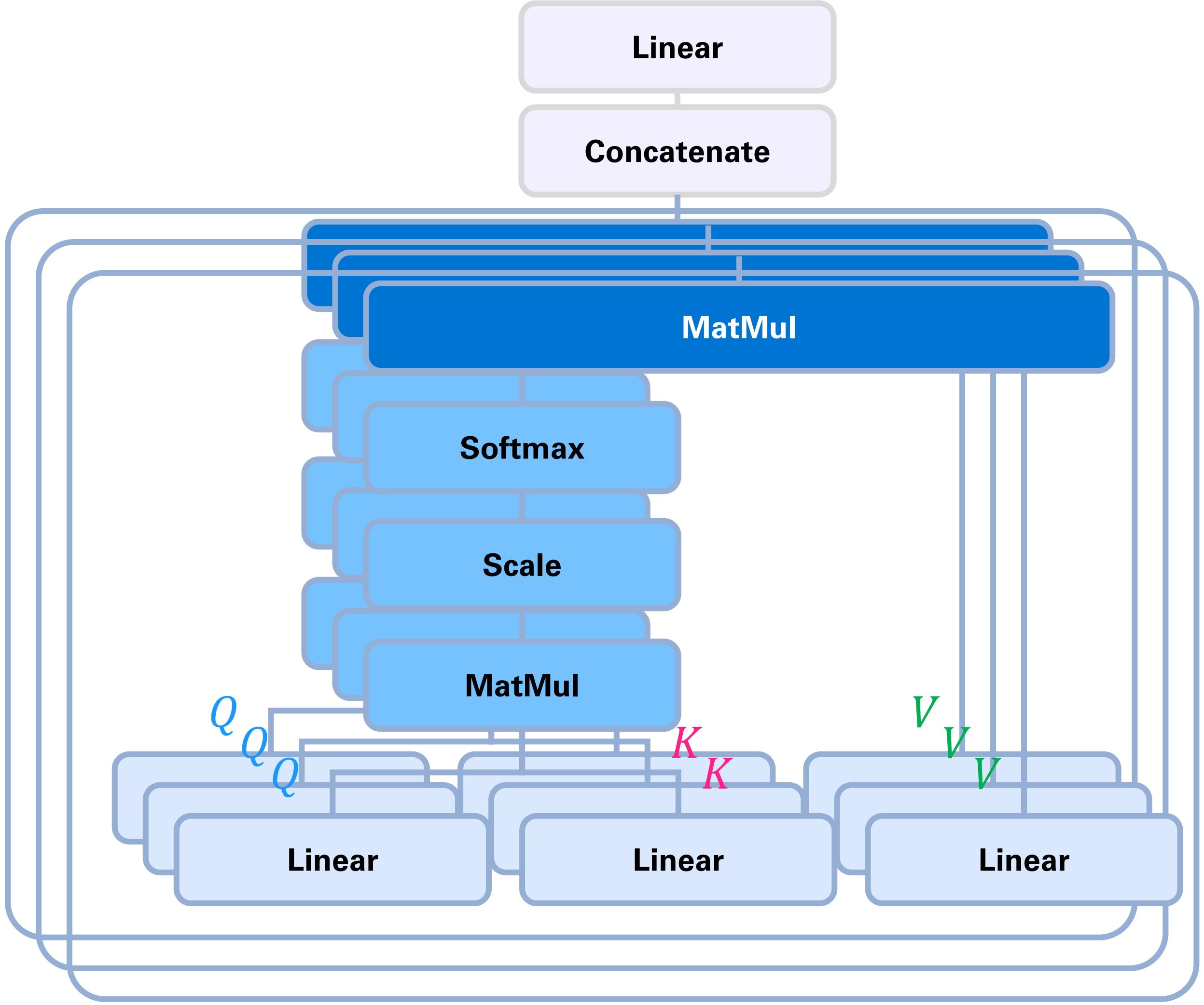
Output of attention head 1

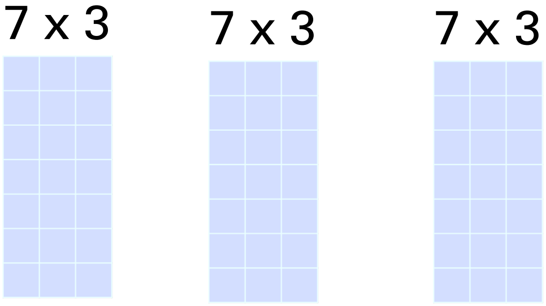


Output of attention head 2

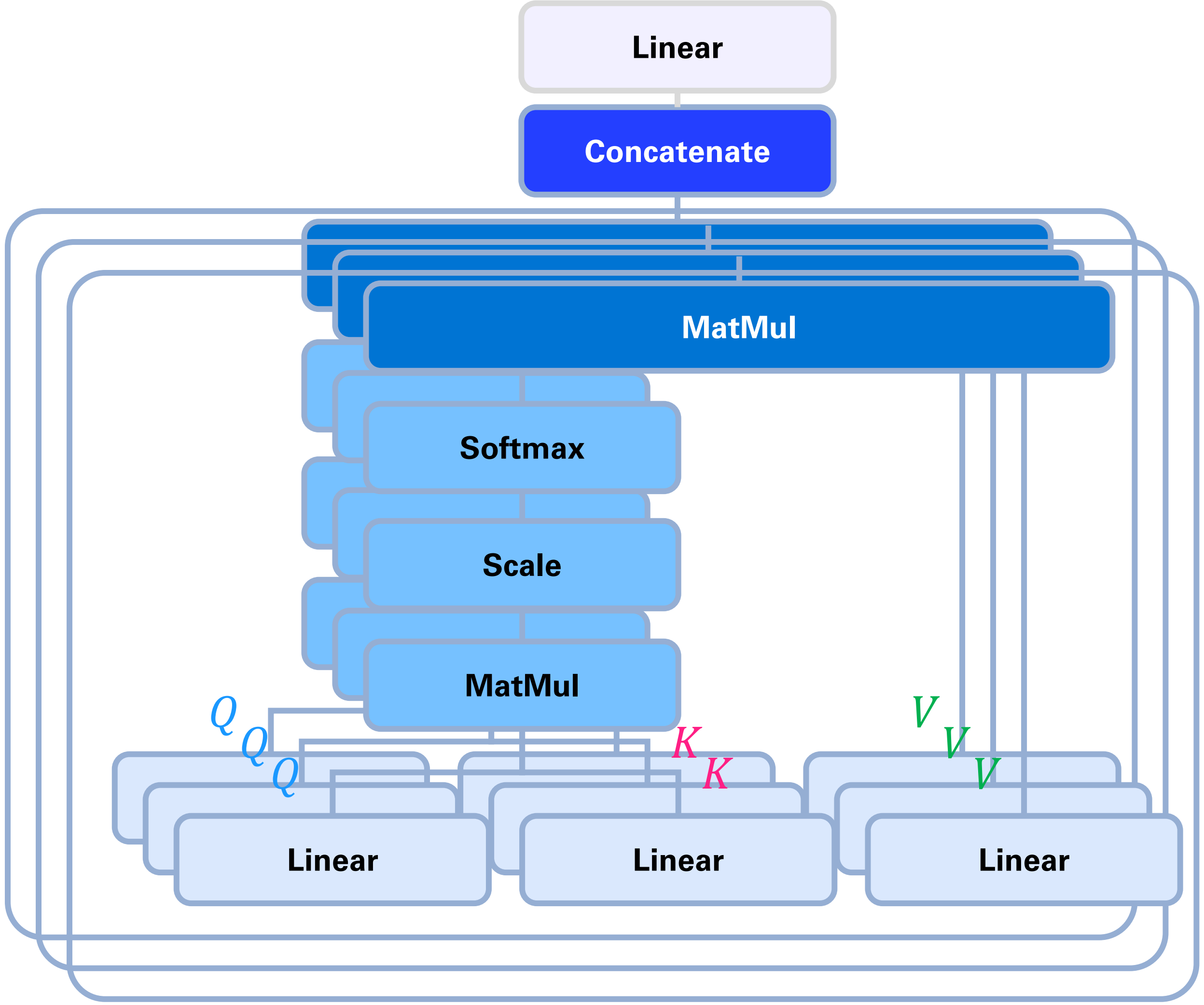
Output of attention head 3

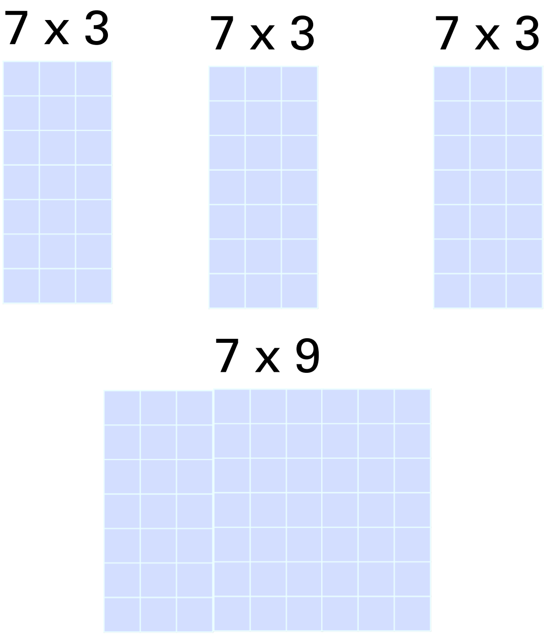
Multi-headed Self-Attention



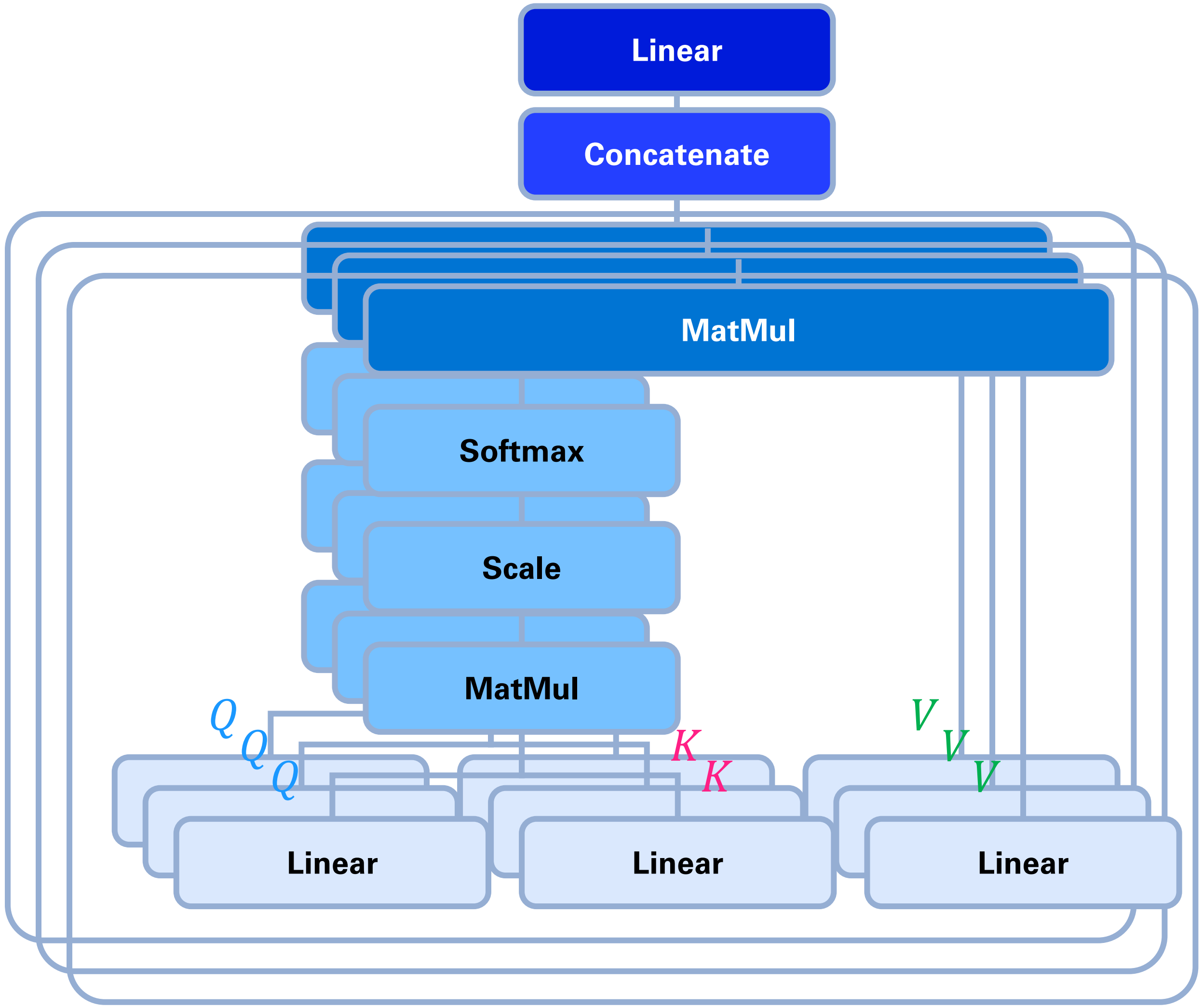


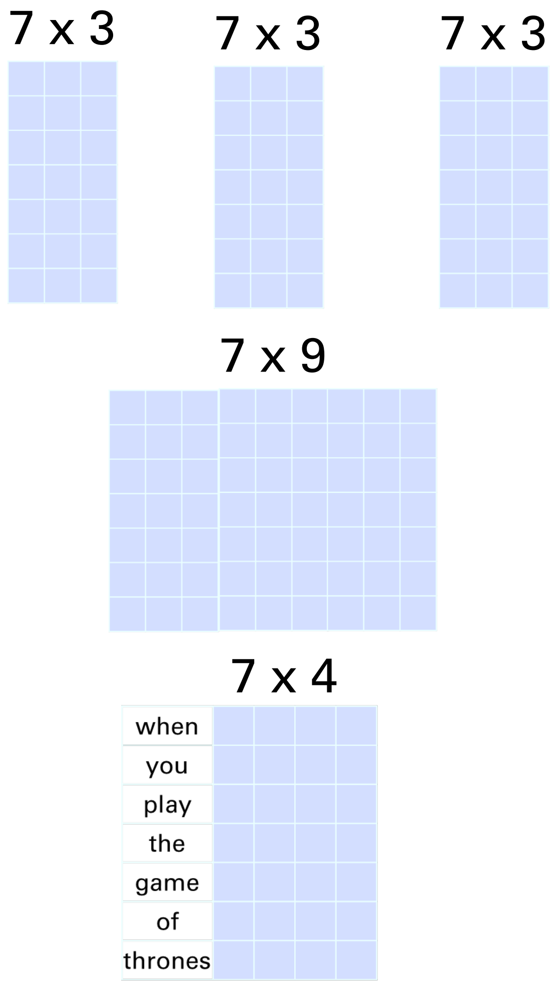
Multi-headed Self-Attention



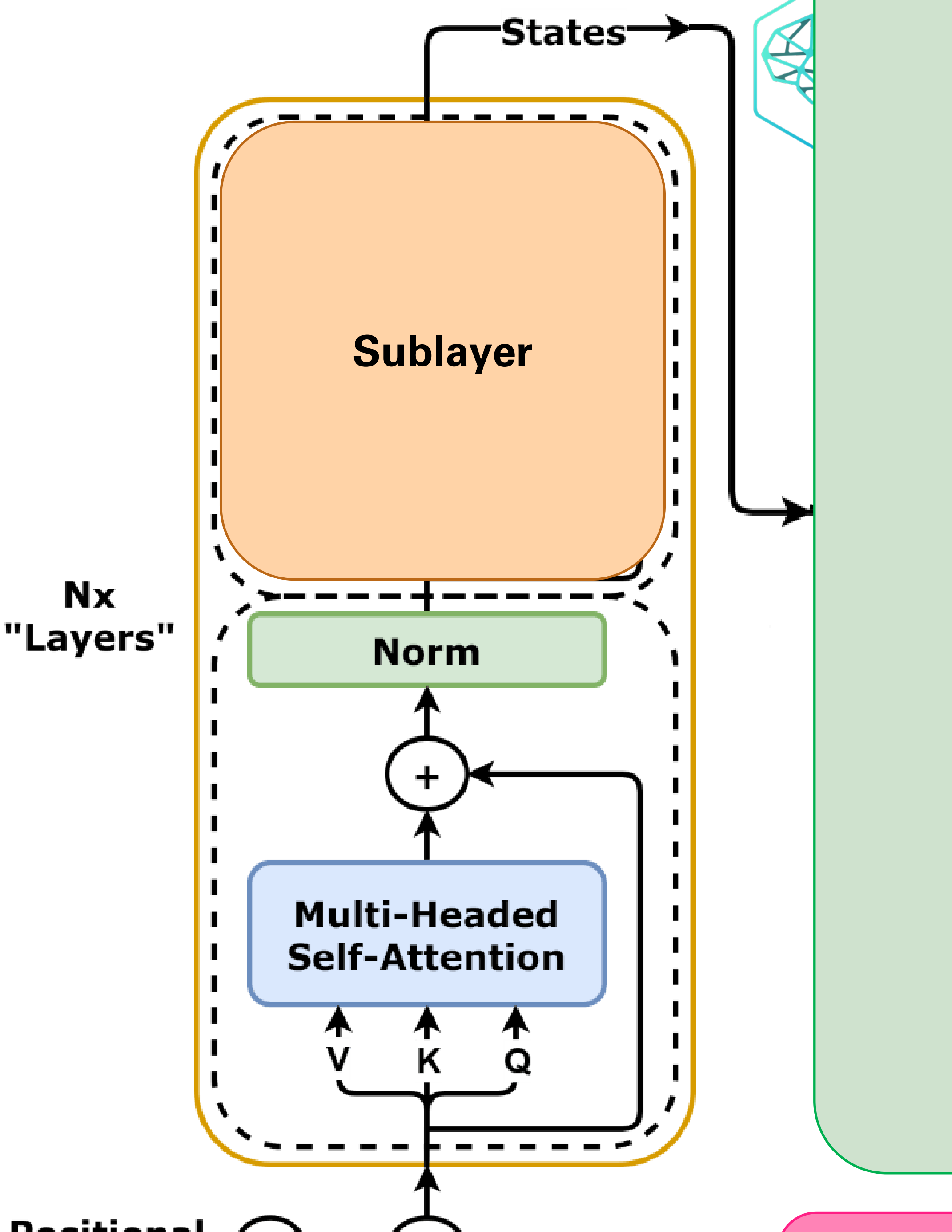


Multi-headed Self-Attention





Architecture

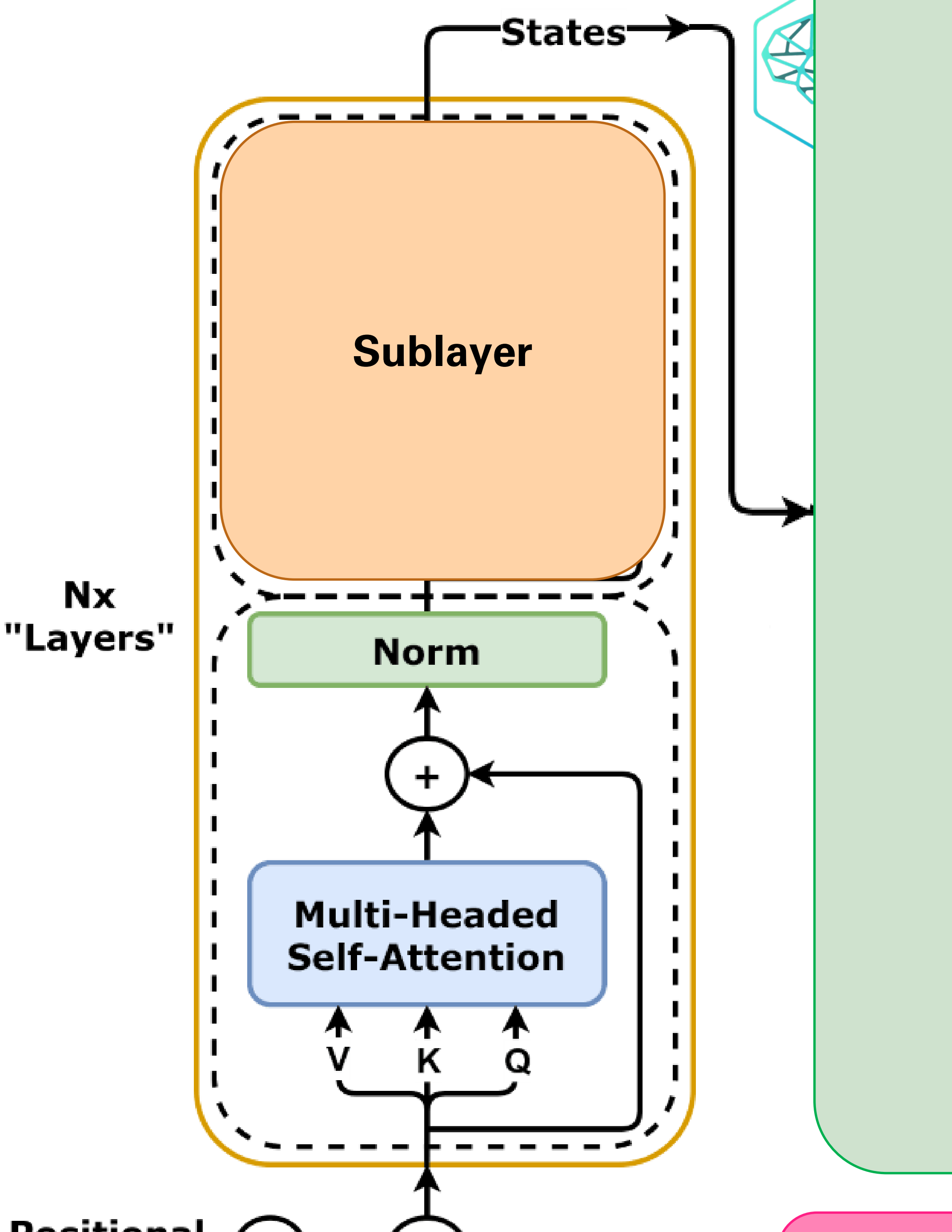


Architecture

To the output of Multi-Headed Self-Attention we apply:

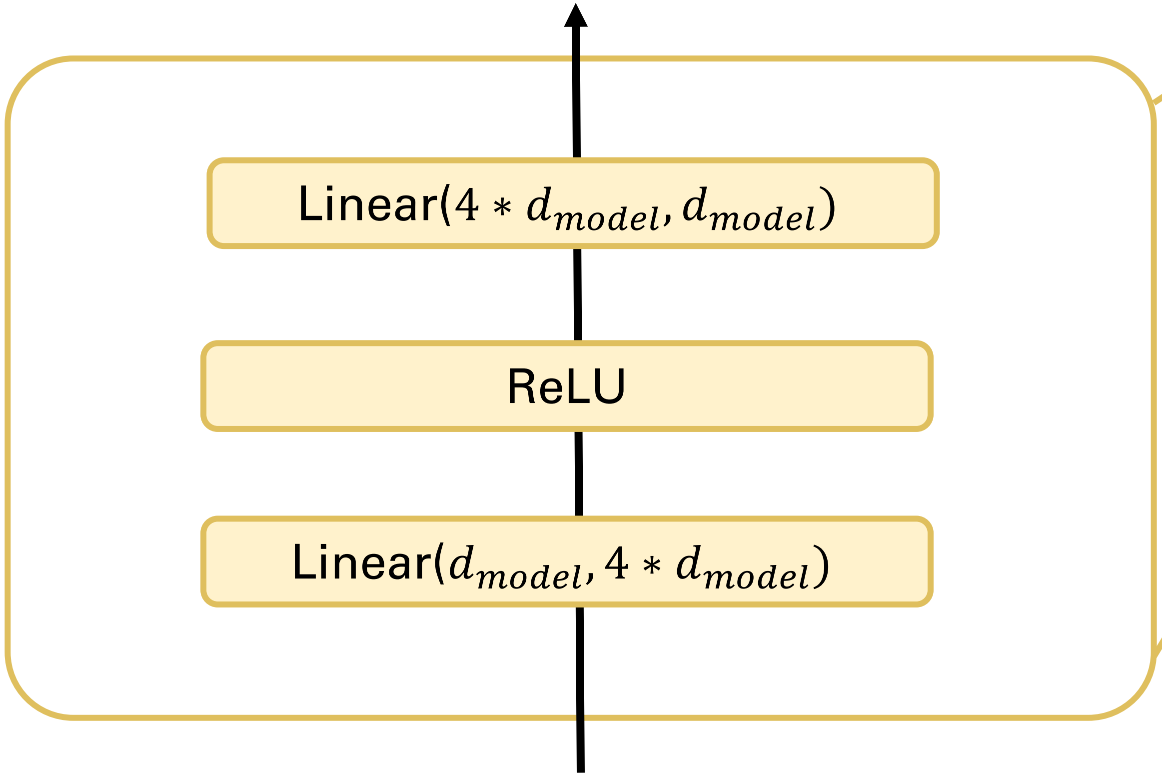
- A Residual Connection;

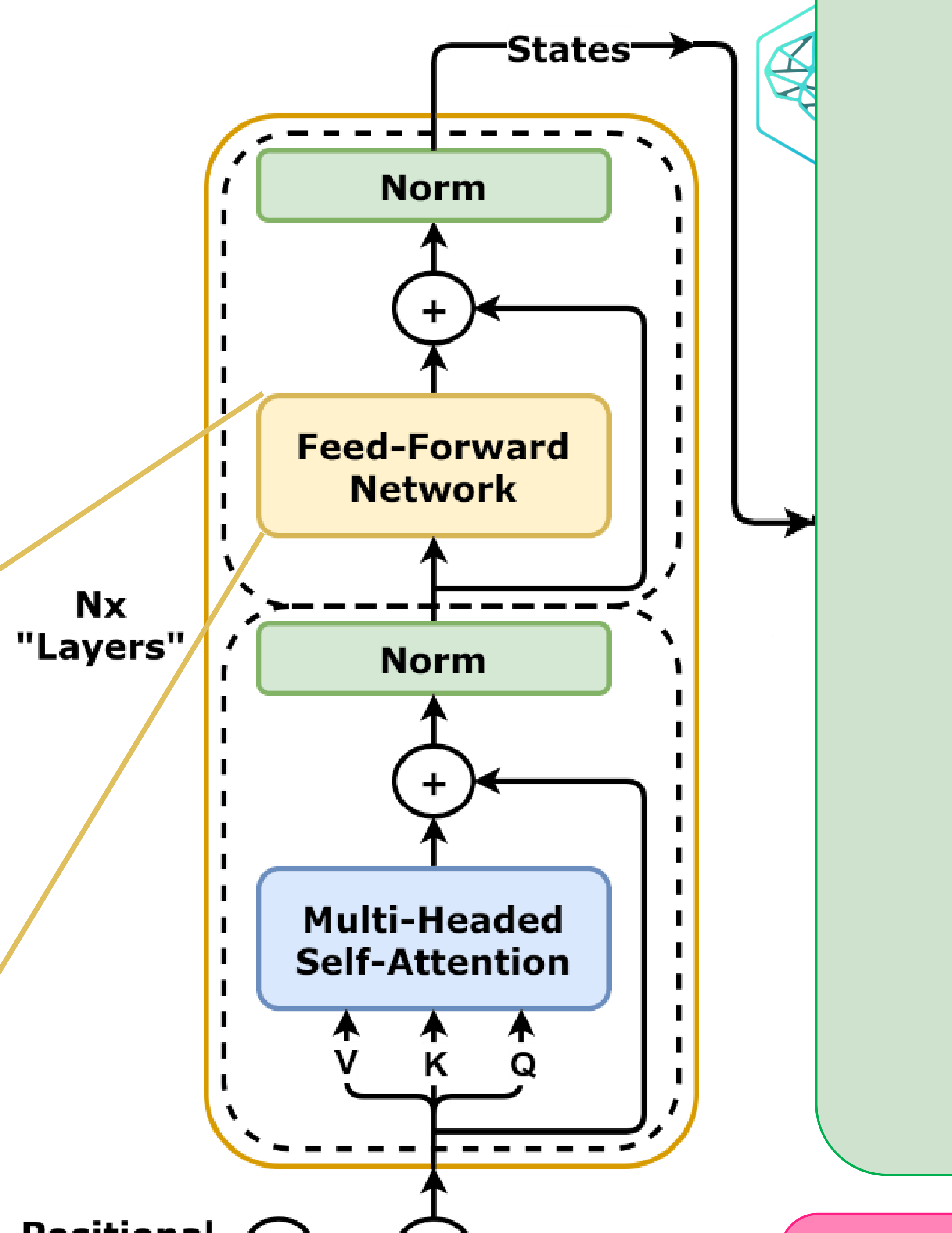
- A Normalization Block:
  - Batch Normalization
  - Layer Normalization



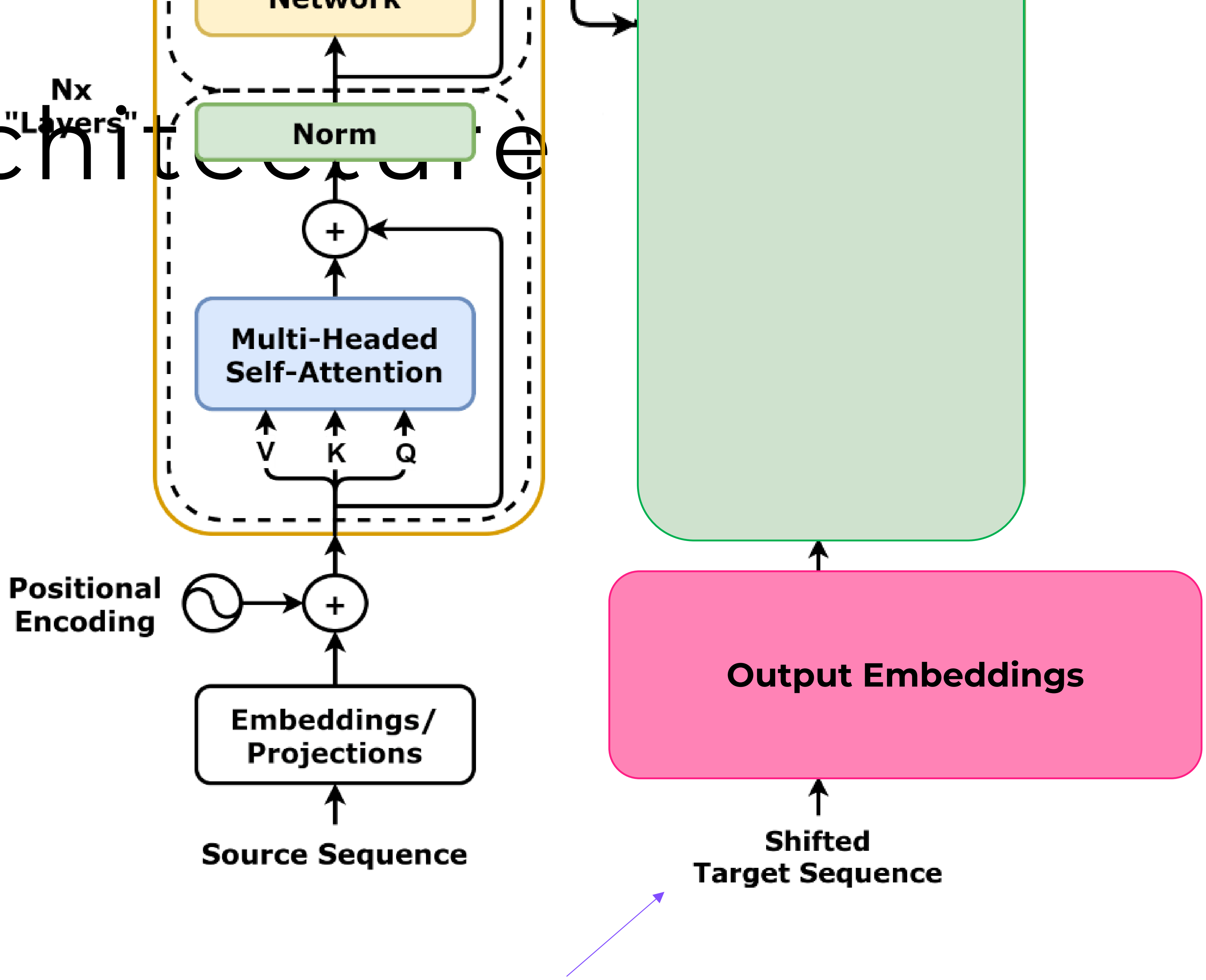
Architecture

A Feed-Forward layer ends the encoder stack.





Architecture



What is target sequence?

Like the source sequence...

y_0 y_1 y_2 $\cdots$ y_{t-1} y_t

It is a sequence of *tokens*

$(1, T)$

which tokens?

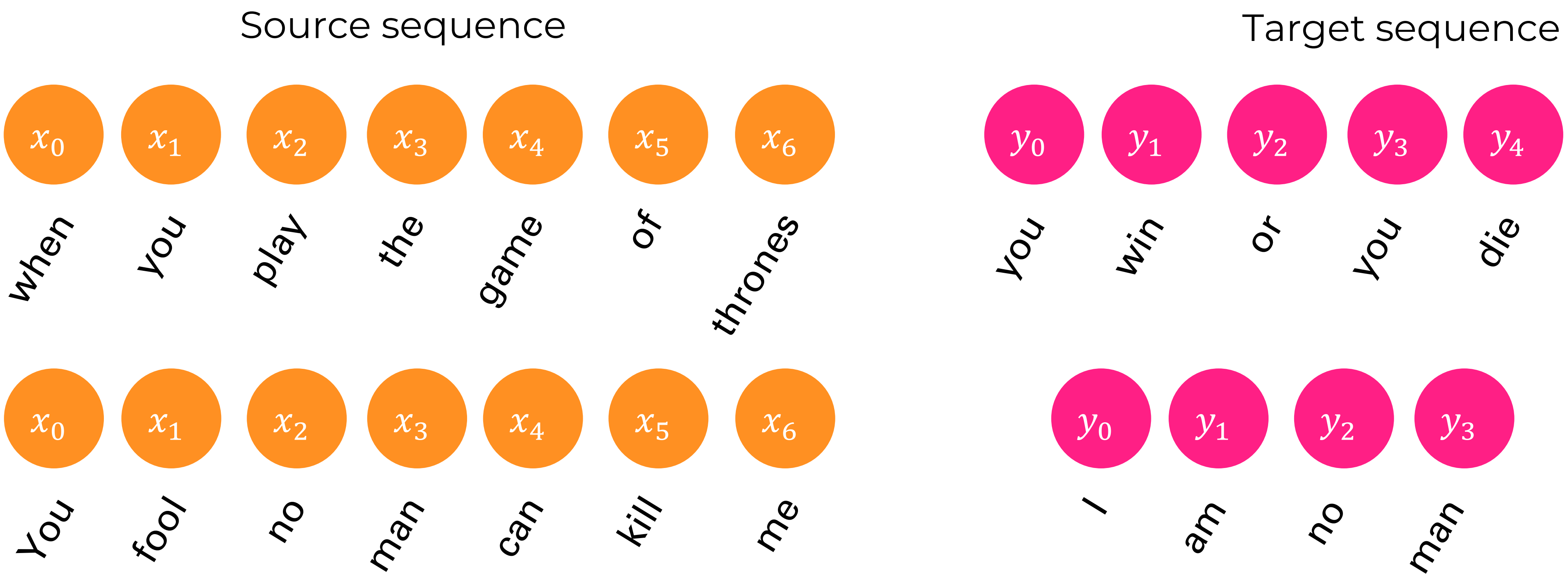
T:

- time dimension
- sequence length
- number of tokens

What is target sequence?

which tokens?

The sequence we want to predict!



What is target sequence?

which tokens?

The sequence we want to predict!

Source sequence

x_0	x_1	x_2	x_3	x_4	x_5	x_6
when	you	play	the	game	of	thrones

x_0	x_1	x_2	x_3	x_4	x_5	x_6
You	fool	no	man	can	kill	me

Shifted target sequence

y_0	y_1	y_2	y_3	y_4	y_5
	you	win	or	you	die

y_0	y_1	y_2	y_3	y_4
	I	am	no	man

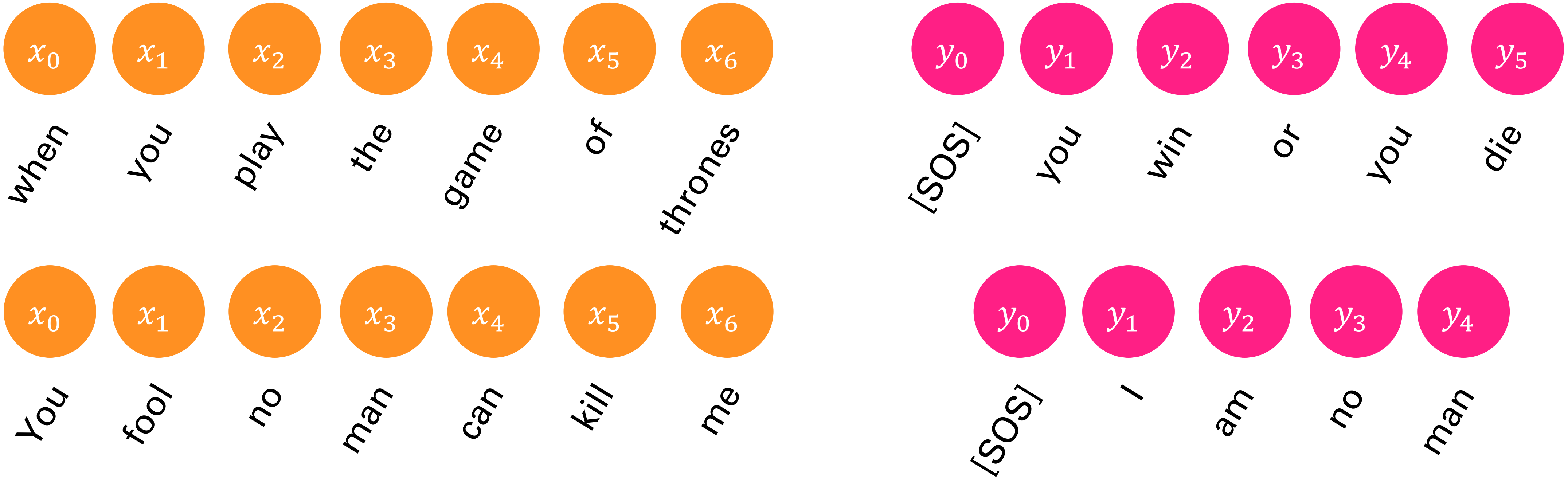
What is target sequence?

which tokens?

The sequence we want to predict!

Source sequence

Shifted target sequence



What is target sequence?

which tokens?

The sequence we want to predict!

Source sequence

x_0	x_1	x_2	x_3	x_4	x_5	x_6
when	you	play	the	game	of	thrones

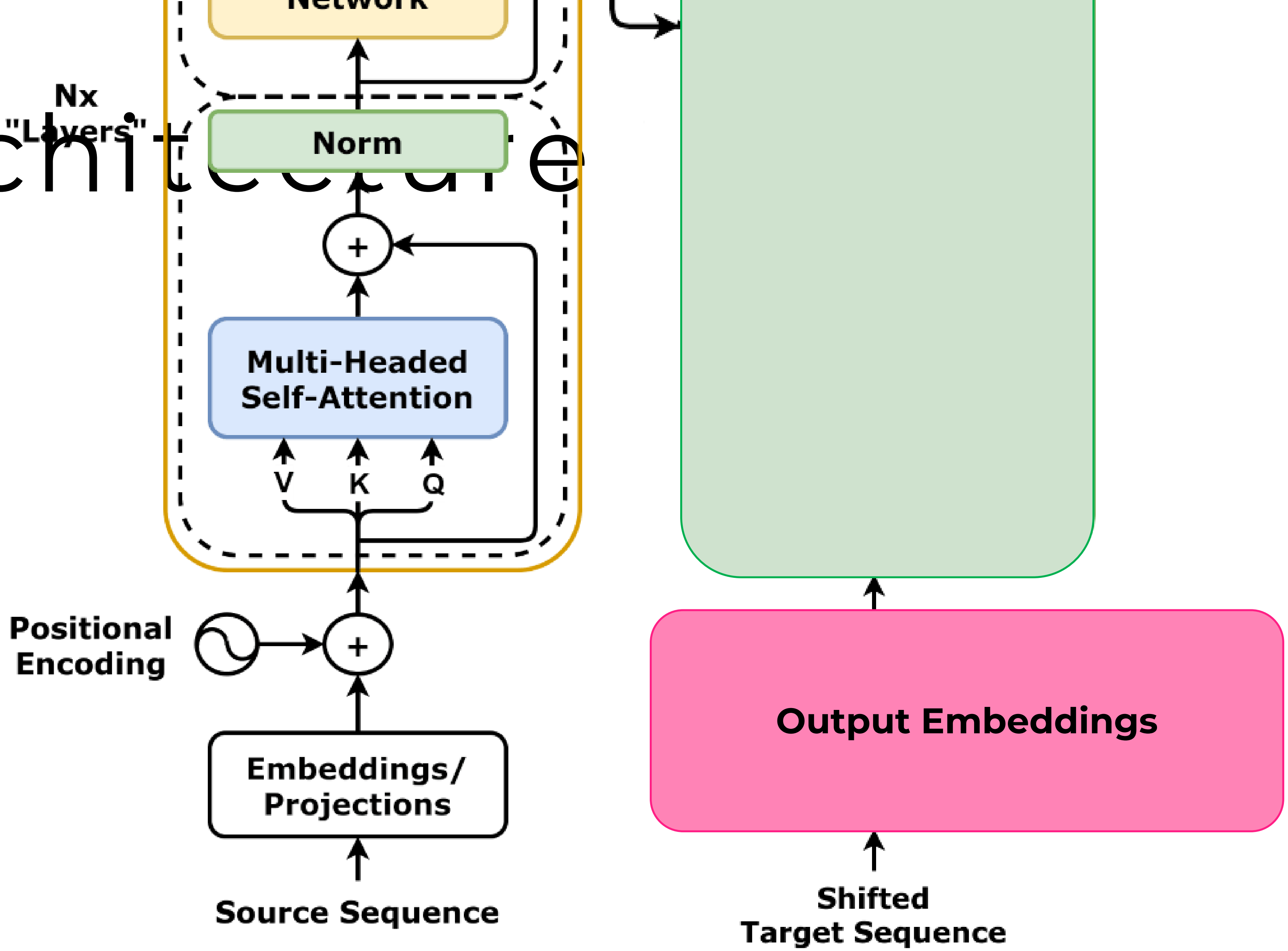
x_0	x_1	x_2	x_3	x_4	x_5	x_6
You	fool	no	man	can	kill	me

Shifted target sequence

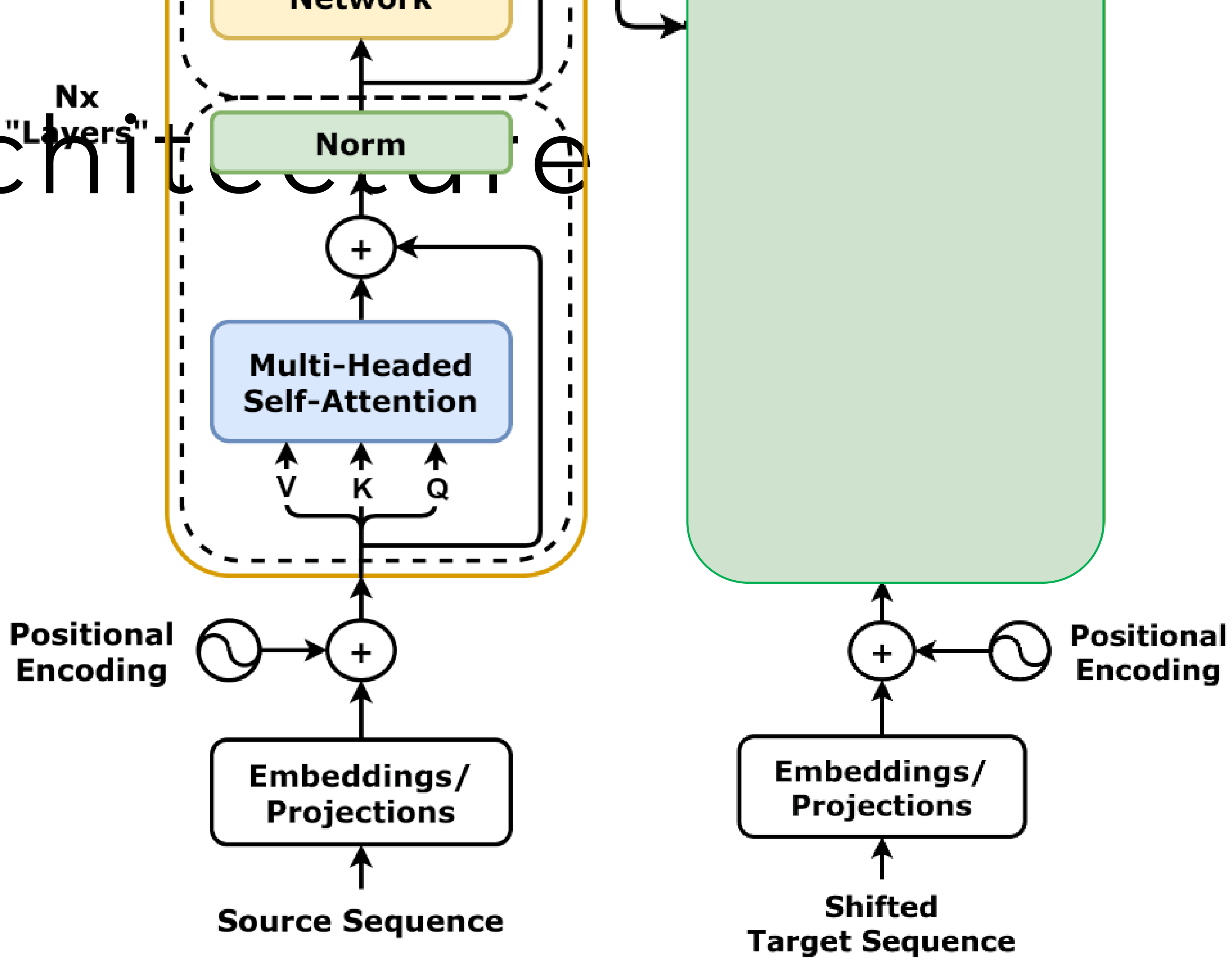
y_0	y_1	y_2	y_3	y_4	y_5	y_6
[SOS]	you	win	or	you	die	[EOS]

y_0	y_1	y_2	y_3	y_4	y_5
[SOS]	I	am	no	man	[EOS]

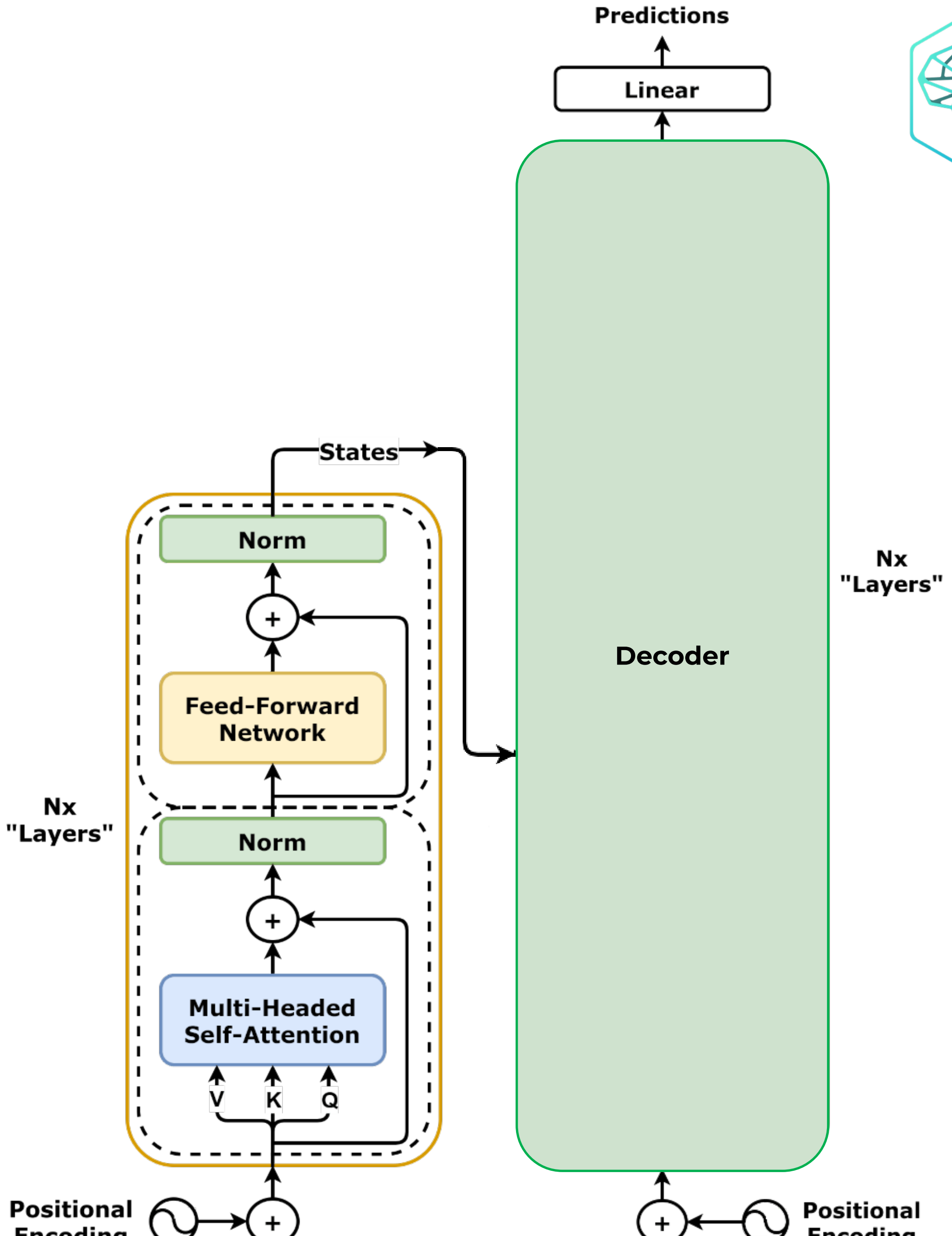
Architecture



Architecture



Architecture

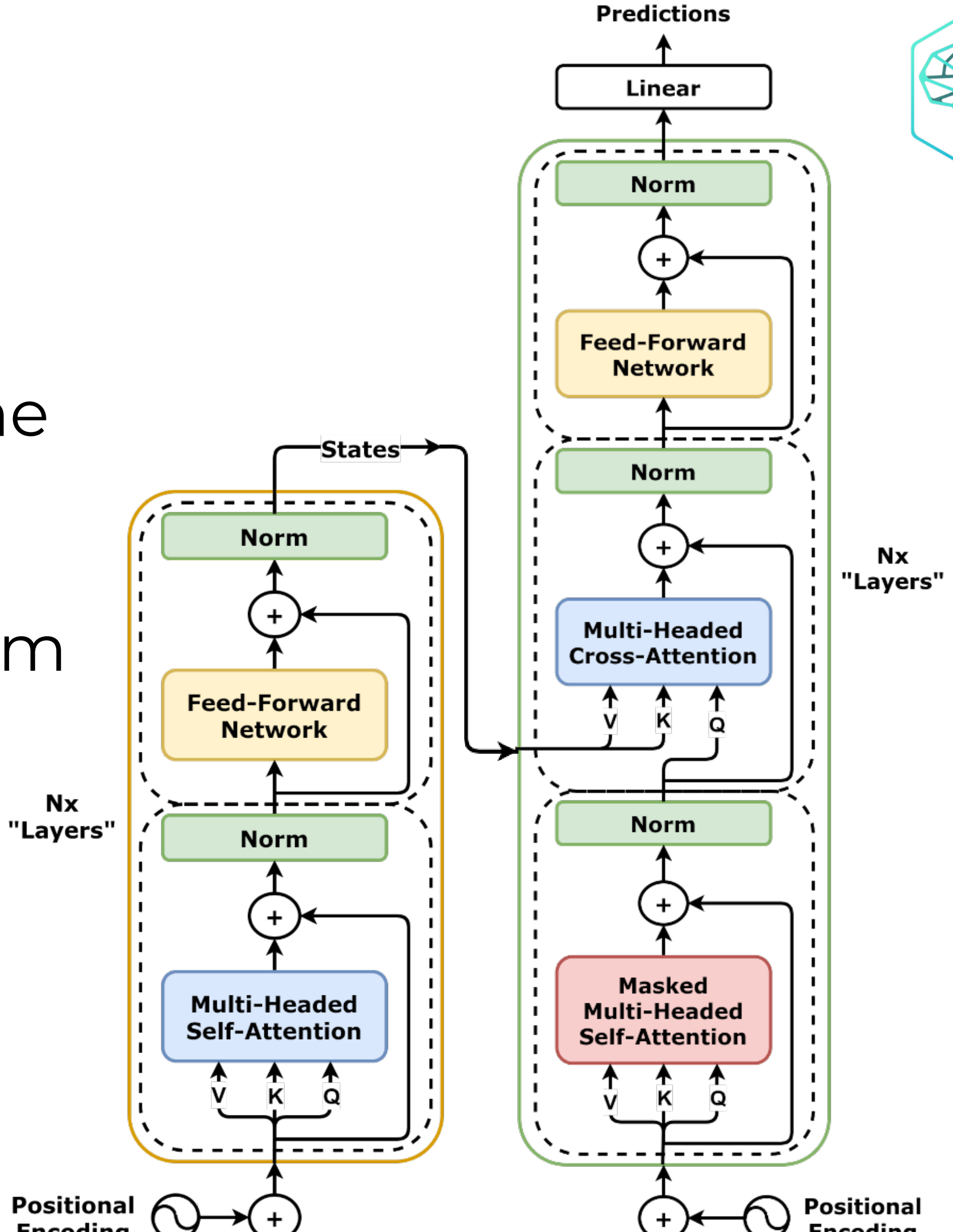


Architecture

Decoder stack is similar to the Encoder one.

The Self-Attention mechanism is sligthly modified by:

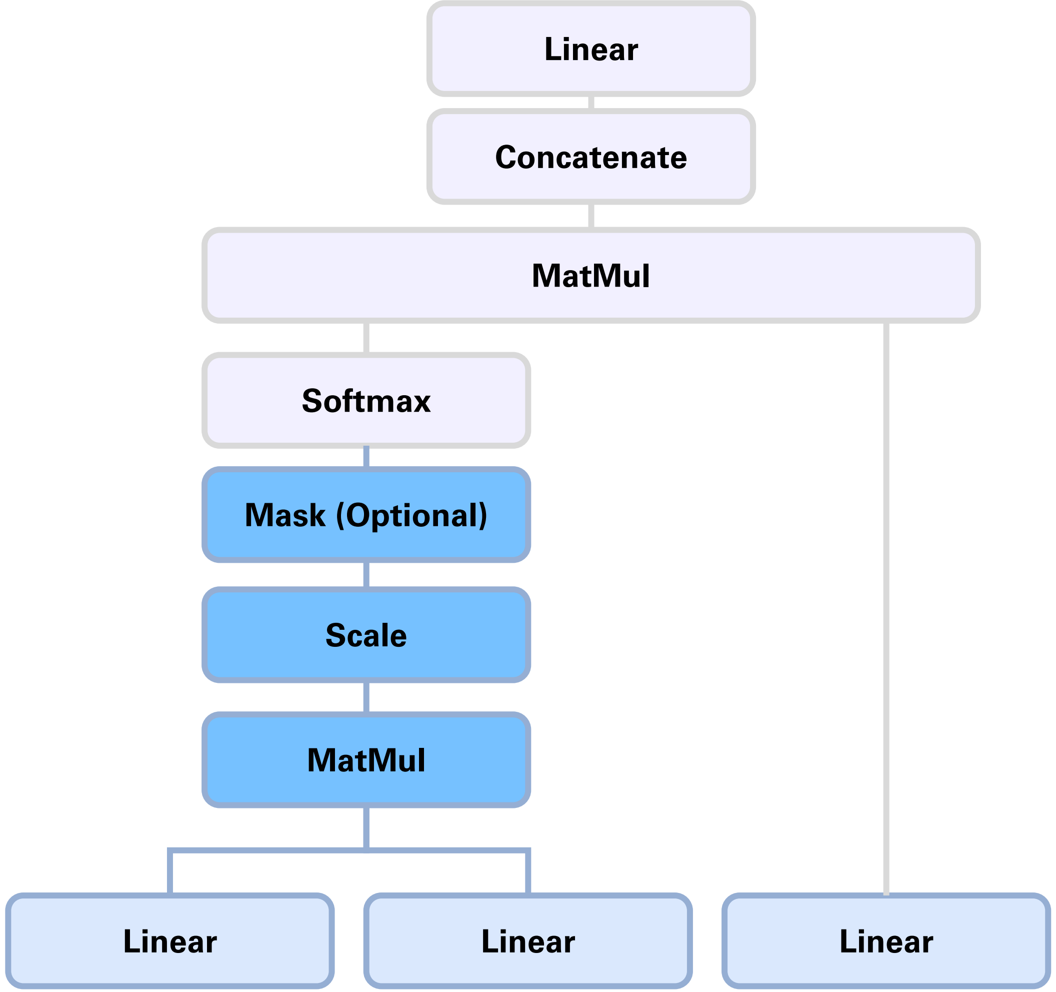
- *Masked* Self-Attention
- *Cross*-Attention



Masked Self-Attention

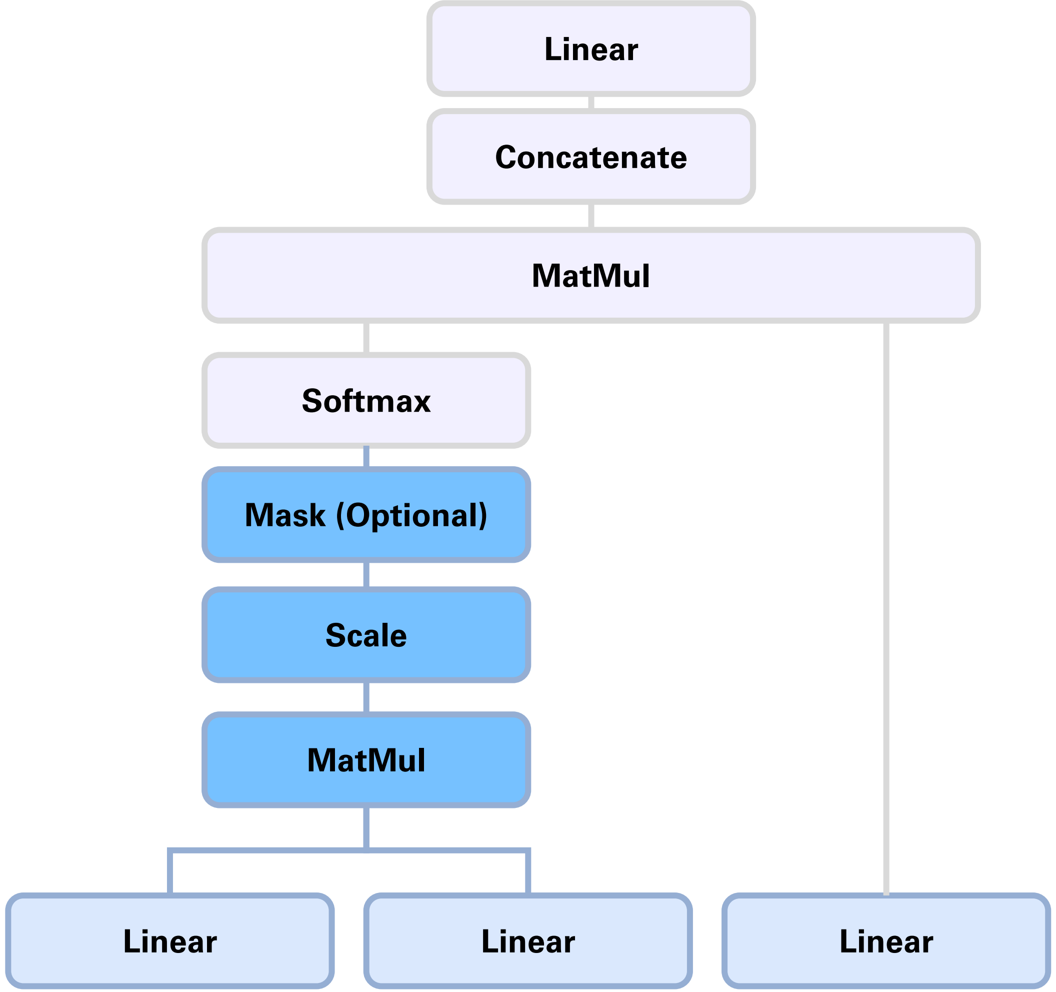
We don't want to cheat

Masked Self-Attention



	[SOS]	you	win	or	you	die	[EOS]
[SOS]	33.6	7.6	15.5	3.8	20.8	3.8	22.3
you	7.6	34.0	30.6	8.3	26.5	5.7	27.2
win	15.5	30.6	35.9	3.8	34.0	11.3	34.8
or	3.8	8.3	3.8	34.8	33.3	15.1	33.6
you	20.8	26.5	34.0	33.3	37.0	16.6	35.9
die	3.8	5.7	11.3	15.1	16.6	32.1	22.3
[EOS]	22.3	27.2	34.8	34.0	35.9	22.3	37.4

Masked Self-Attention



	[SOS]	you	win	or	you	die	[EOS]
[SOS]	33.6	7.6	15.5	3.8	20.8	3.8	22.3
you	7.6	34.0	30.6	8.3	26.5	5.7	27.2
win	15.5	30.6	35.9	3.8	34.0	11.3	34.8
or	3.8	8.3	3.8	34.8	33.3	15.1	33.6
you	20.8	26.5	34.0	33.3	37.0	16.6	35.9
die	3.8	5.7	11.3	15.1	16.6	32.1	22.3
[EOS]	22.3	27.2	34.8	34.0	35.9	22.3	37.4

+

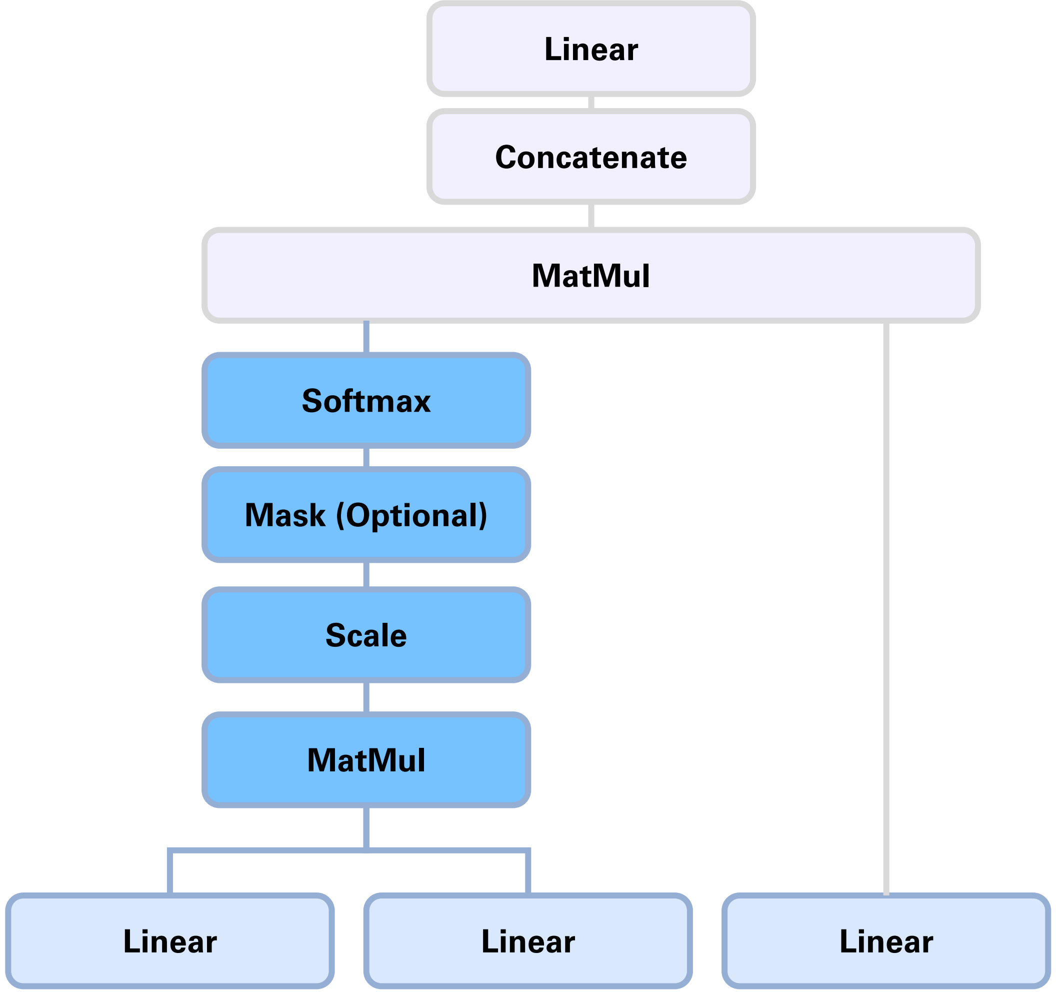
0	-inf	-inf	-inf	-inf	-inf	-inf
0	0	-inf	-inf	-inf	-inf	-inf
0	0	0	-inf	-inf	-inf	-inf
0	0	0	0	-inf	-inf	-inf
0	0	0	0	0	-inf	-inf
0	0	0	0	0	0	-inf
0	0	0	0	0	0	0

=

=

	[SOS]	you	win	or	you	die	[EOS]
[SOS]	33.6	-inf	-inf	-inf	-inf	-inf	-inf
you	7.6	34.0	-inf	-inf	-inf	-inf	-inf
win	15.5	30.6	35.9	-inf	-inf	-inf	-inf
or	3.8	8.3	3.8	34.8	-inf	-inf	-inf
you	20.8	26.5	34.0	33.3	37.0	-inf	-inf
die	3.8	5.7	11.3	15.1	26.6	32.1	-inf
[EOS]	22.3	27.2	34.8	34.0	35.9	22.3	37.4

Masked Self-Attention

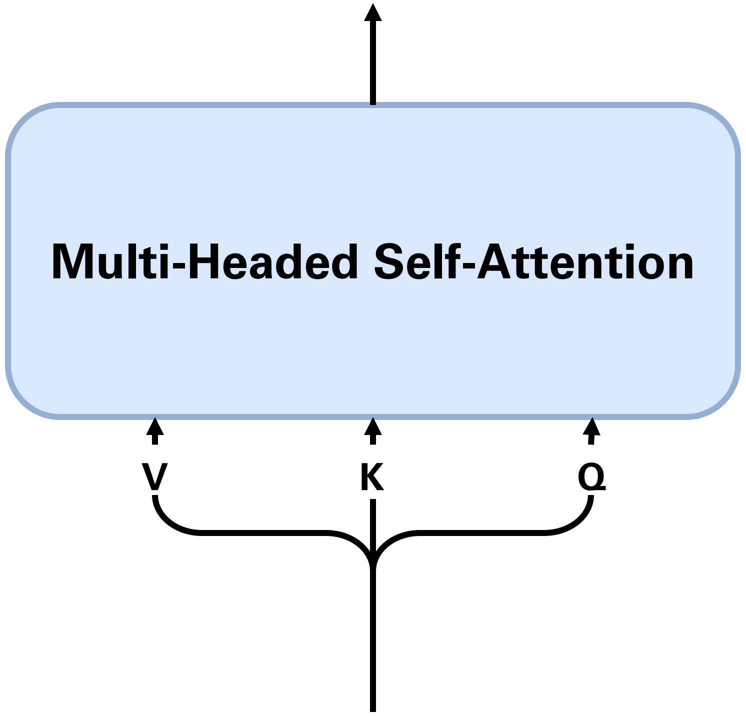


	[SOS]	you	win	or	you	die	[EOS]
[SOS]	1	0	0	0	0	0	0
you	0.01	0.99	0	0	0	0	0
win	0.001	0.004	0.995	0	0	0	0
or	0.003	0.004	0.003	0.99	0	0	0
you	0.003	0.003	0.04	0.02	0.93	0	0
die	0.001	0.001	0.001	0.001	0.001	0.995	0
[EOS]	0.00	0.00	0.05	0.03	0.17	0.00	0.75

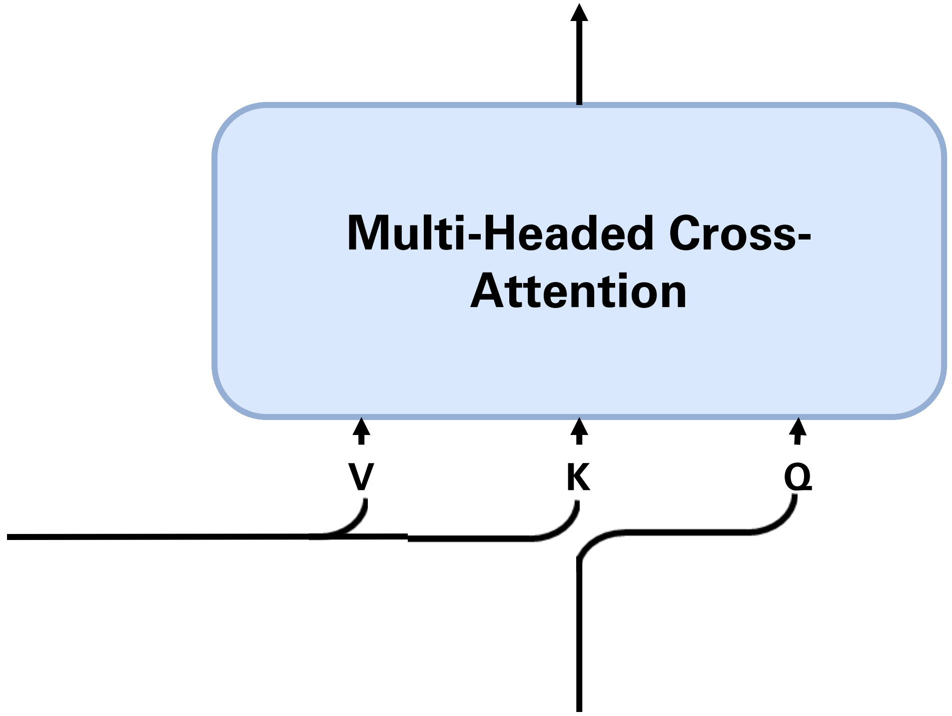
Masked Self-Attention

	[SOS]	you	win	or	you	die	[EOS]
[SOS]	1	0	0	0	0	0	0
you	0.01	0.99	0	0	0	0	0
win	0.001	0.004	0.995	0	0	0	0
or	0.003	0.004	0.003	0.99	0	0	0
you	0.003	0.003	0.04	0.02	0.93	0	0
die	0.001	0.001	0.001	0.001	0.001	0.995	0
[EOS]	0.00	0.00	0.05	0.03	0.17	0.00	0.75

Cross-Attention



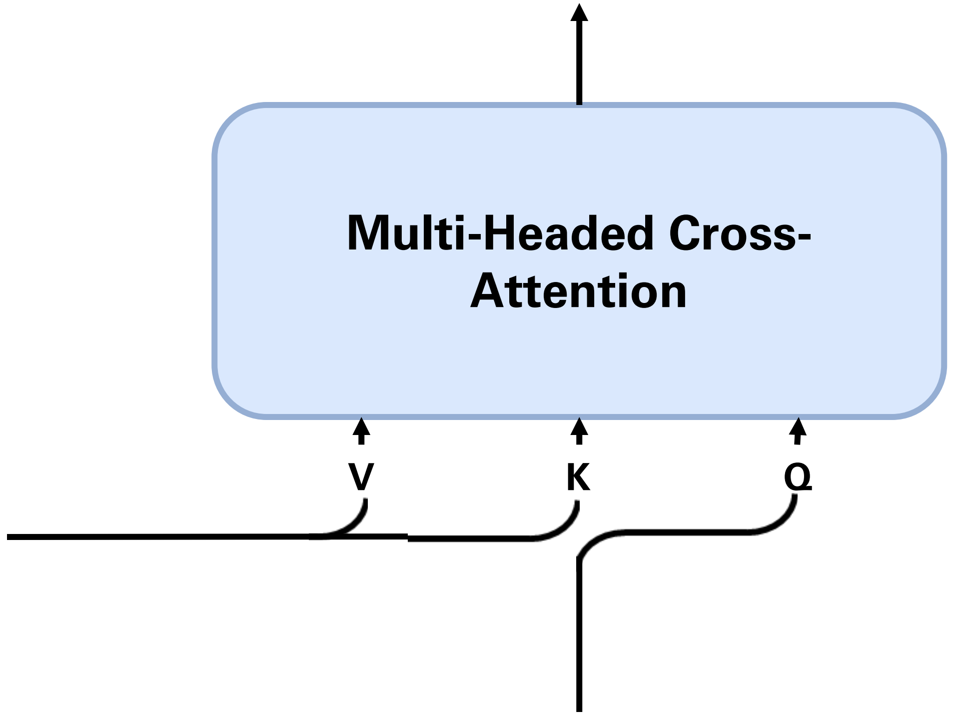
Queries, Keys and Values all come from the same sequence



Queries come from the target sequence.
Keys and Values come from the source sequence (last hidden states)

Cross-Attention

Keys and Values always come in pairs



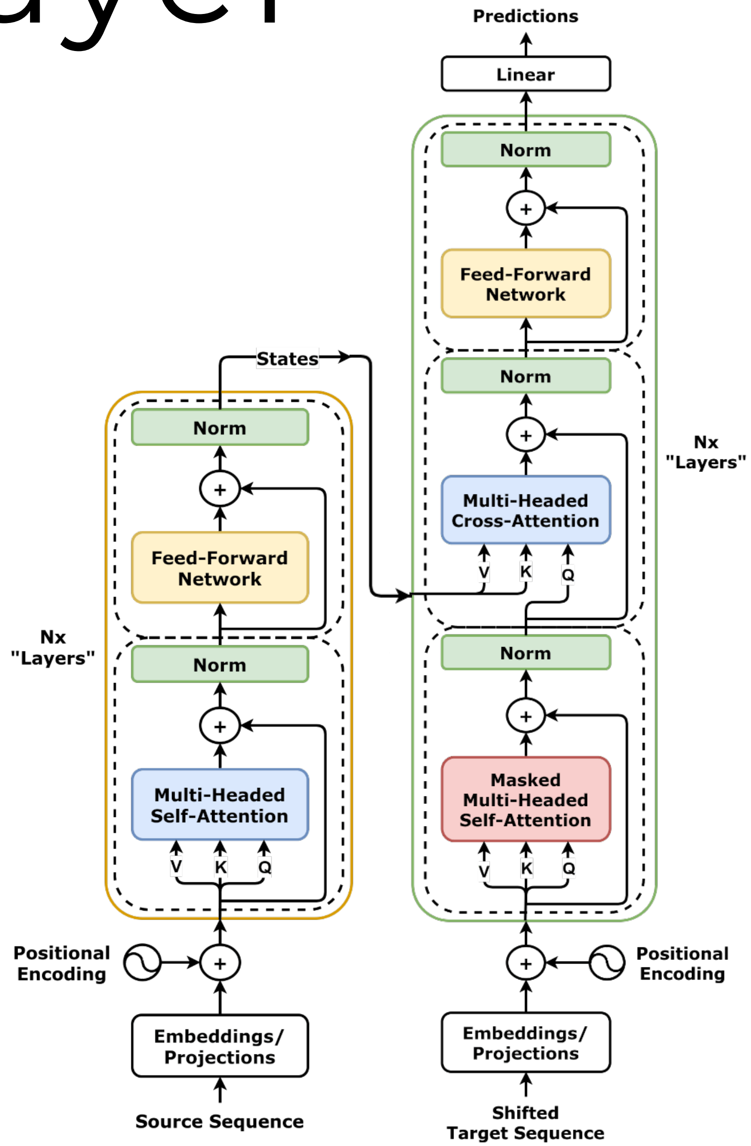
Queries come from the target sequence.
Keys and Values come from the source sequence (last hidden states)

The final linear layer

As always, all tasks can be regarded as classification or regression problems

Here we have a classification problem:

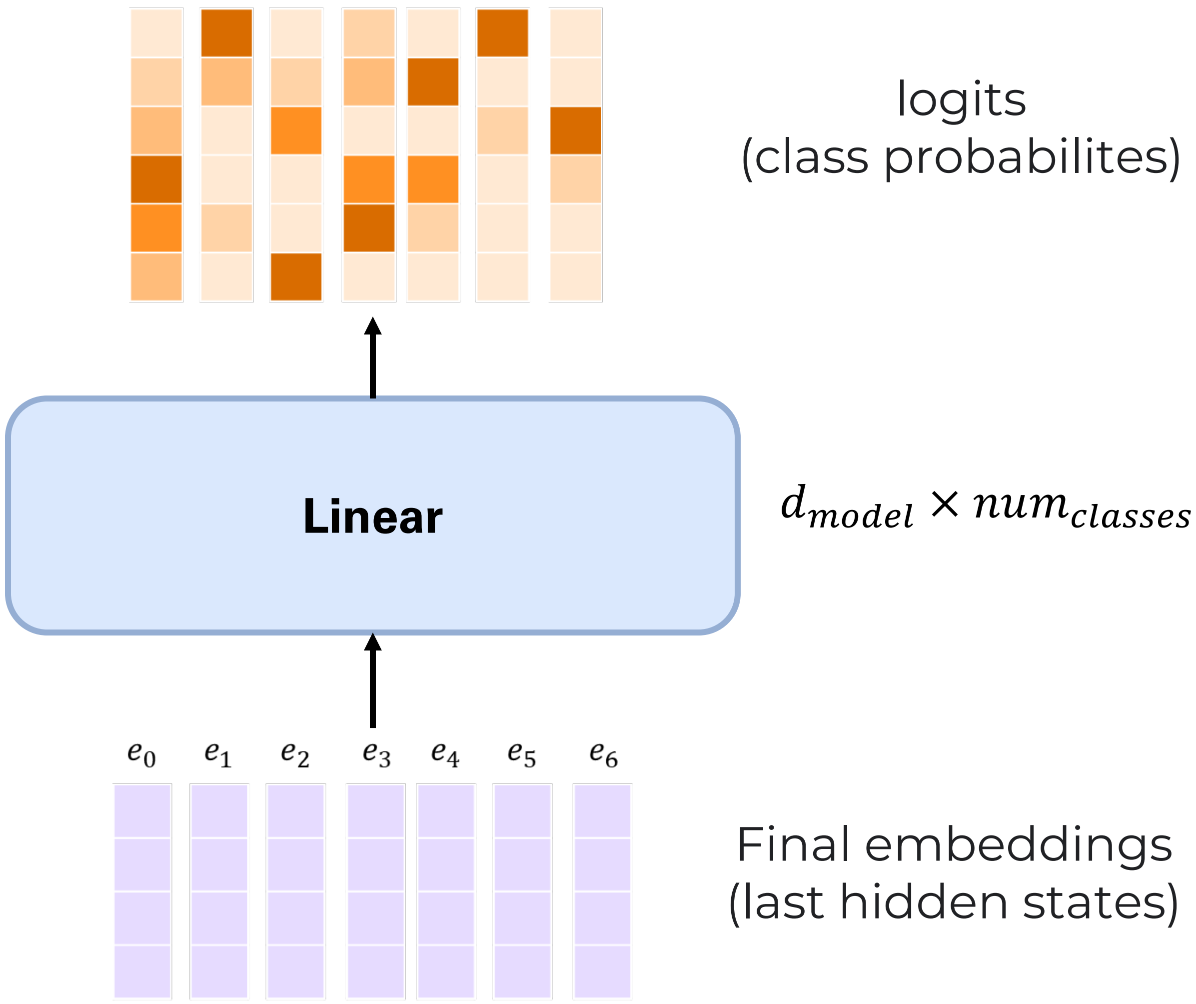
- for each input token we want to predict the next one
- we choose between all the known words (the size of vocabulary)



The final linear layer

Vocabulary	
The	0
cat	1
is	2
...	
provide	29998
access	29999

30k words in our vocabulary
⇒ 30k classes

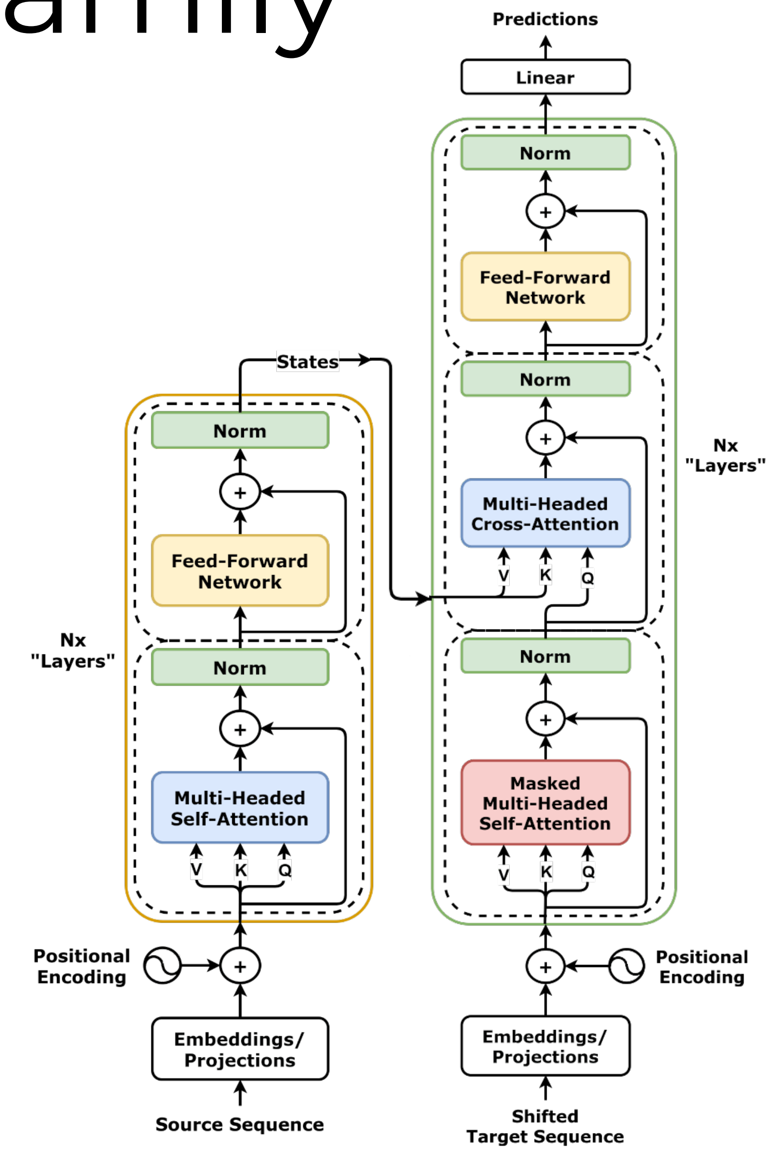


The Transformers Family

We don’t need always the complete architecture.

We can have:

- Encoder-Only Models
- Decoder-Only Models
- Encoder-Decoder Models

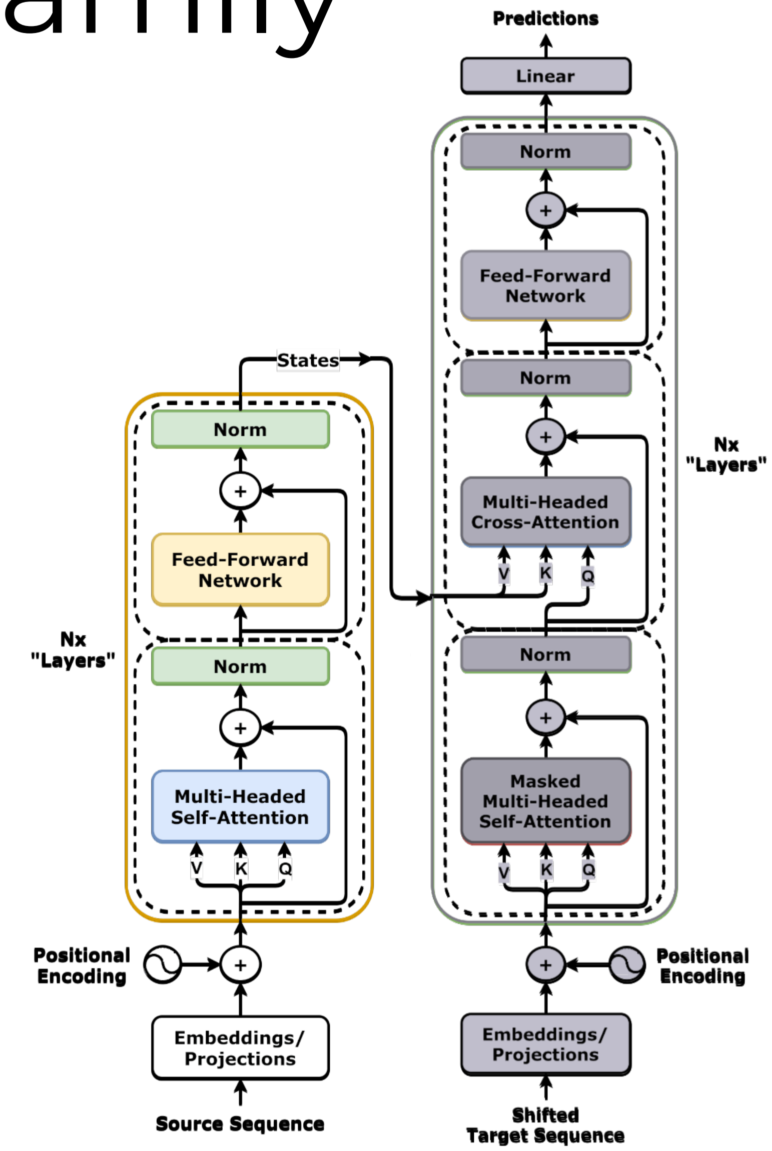


The Transformers Family

Encoders are suitable anytime we want to represent a sequence in a latent space

Famous Encoder architectures:

- BERT
- ELECTRA
- ViT

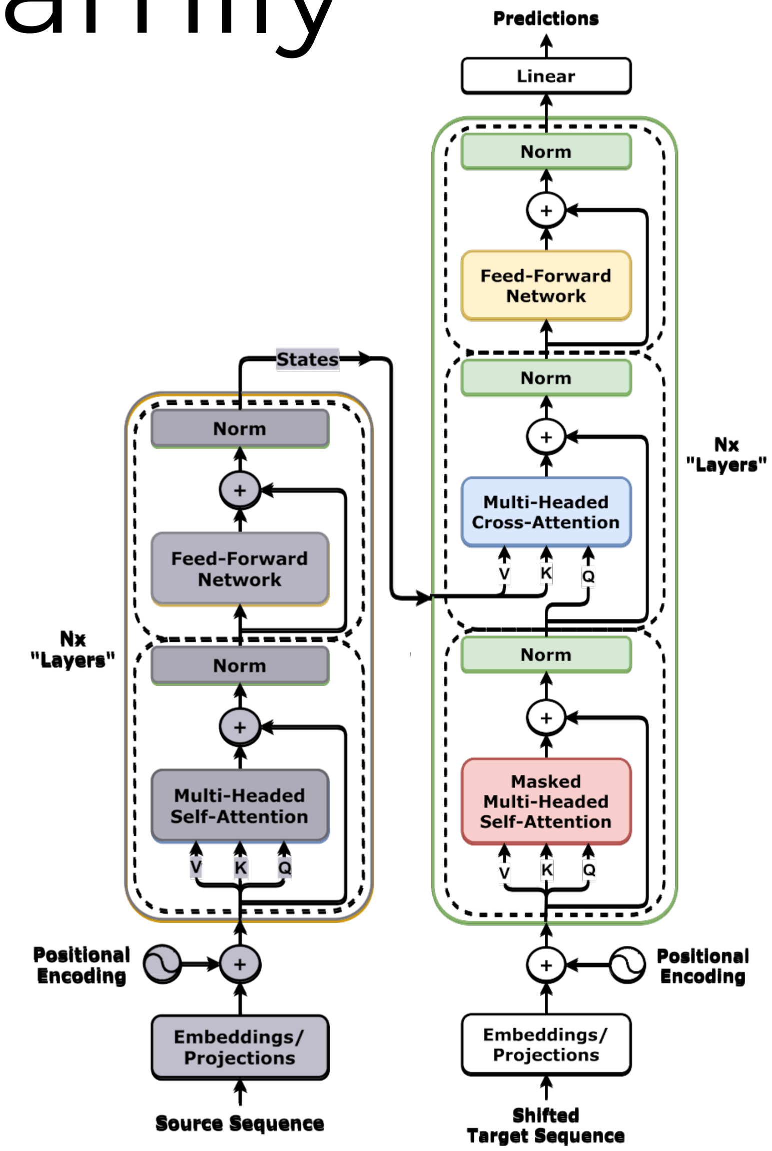


The Transformers Family

Decoders are suitable anytime we want to generate something (Text Generation)

Famous Decoder architectures:

- GPT

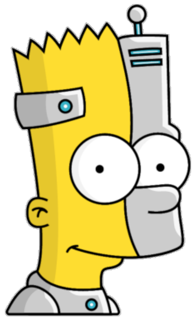


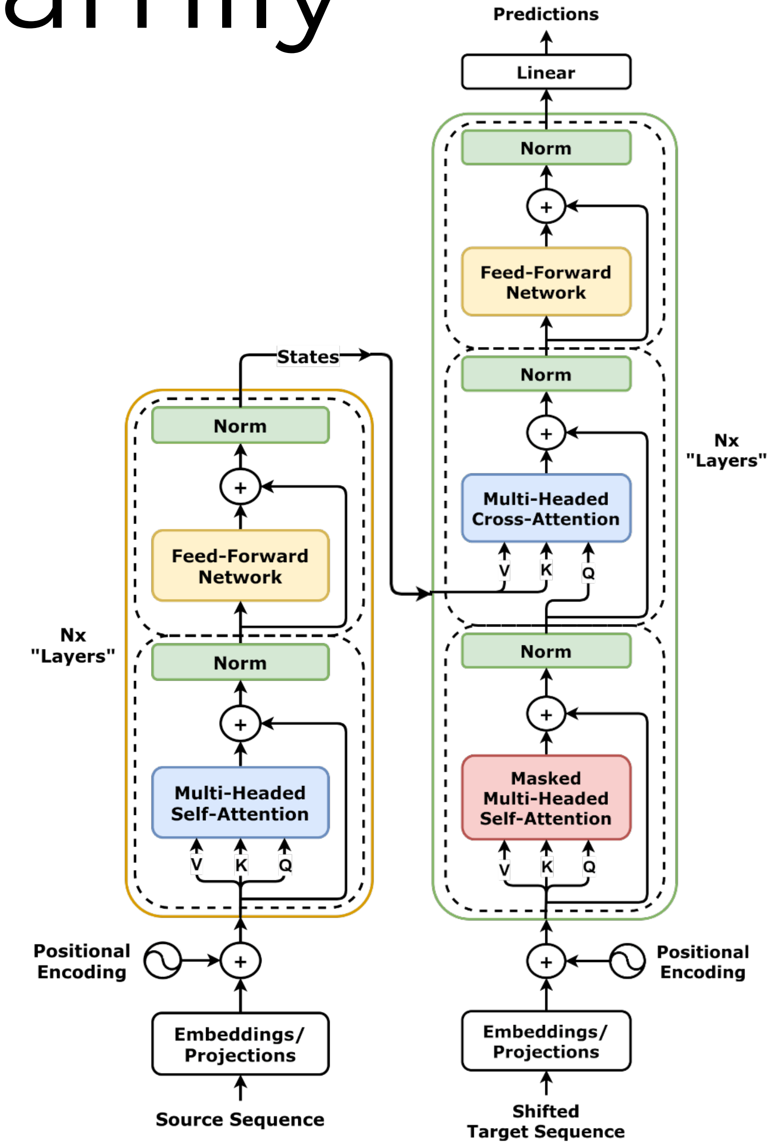
The Transformers Family

Encoder-Decoder is used anytime we want to predict a new sequence given a source sequence (Machine Translation, Forecasting, Summarization…)

Famous Encoder-Decoder models:

- BART
- T5





VISION TRANSFORMER (VIT)

Transformers were born for text…

- Transformers are domain-agnostic models, designed to process any type of sequential input.

- Any input that can be represented as a sequence of tokens can be fed into a Transformer model:
  - Audio
  - Protein sequences
  - Time series
  - EEG signals
  - …

What about images?

How can we convert images to sequences?

- Images are 2D grids of pixels, not sequences.

How can we convert images to sequences?

- Images are 2D grids of pixels, not sequences

But…

How can we convert images to sequences?

- Images are 2D grid of pixels, not sequences

But…

…we can reshape them into a sequence of **patches**

How can we convert images to sequences?

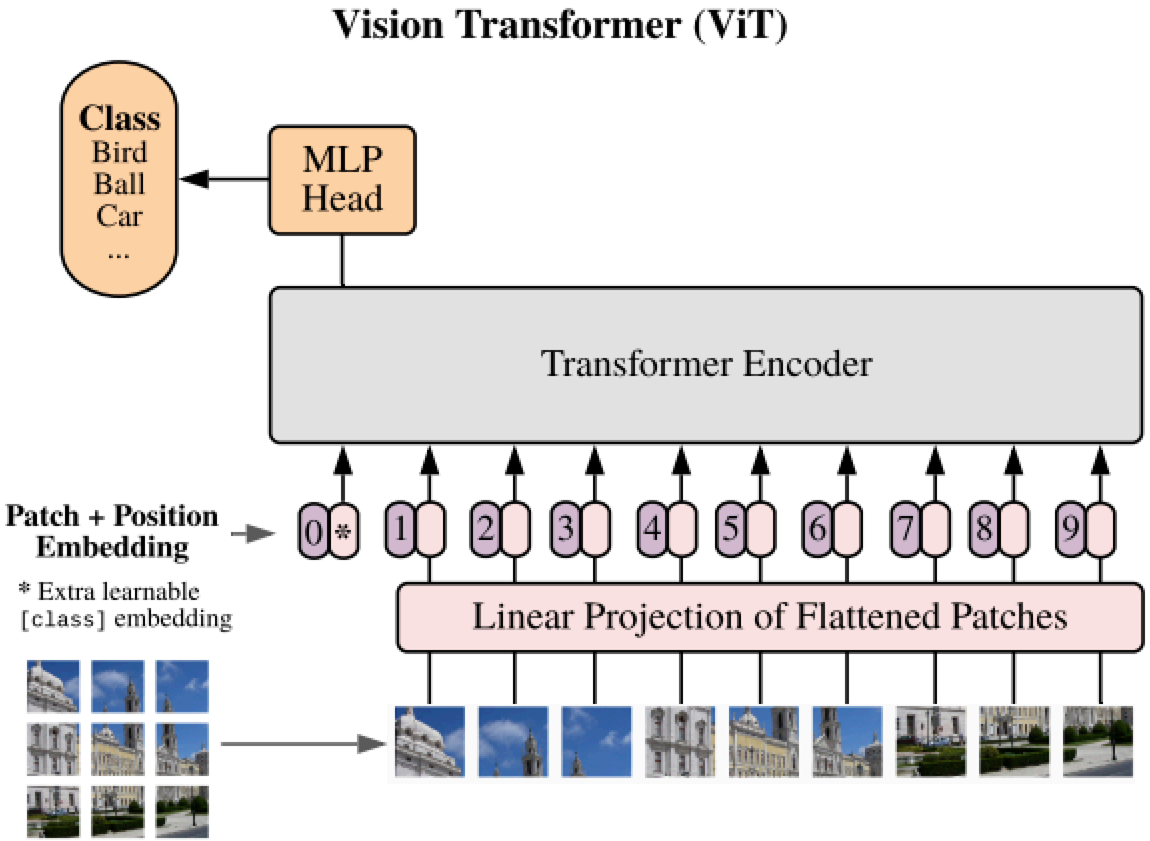
1) Image is divided into *patches*

2) Each patch is treated as a token in a sequence

3) Since patches have 2 dimensions they are flattened

ViT Architecture

Then, we can use the Transformer Architecture as we know

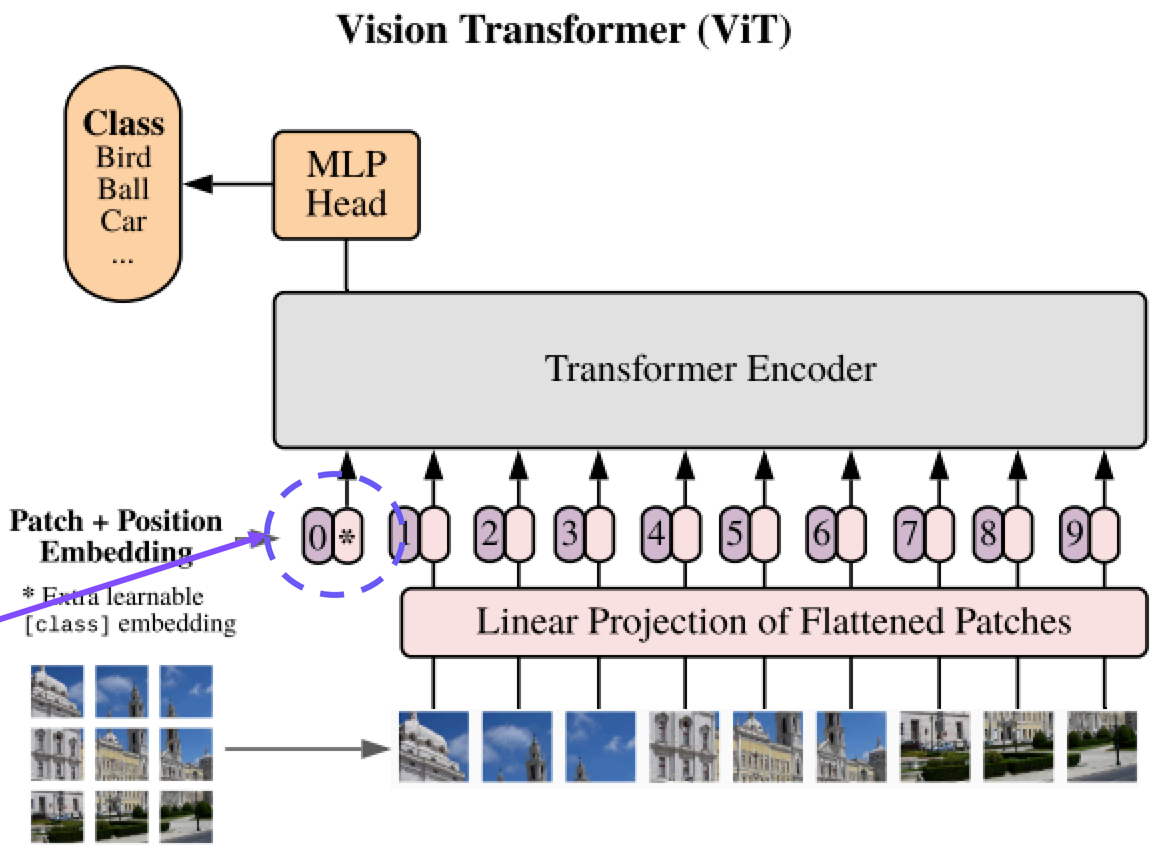


Dosovitskiy et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. ICLR2021. [Paper]

ViT Architecture

Then, we can use the Transformer Architecture as we know

We are adding the special [CLS] token to the embeddings.

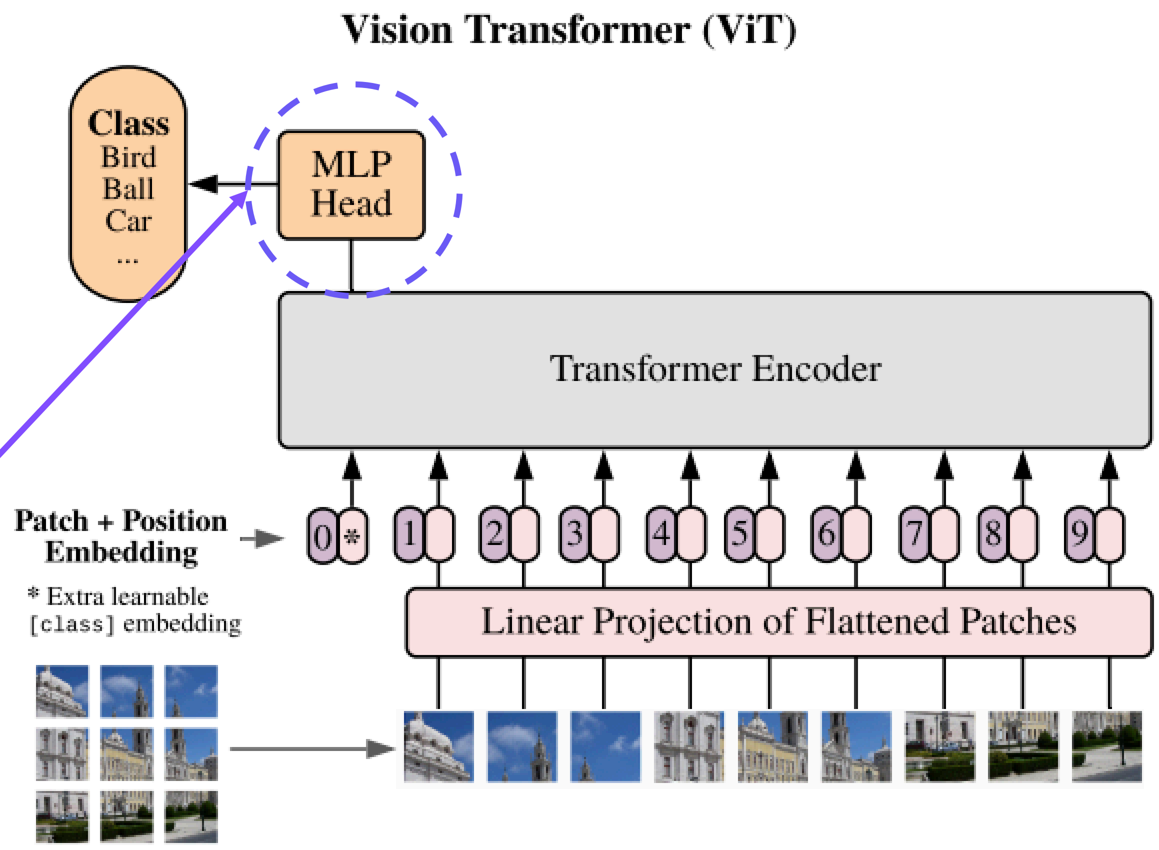


Dosovitskiy et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. ICLR2021. [Paper]

ViT Architecture

Then, we can use the Transformer Architecture as we know

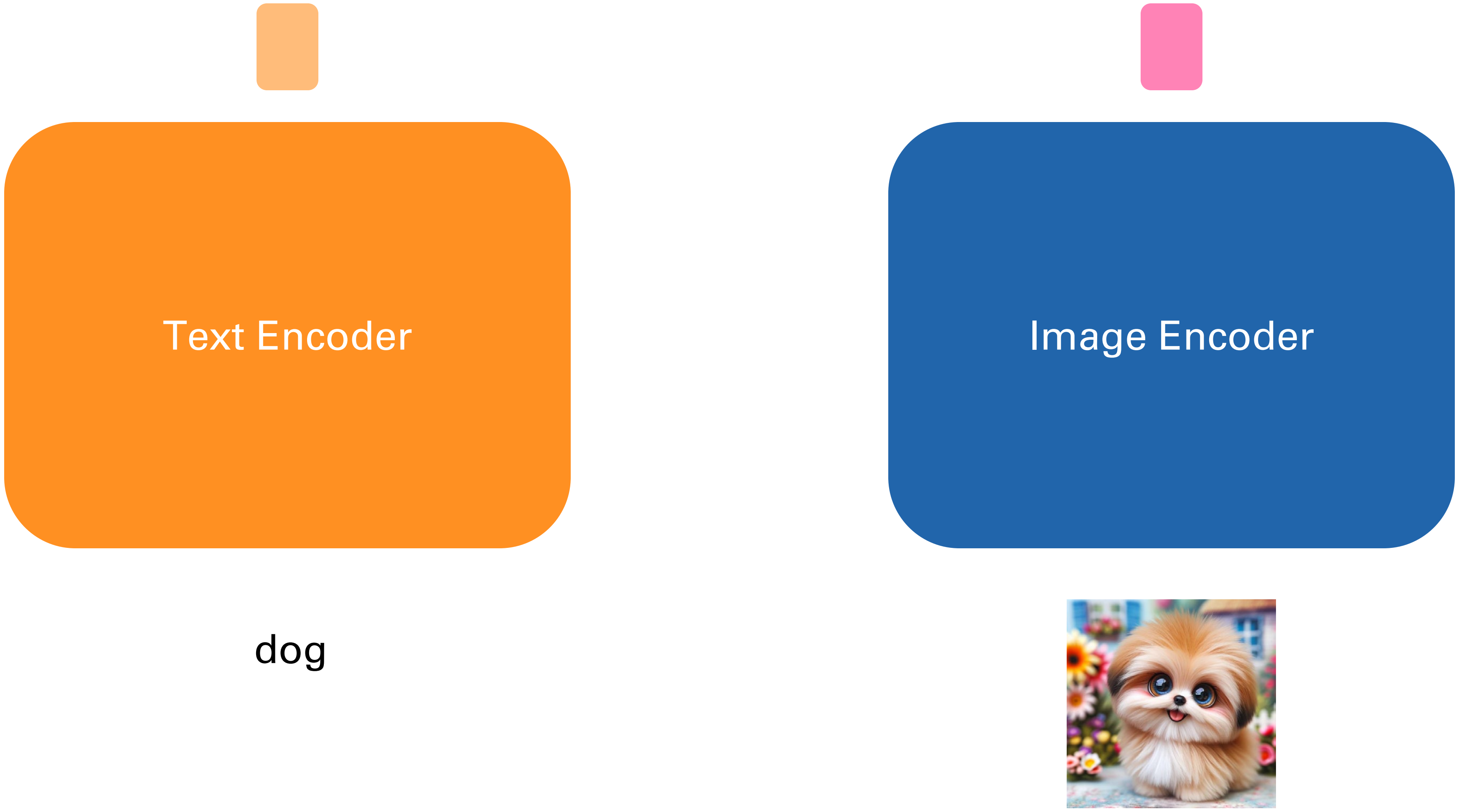
At the end we use only the [CLS] hidden state to classify our image



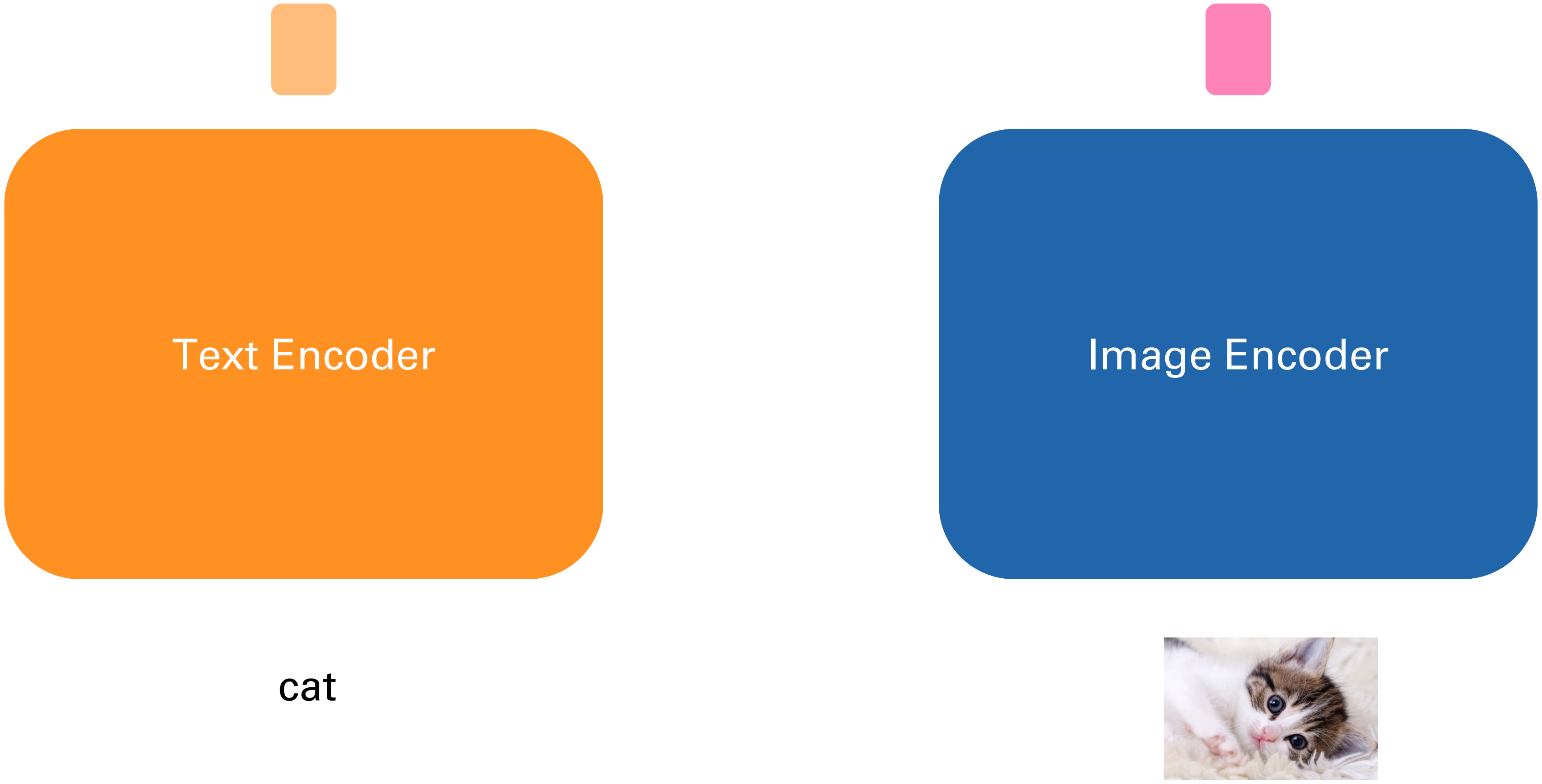
Dosovitskiy et al. “An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale”. ICLR2021. [Paper]

CLIP: CREATE A BRIDGE TO MULTIMODALITY

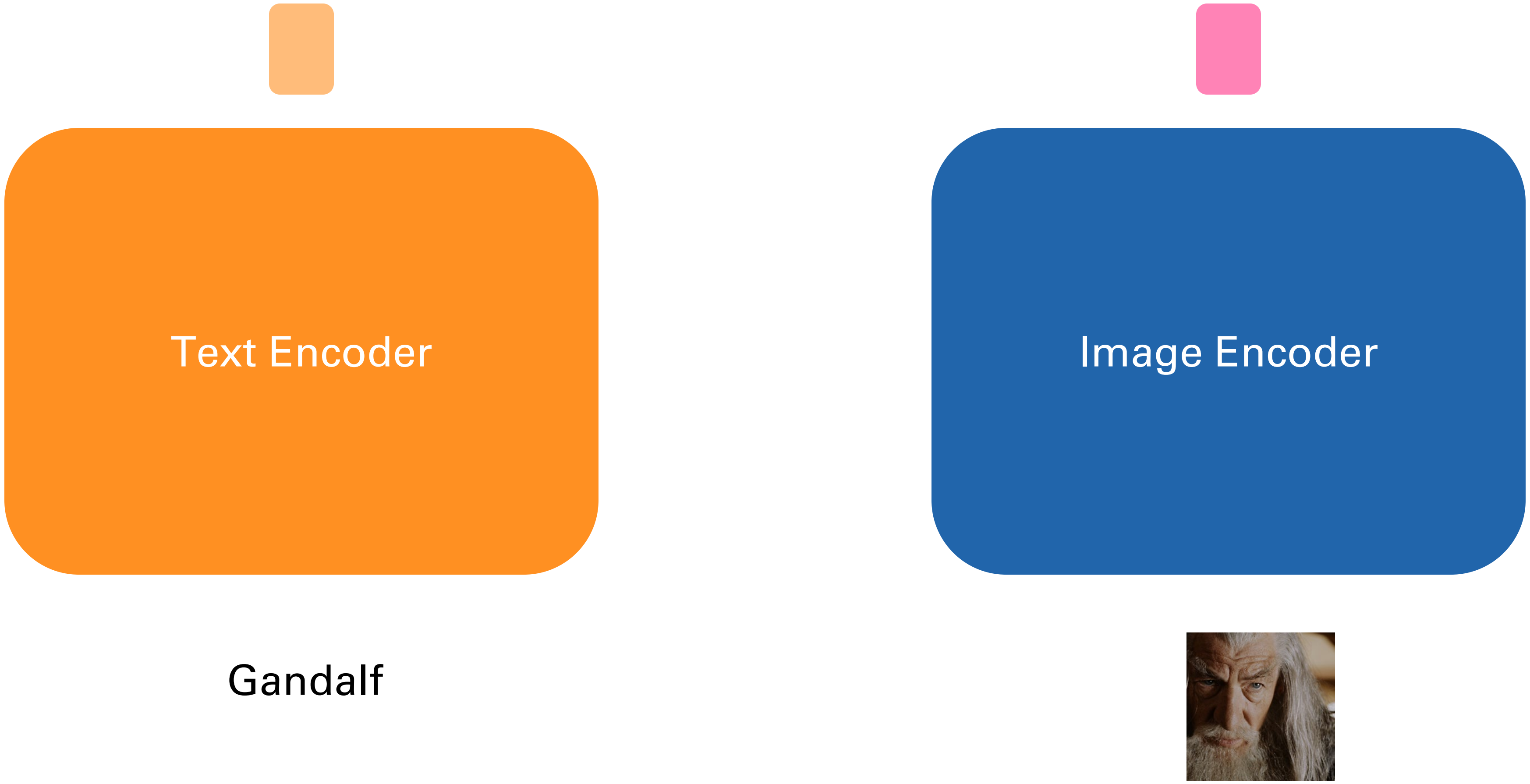
CLIP



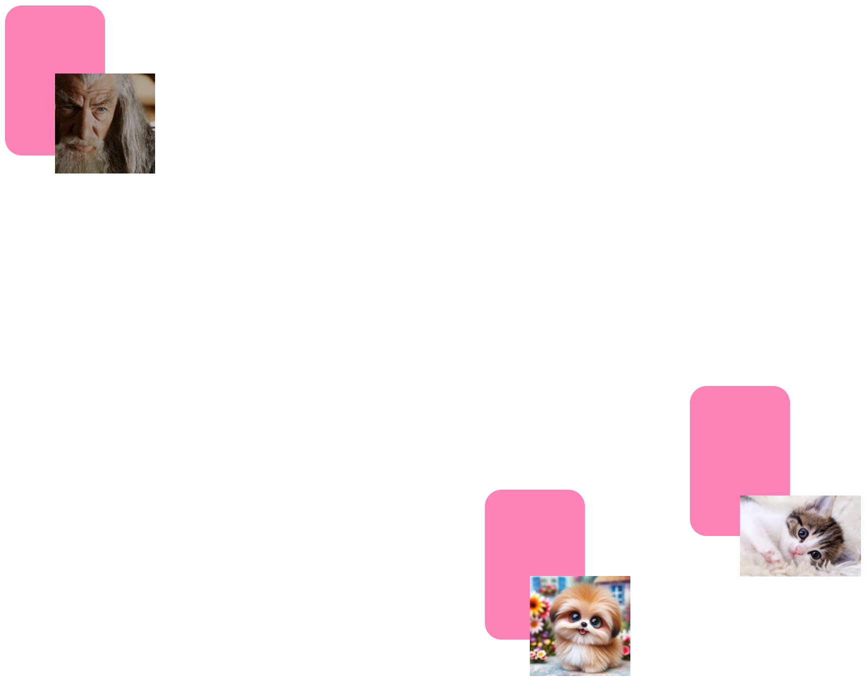
CLIP



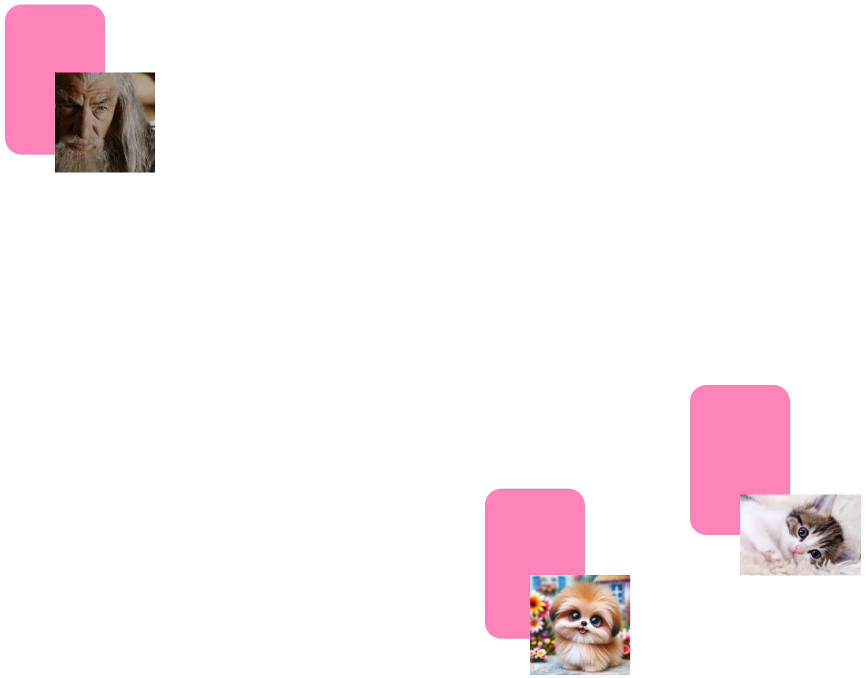
CLIP



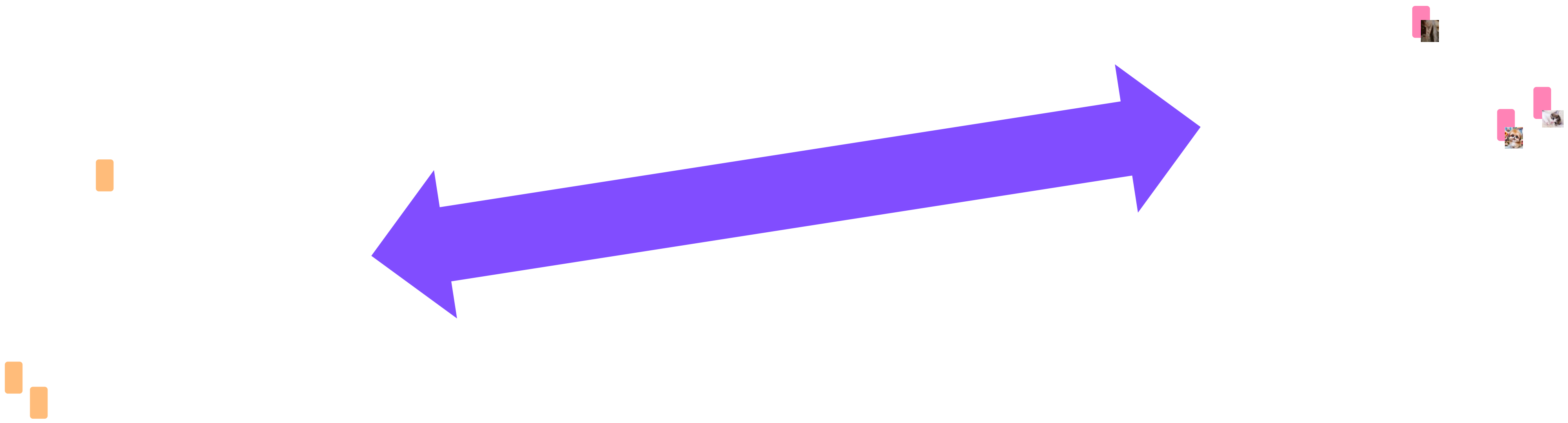
CLIP



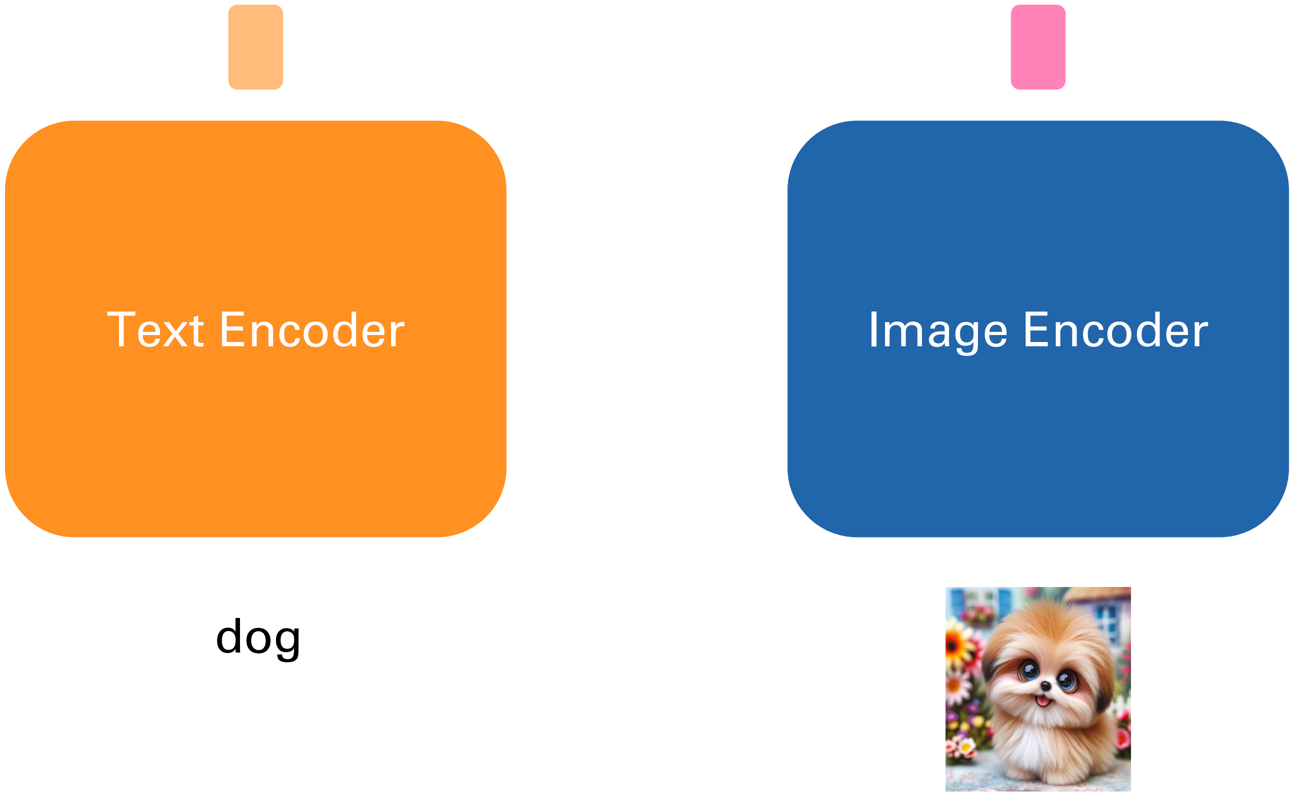
CLIP



CLIP



CLIP



What if we force encoders to have the *same* representation for the same concept?

Contrastive Image Language Pretraining

Pepper the ussie pup

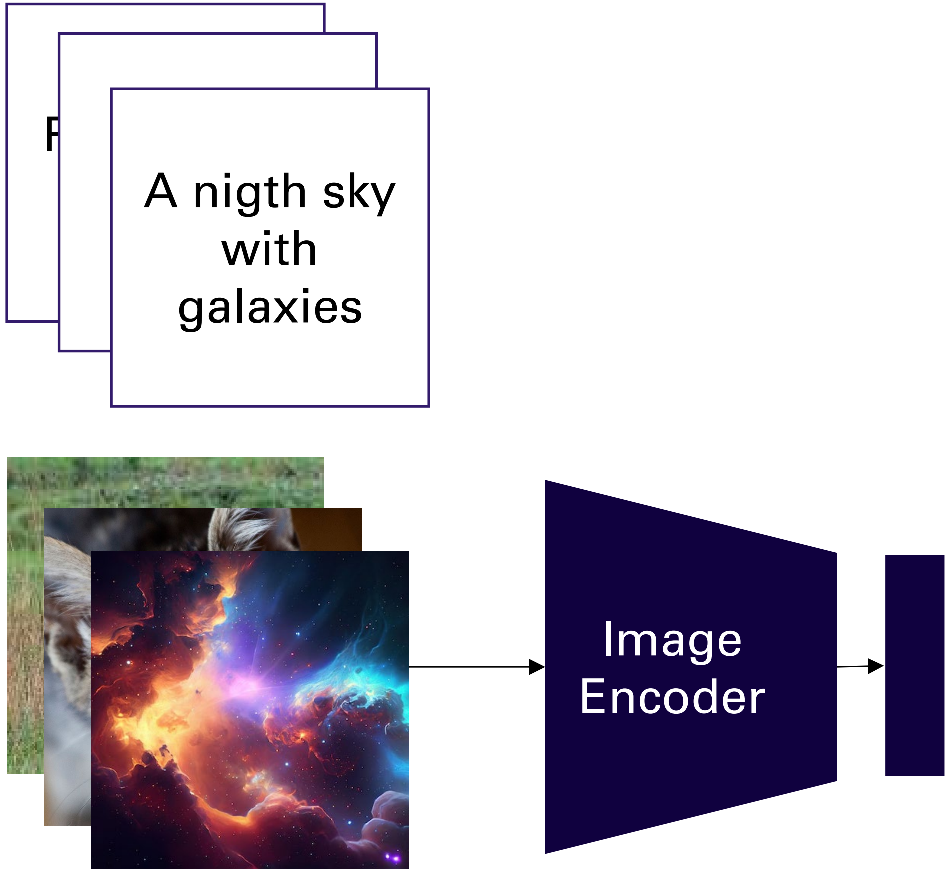
Cute cat looking in camera

A nigth sky with galaxies

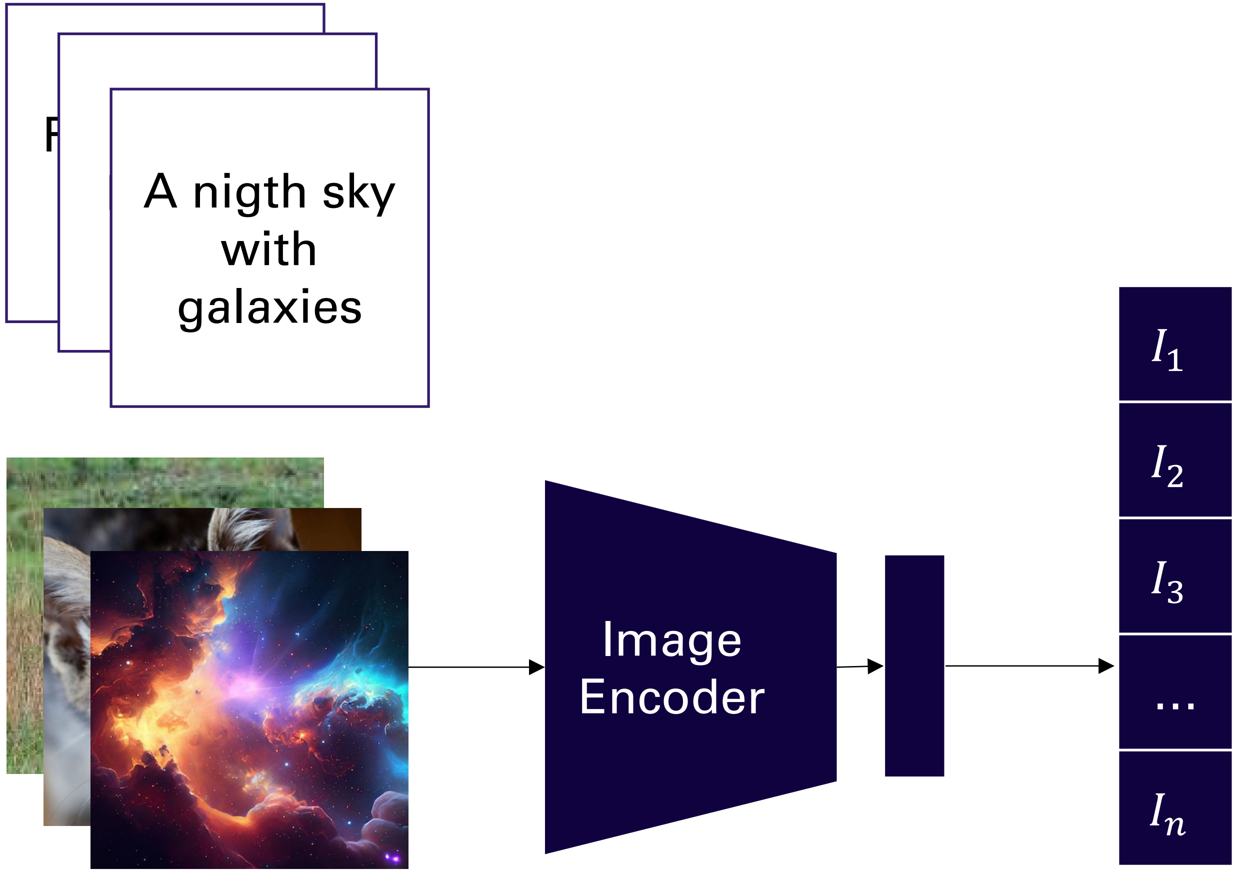
Contrastive Image Language Pretraining

A nigth sky with galaxies

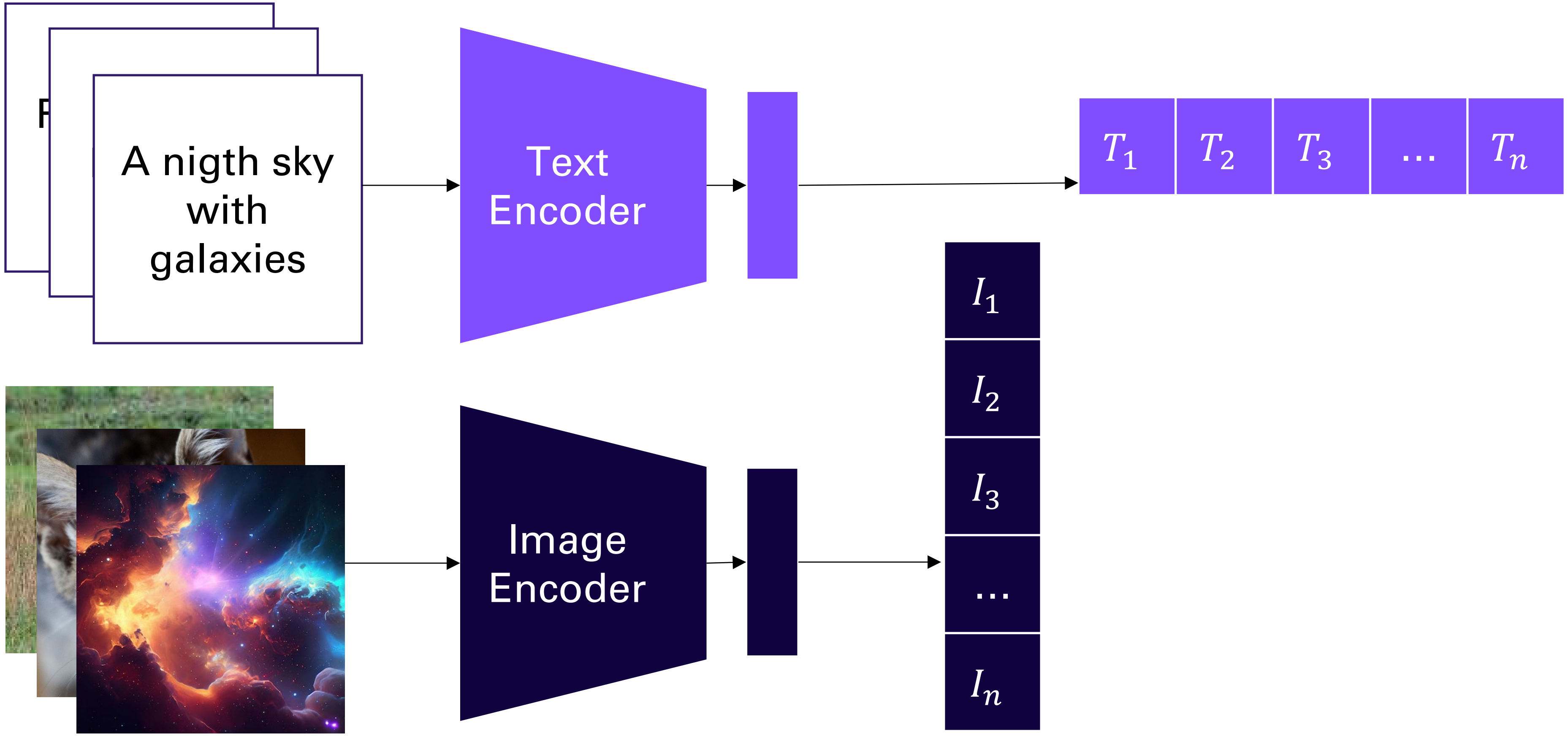
Contrastive Image Language Pretraining



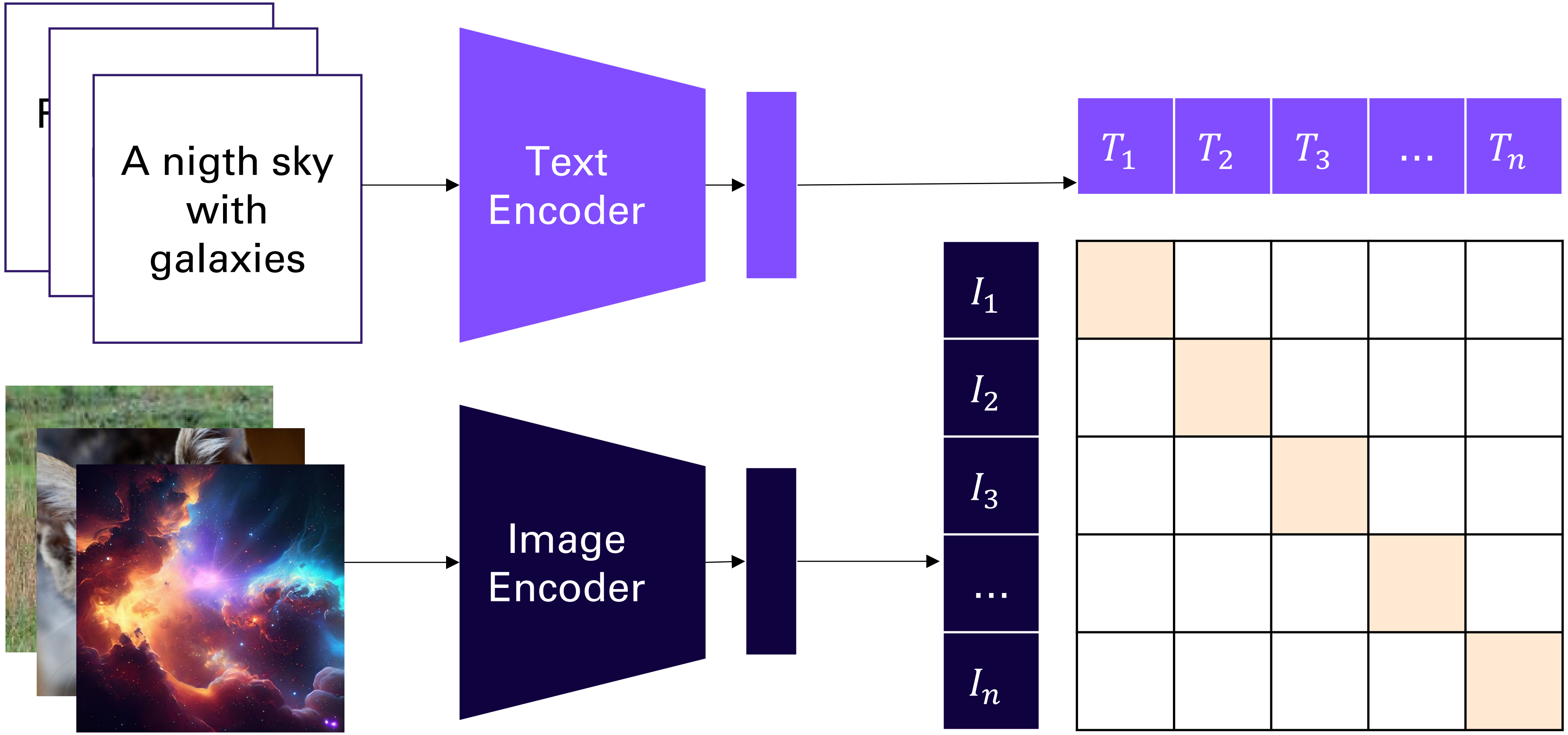
Contrastive Image Language Pretraining



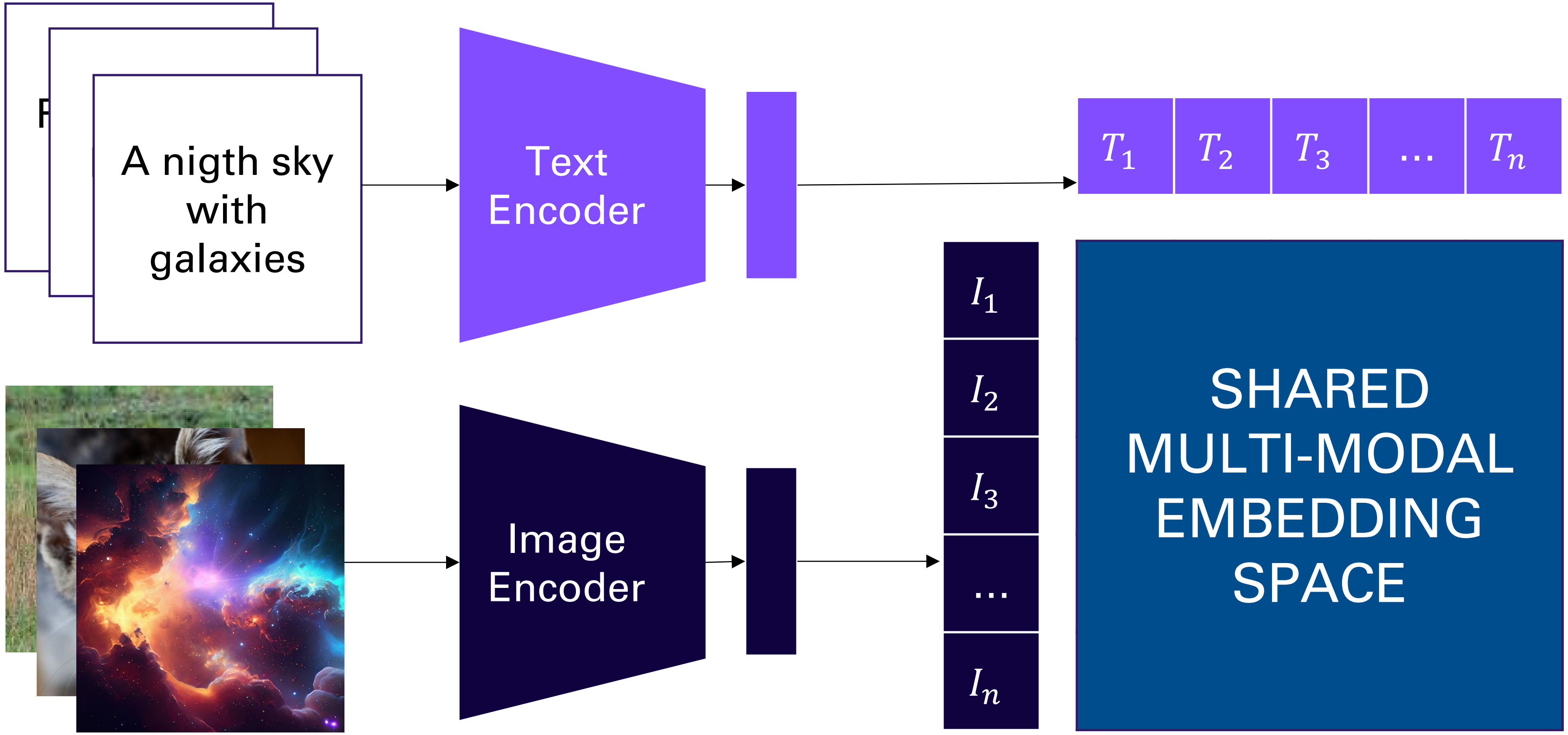
Contrastive Image Language Pretraining



Contrastive Image Language Pretraining



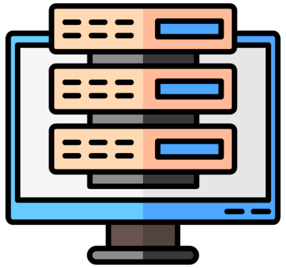
Contrastive Image Language Pretraining



LEARNIG PARADIGMS

Training a Neural Network

Data



Computing Resources

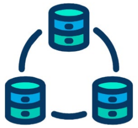
Training a Neural Network

Data

Computing resources

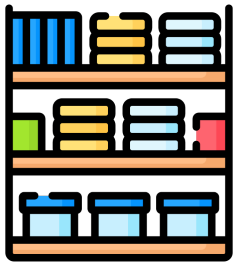
Label Availability (Labeled/ Unlabeled)

Temporal Availability (Offline / Continual)



Data Distribution (Centralized / Federated)

Label Availability



Labeled

Expensive annotation (possibly requiring domain expertise)

Relatively small datasets

High-quality training signal (?)

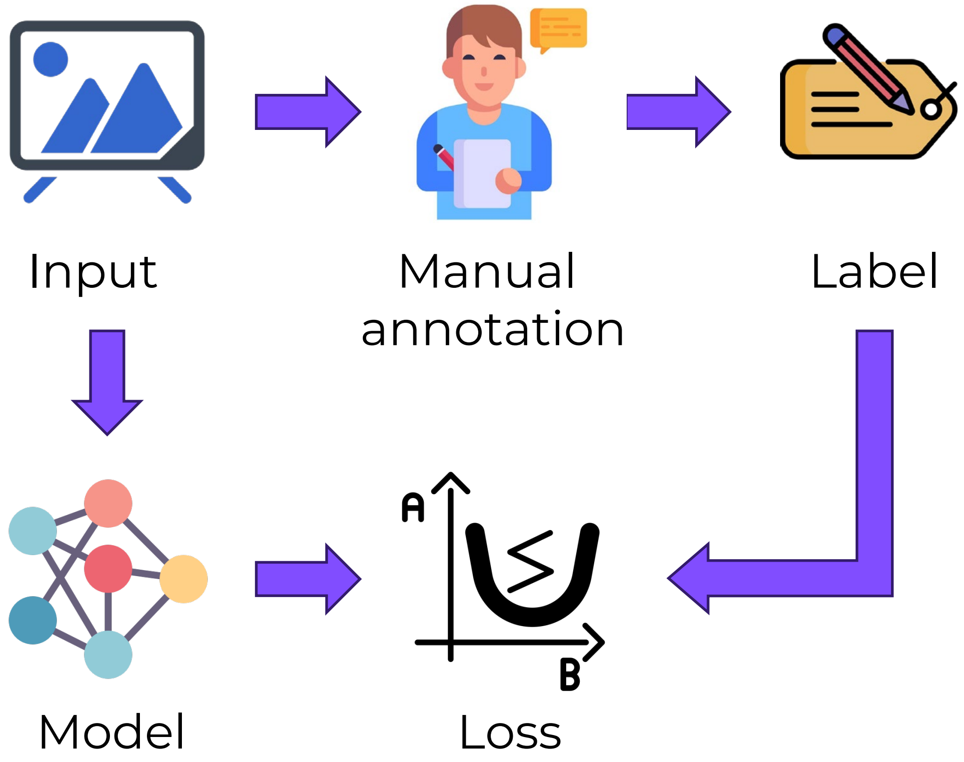
Unlabeled

Easy to collect

Abundant supply

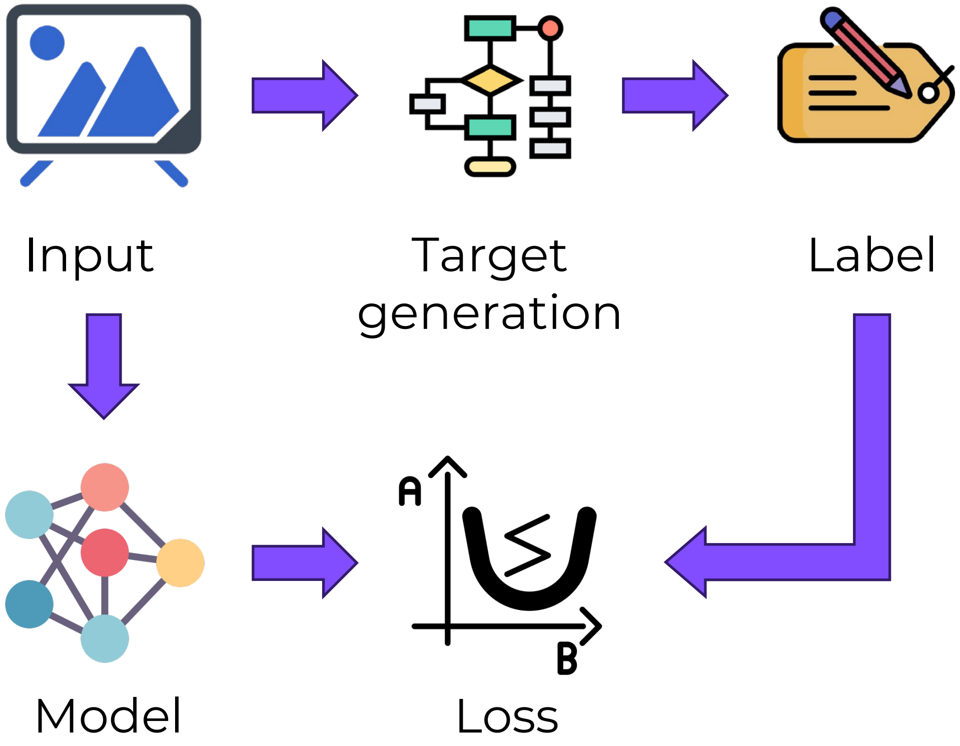
No explicit Supervision

Supervised Learning (SL)



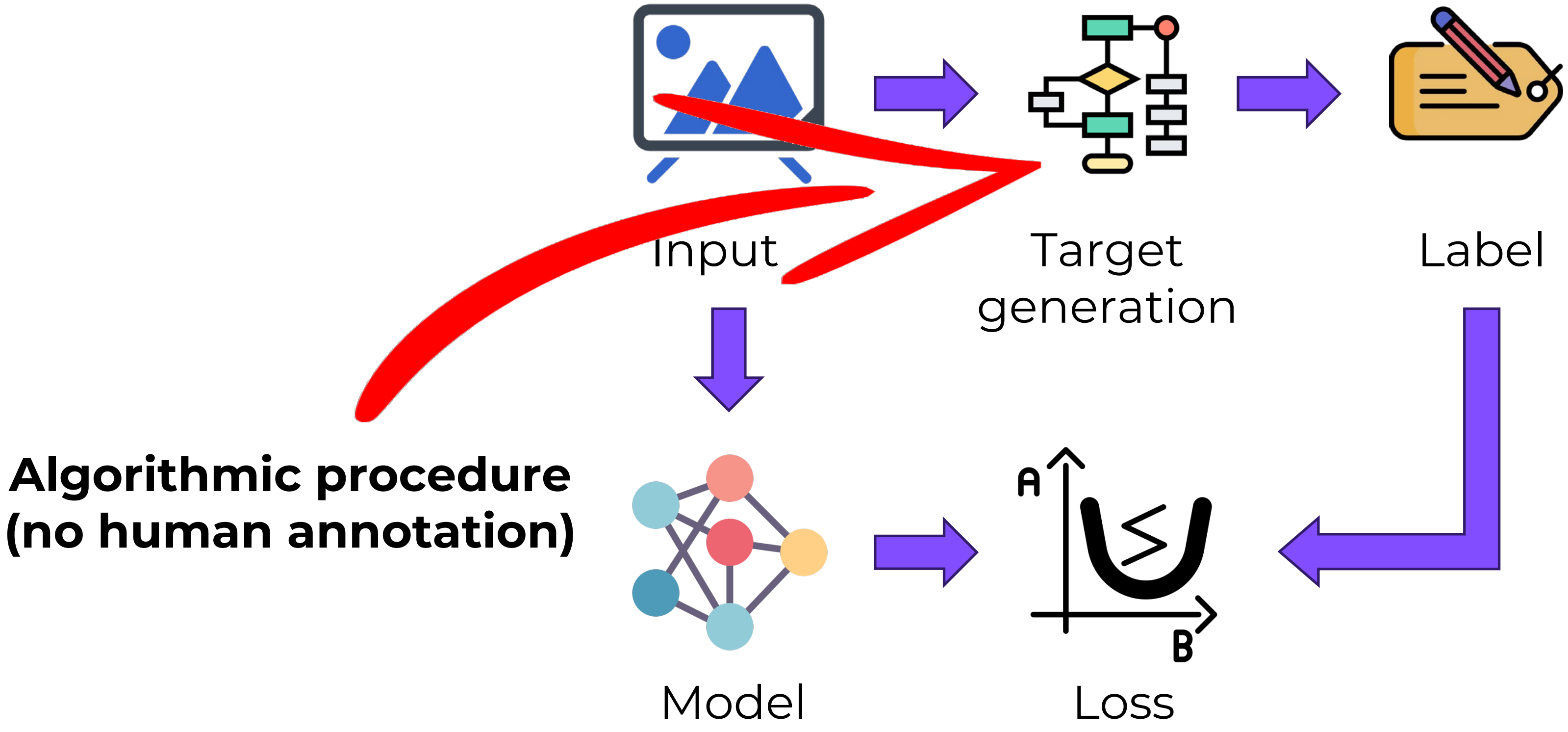
For each observation x_i there is an associated response y_i

Self-Supervised Learning (SSL)



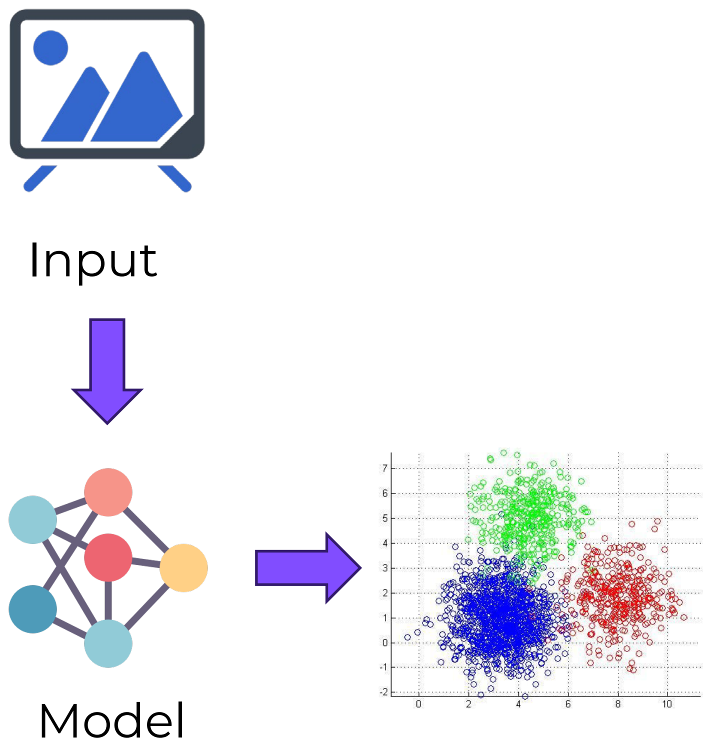
For each observation x_i a response y_i is generated from the data itself

Self-Supervised Learning (SSL)



For each observation x_i a response y_i is generated from the data itself

Unsupervised Learning (UL)



No response y_i is provided: the model learns patterns directly from the input x_i

Self-Supervised vs Unsupervised

- Both assume the lack of manual annotated supervisory signals
- Different objectives:
  - UL: identify patterns in data, usually for clustering, dimensionality reduction, anomaly detection.
  - SSL: learn a data representation that can be transferred to other tasks
- Self-Supervised Learning tends to use loss functions typical of supervised learning (e.g., MSE, NLL)

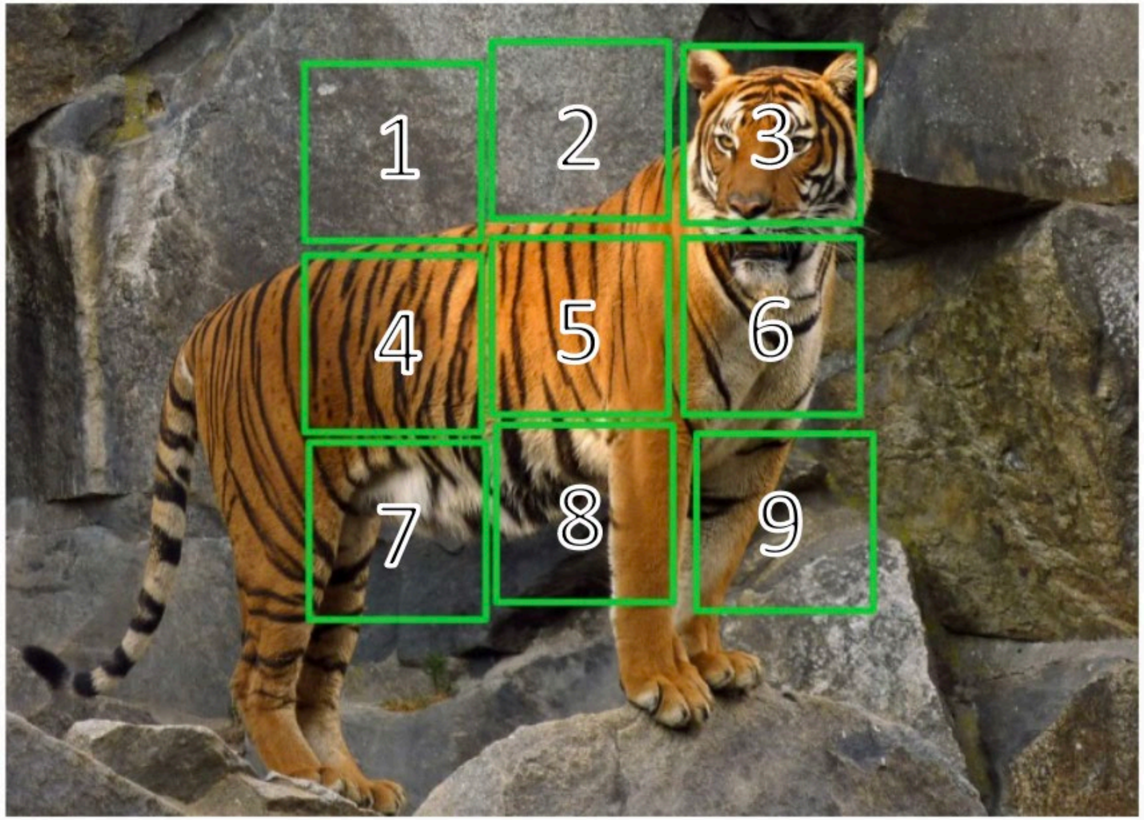
Self-Supervised Learning

- How to generate effective labels?
- **Design a prediction task that requires high-level understanding of the inputs**

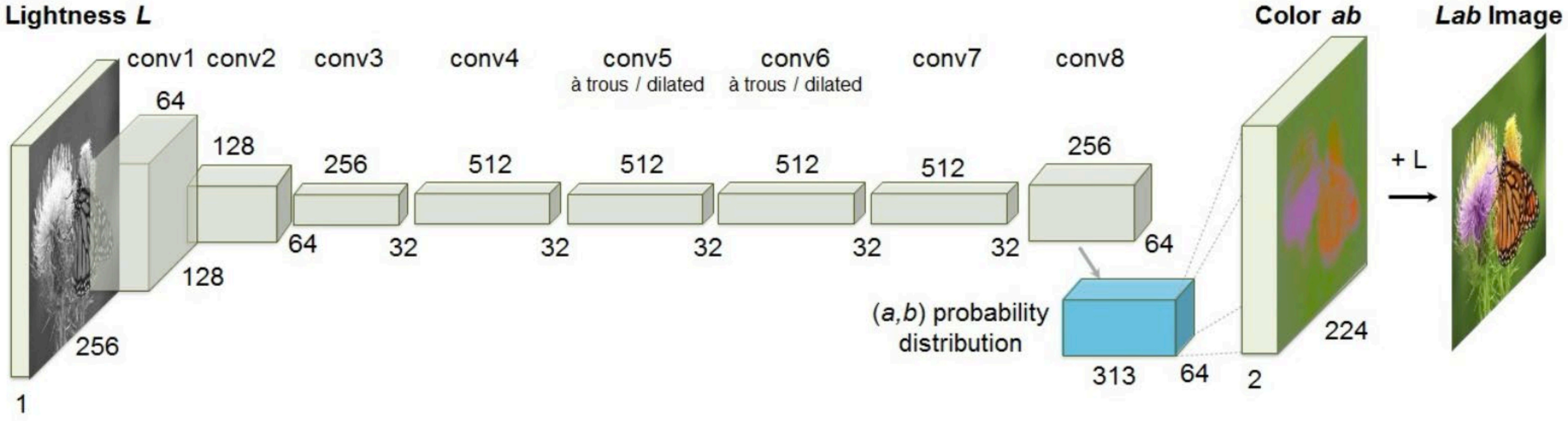
Pretext tasks

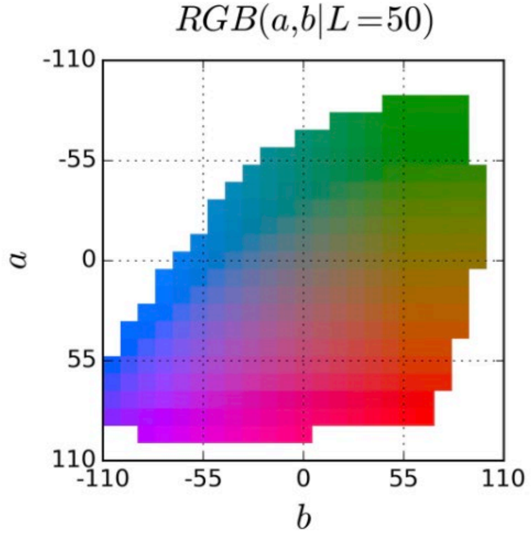
- Hand-craft a task that requires domain knowledge to be solved
- Generally posed as a classification problem

Jigsaw puzzle



Colorization

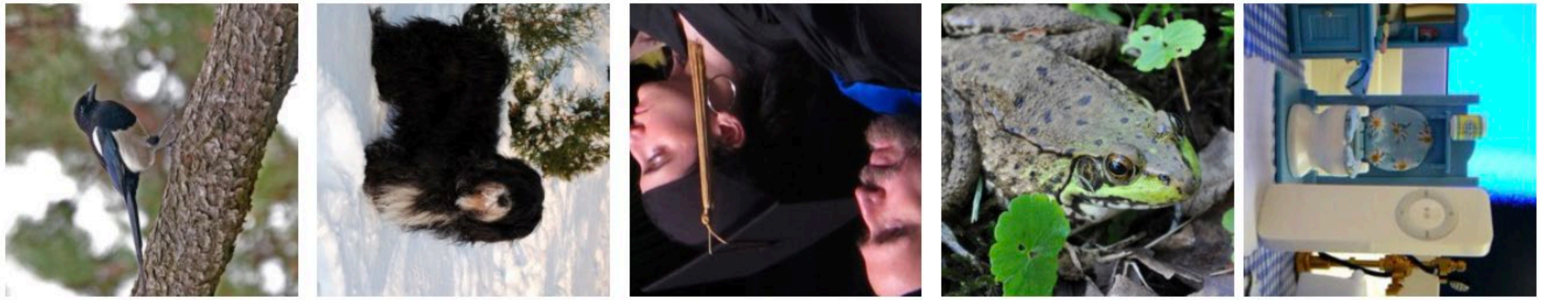




313 classes

Rotation

input

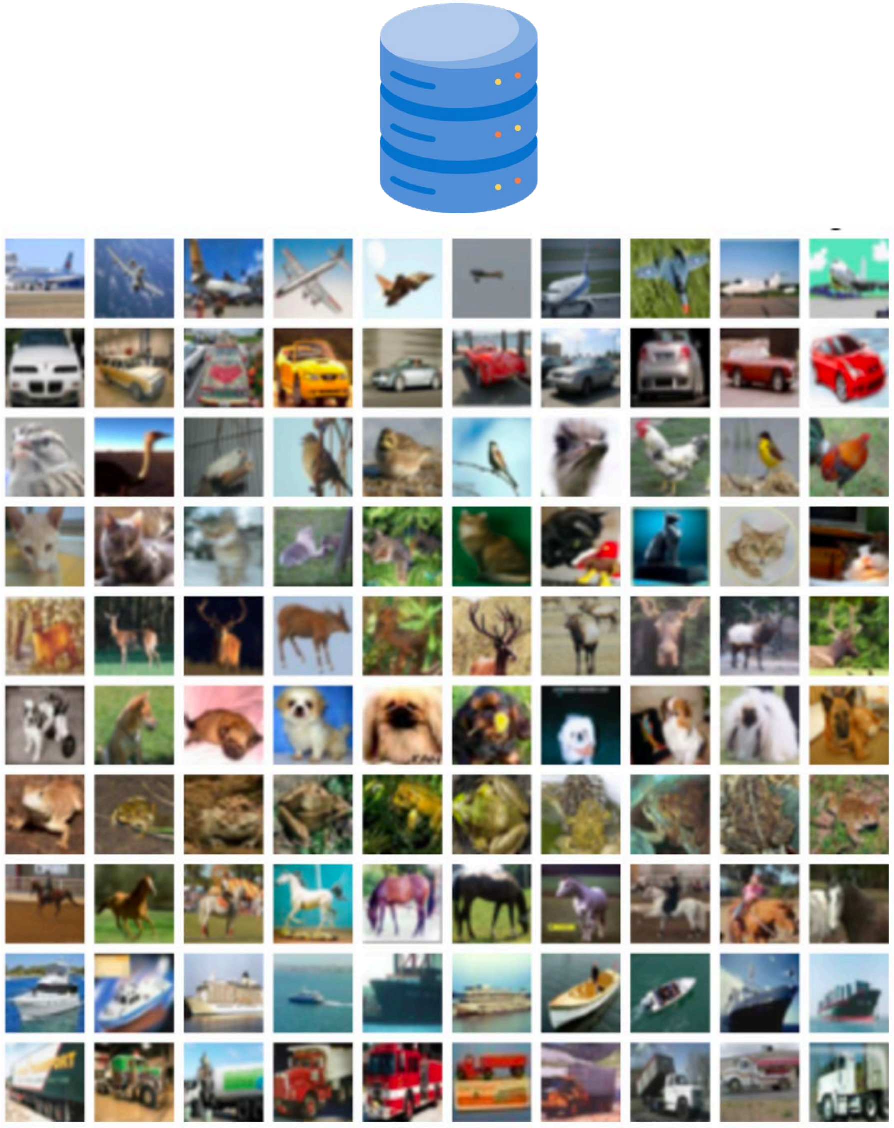


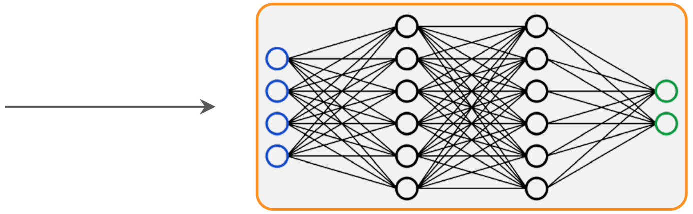
label

90° rotation | 270° rotation | 180° rotation | 0° rotation | 270° rotation

4 classes

Temporal Availability

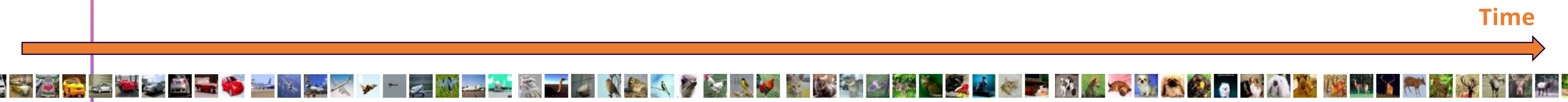




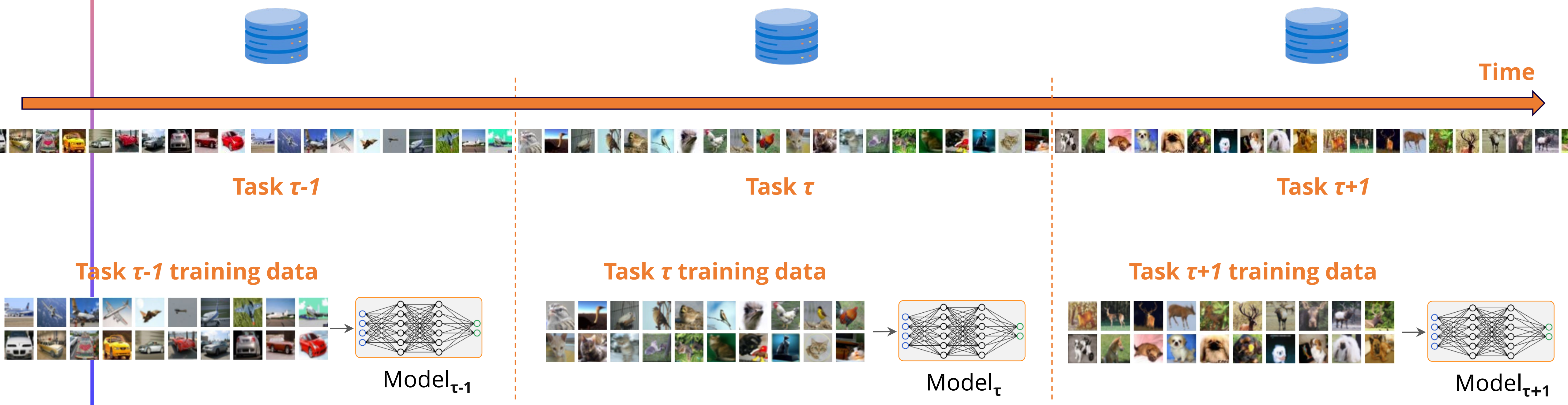
Offline Learning

All data is available at once

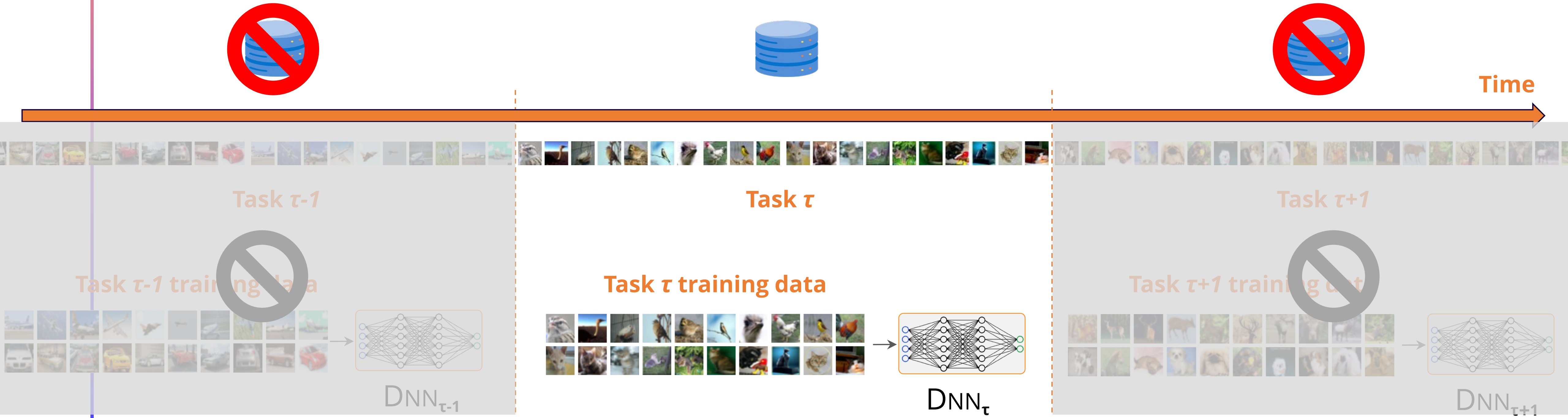
Temporal Availability



Temporal Availability



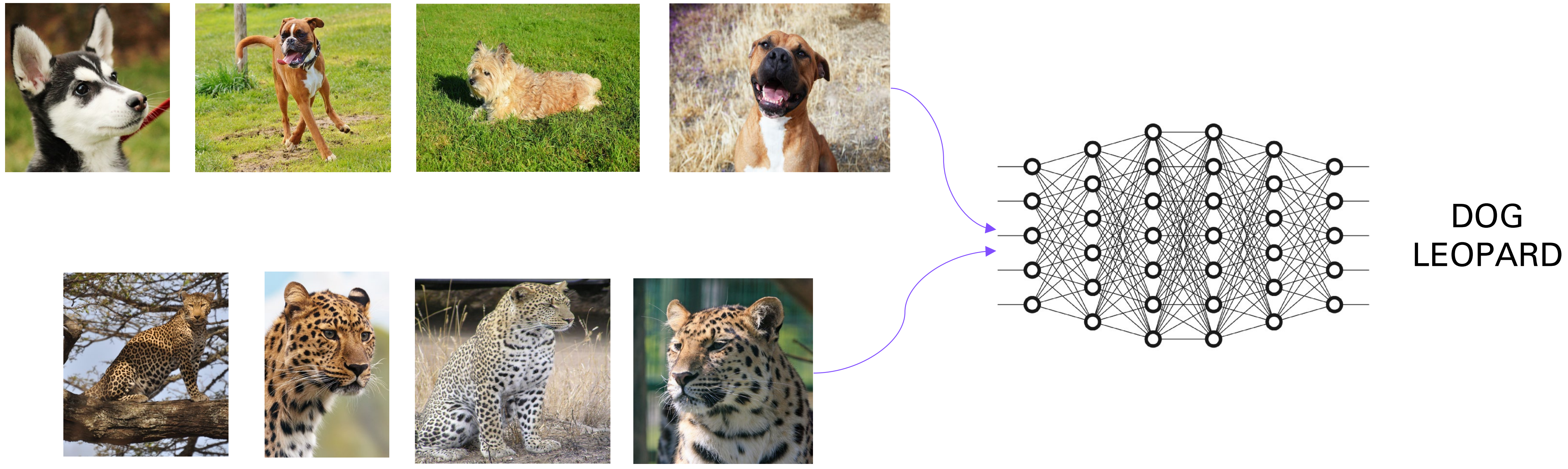
Temporal Availability



Continual Learning

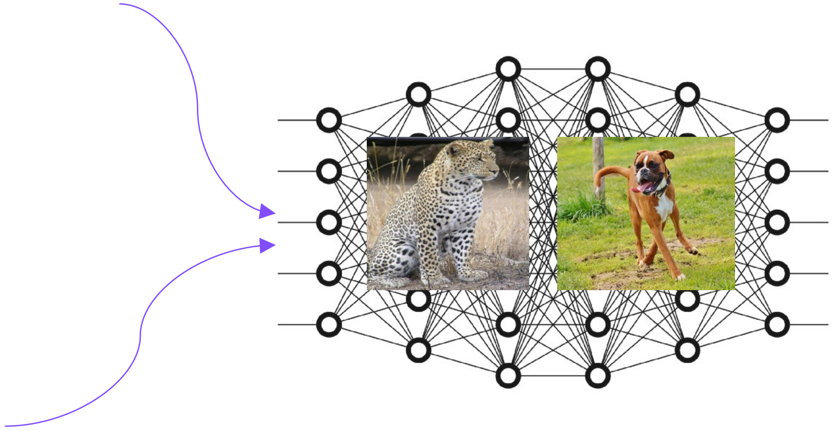
Data arrives in a stream and cannot be stored

Continual Training



Task τ=1

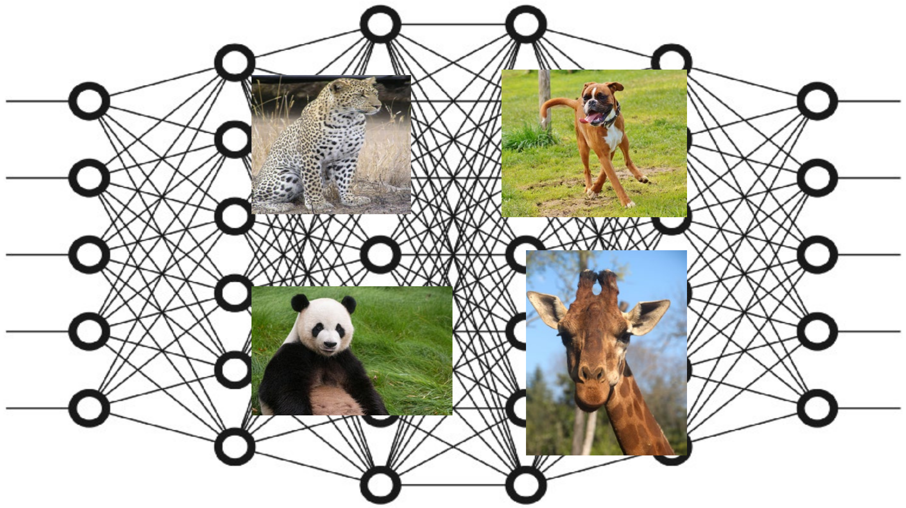
Continual Training



DOG
LEOPARD
PANDA
GYRAFFE

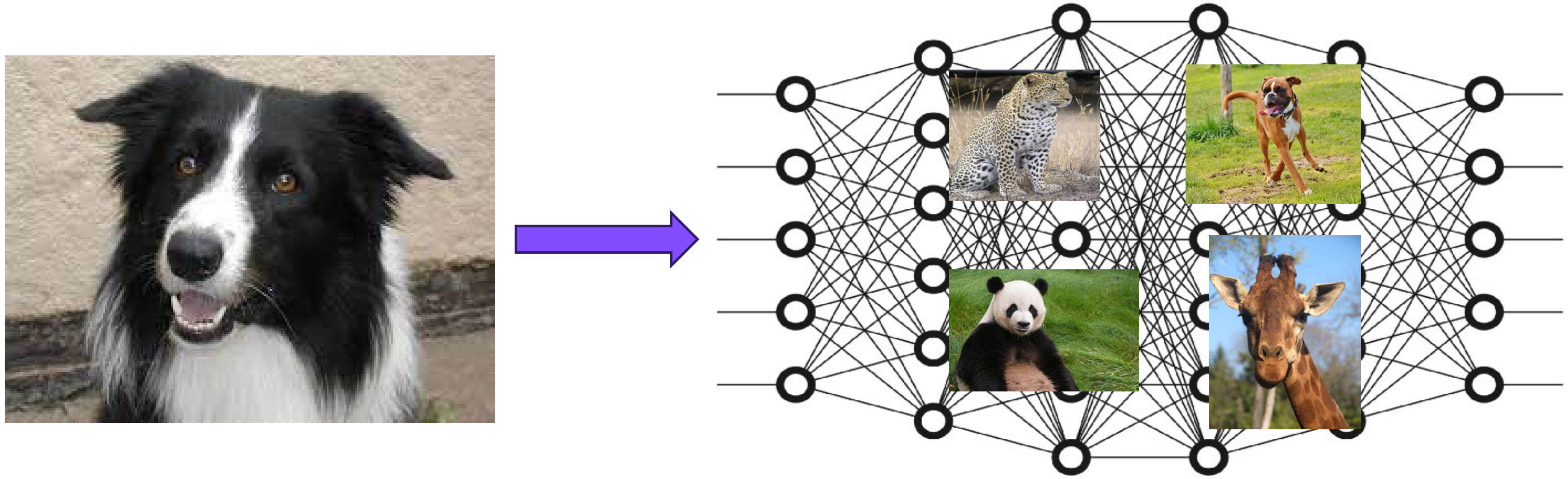
Task τ=2

Continual Training



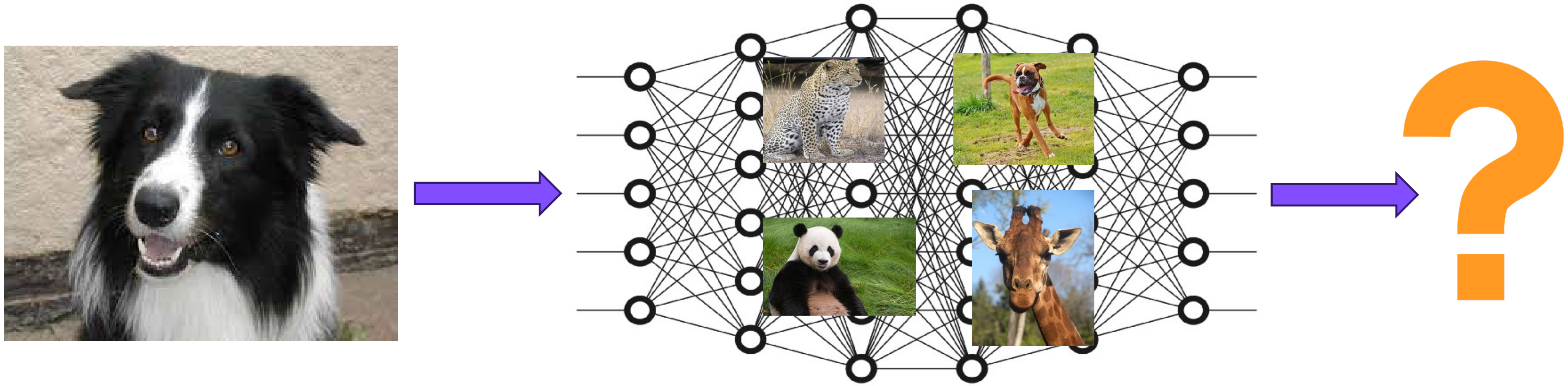
Trained model

Continual Training



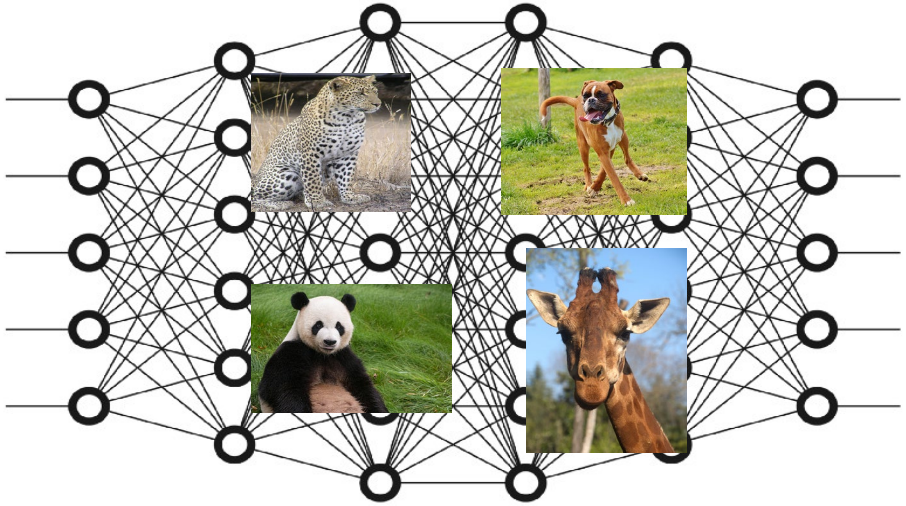
Trained model

Continual Training



Trained model

Continual Training



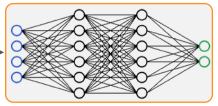
It’s a Panda!

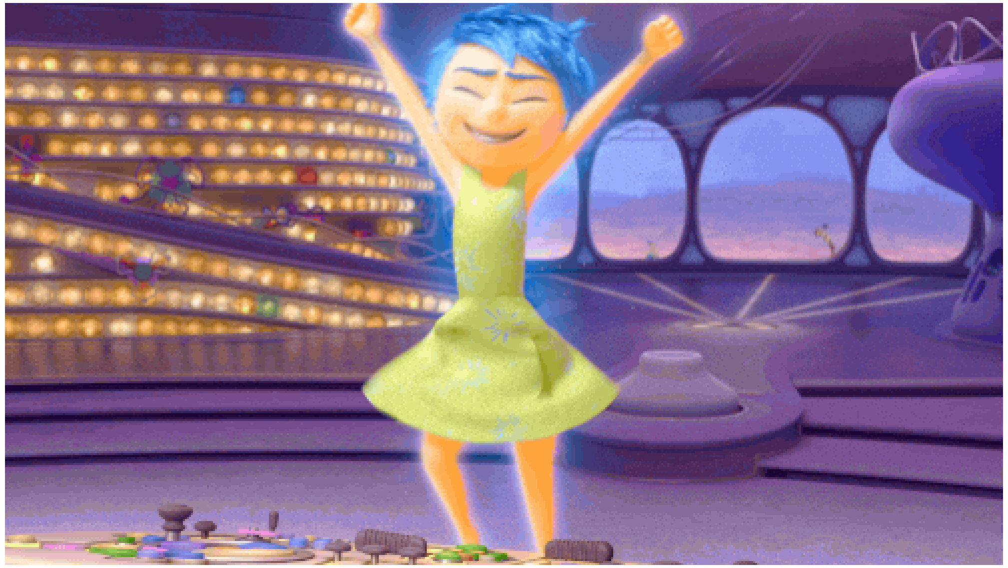
Trained model

Continual Training

Offline learning

Evaluation after training:

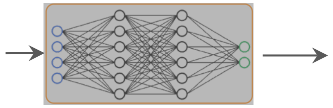
→  → ***93.25 % Accuracy***



Continual Training

Offline learning

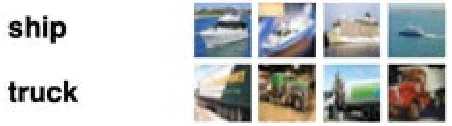
Evaluation after training:

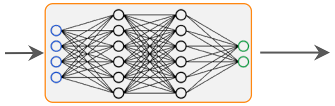


93.25 % Accuracy

Continual learning

Evaluation after τ tasks:



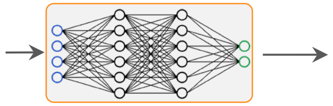


19.62 % Accuracy

Continual Training

Isolated learning

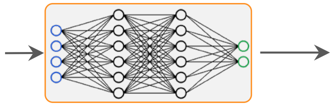
Evaluation after training:



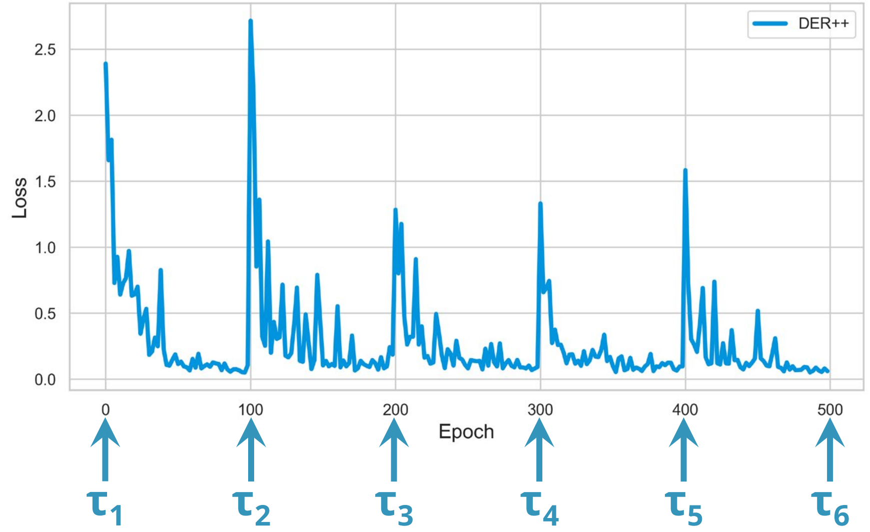
93.25 % Accuracy

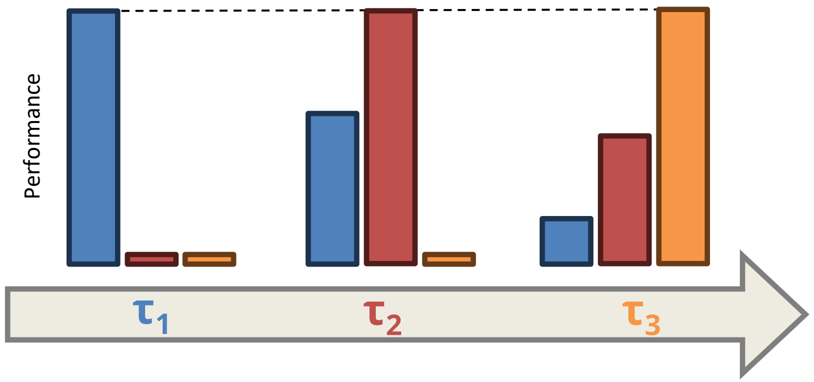
Continual learning

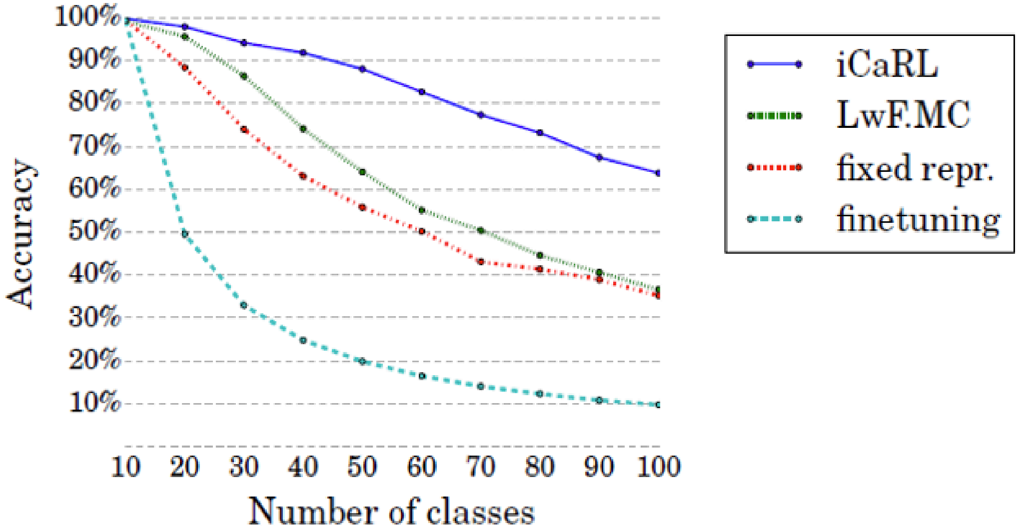
Evaluation after τ tasks:



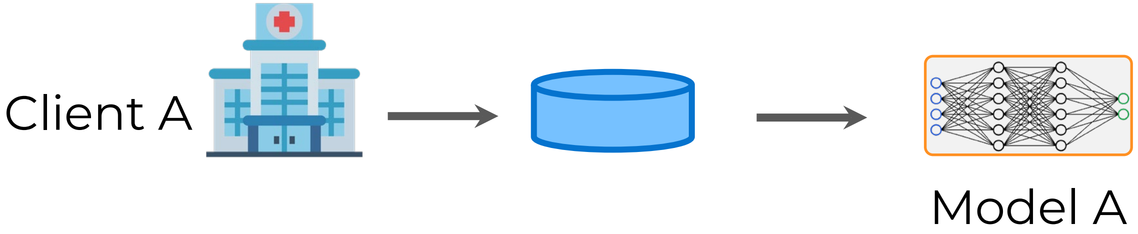
19.62 % Accuracy





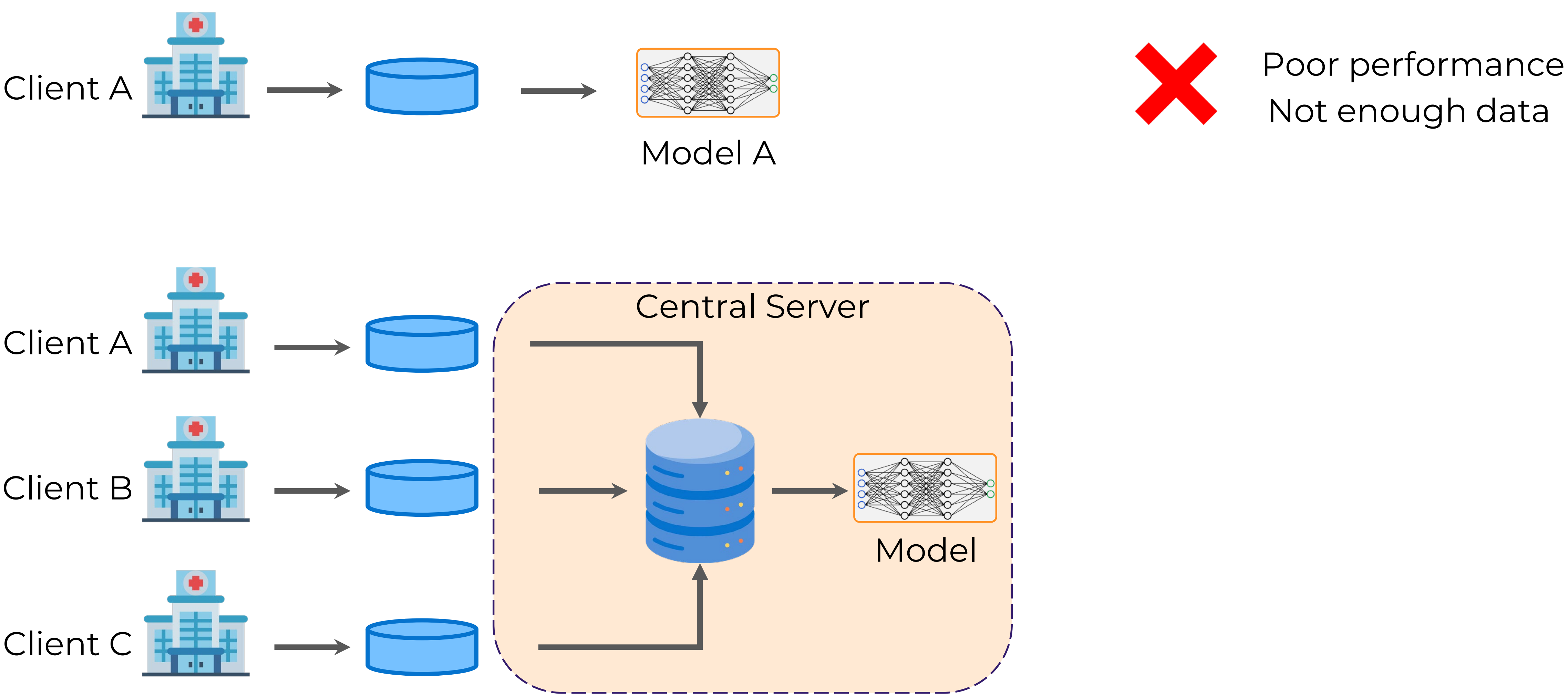


Data Distribution

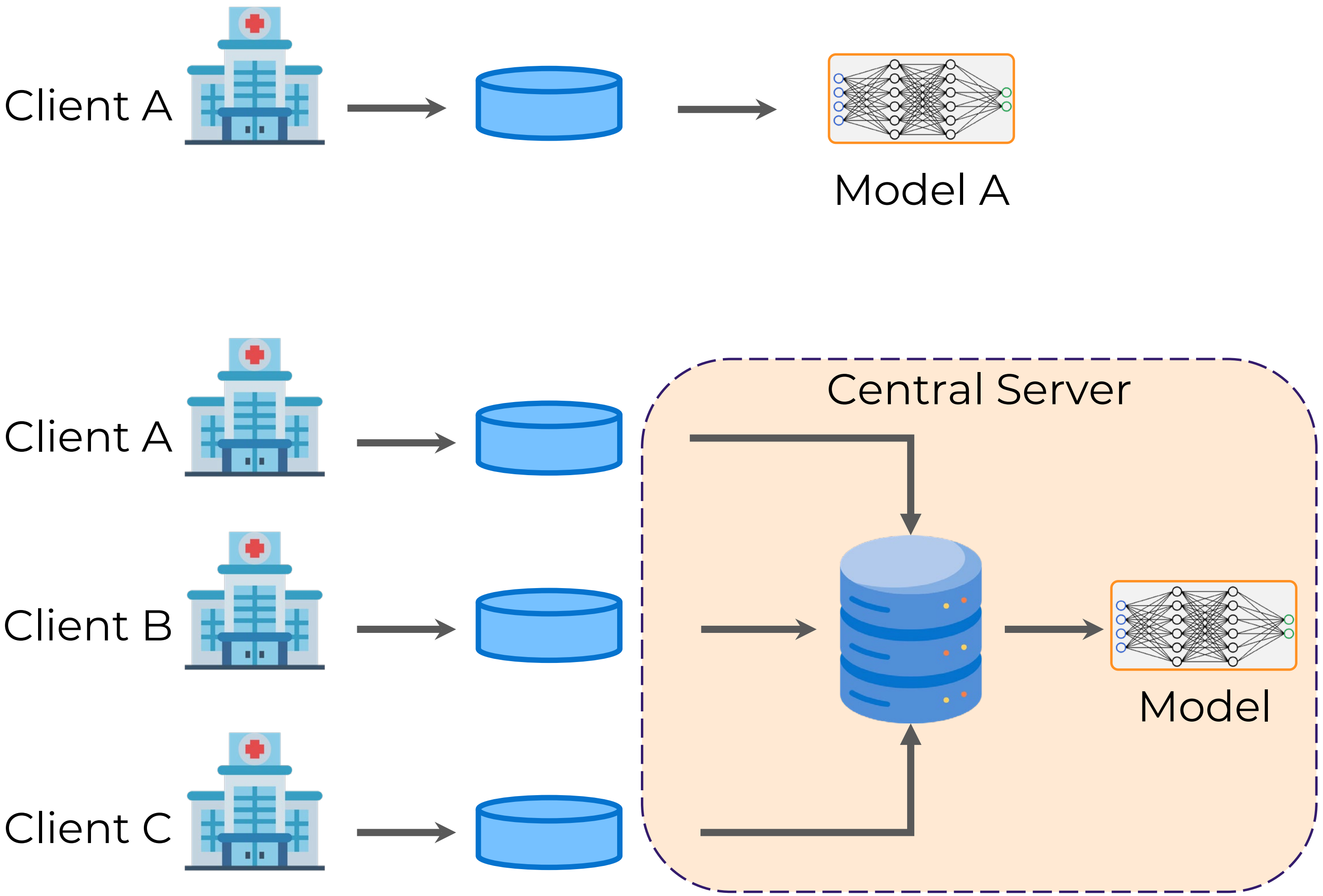


Poor performance
Not enough data

Data Distribution



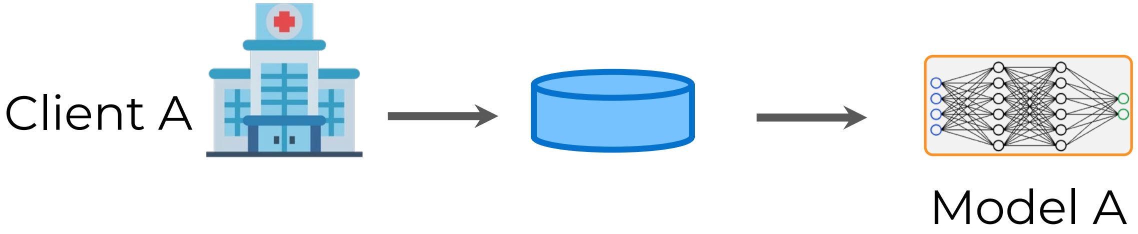
Data Distribution



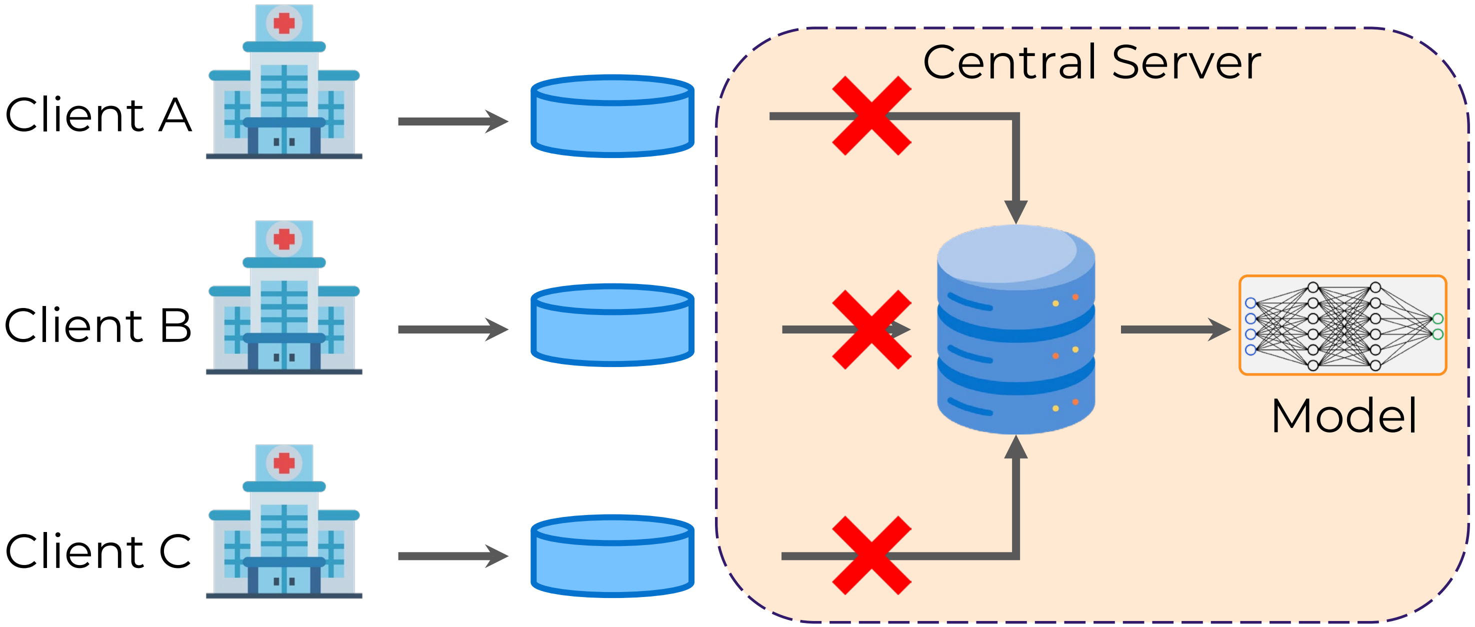
Poor performance
Not enough data

Superior results

Data Distribution



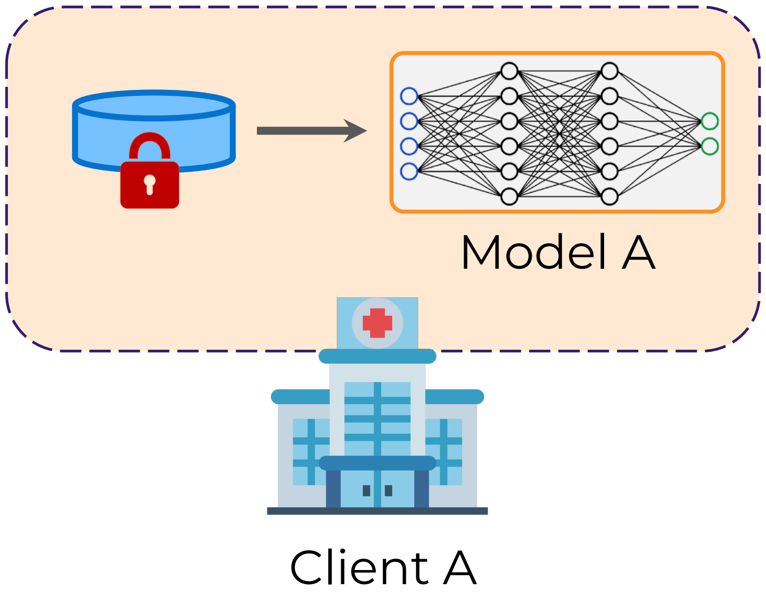
Poor performance
Not enough data



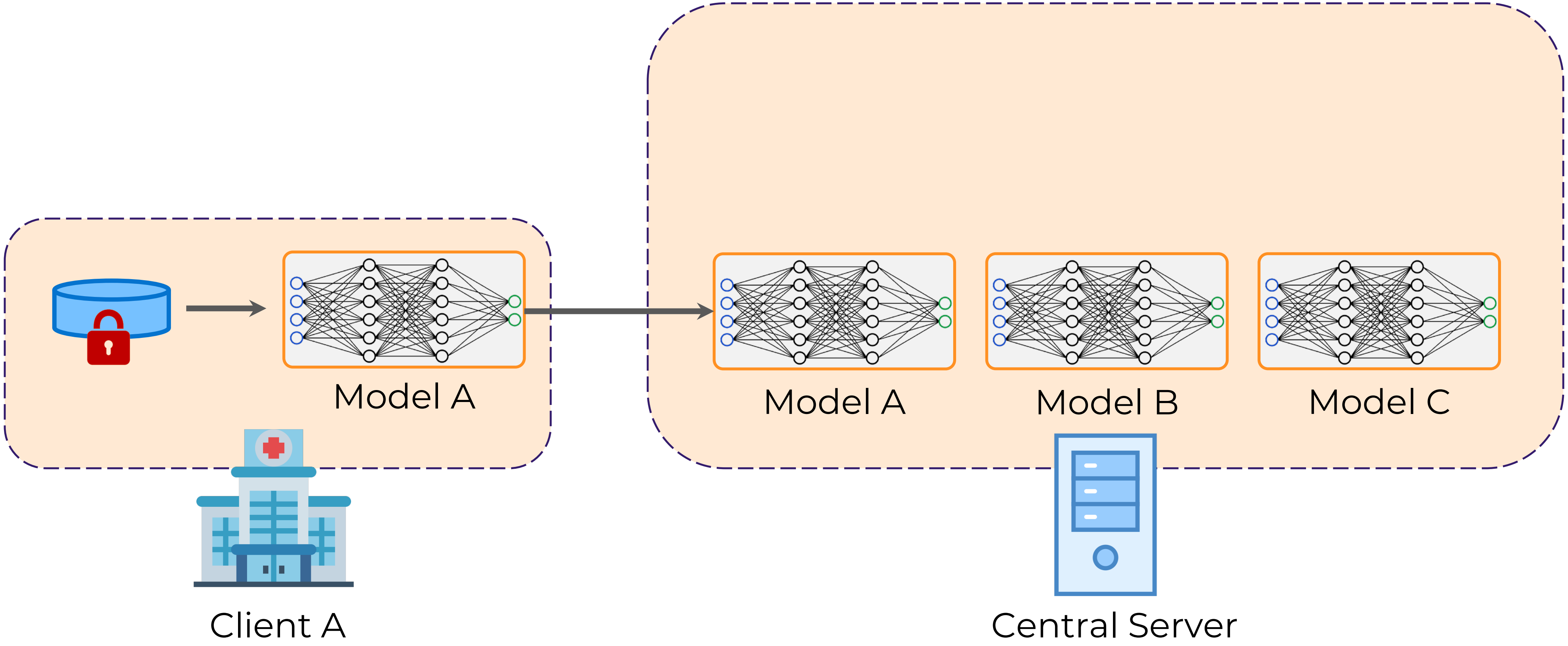
Superior results

Privacy breach

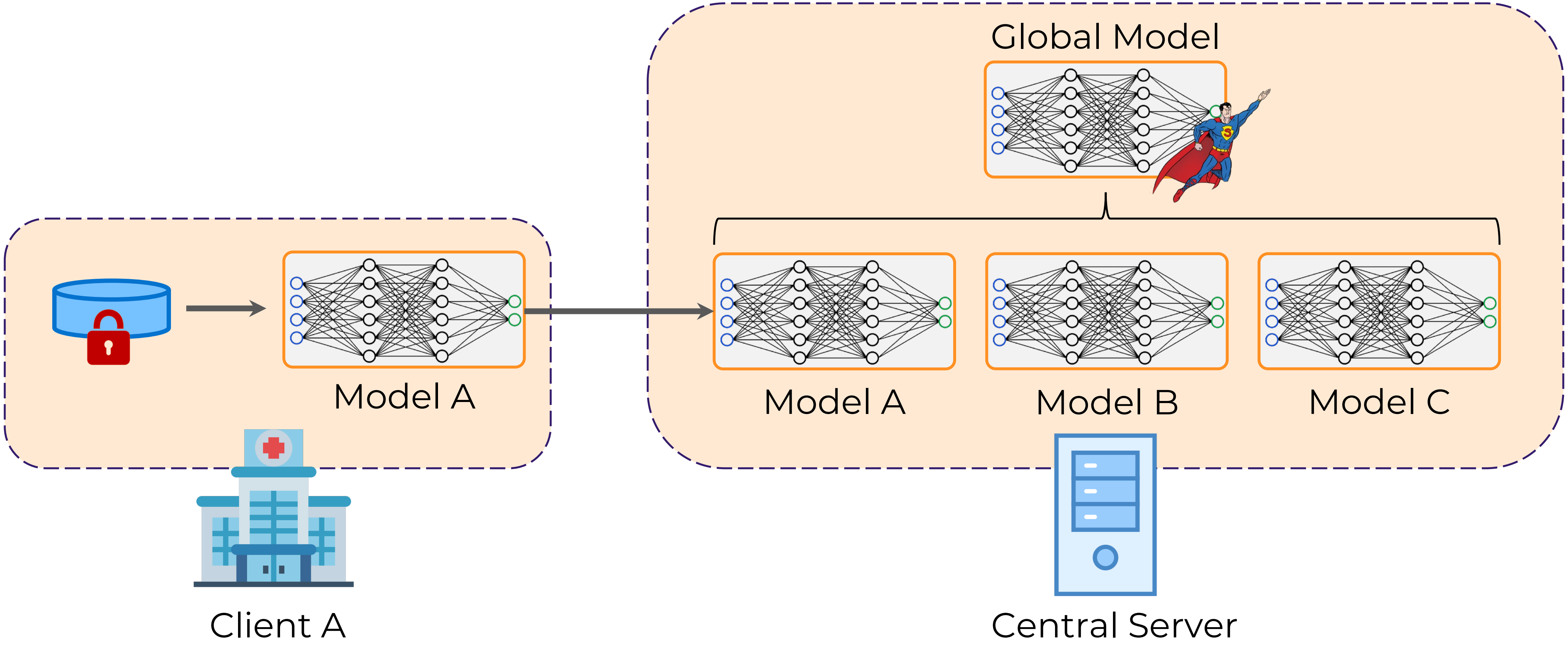
Data Distribution



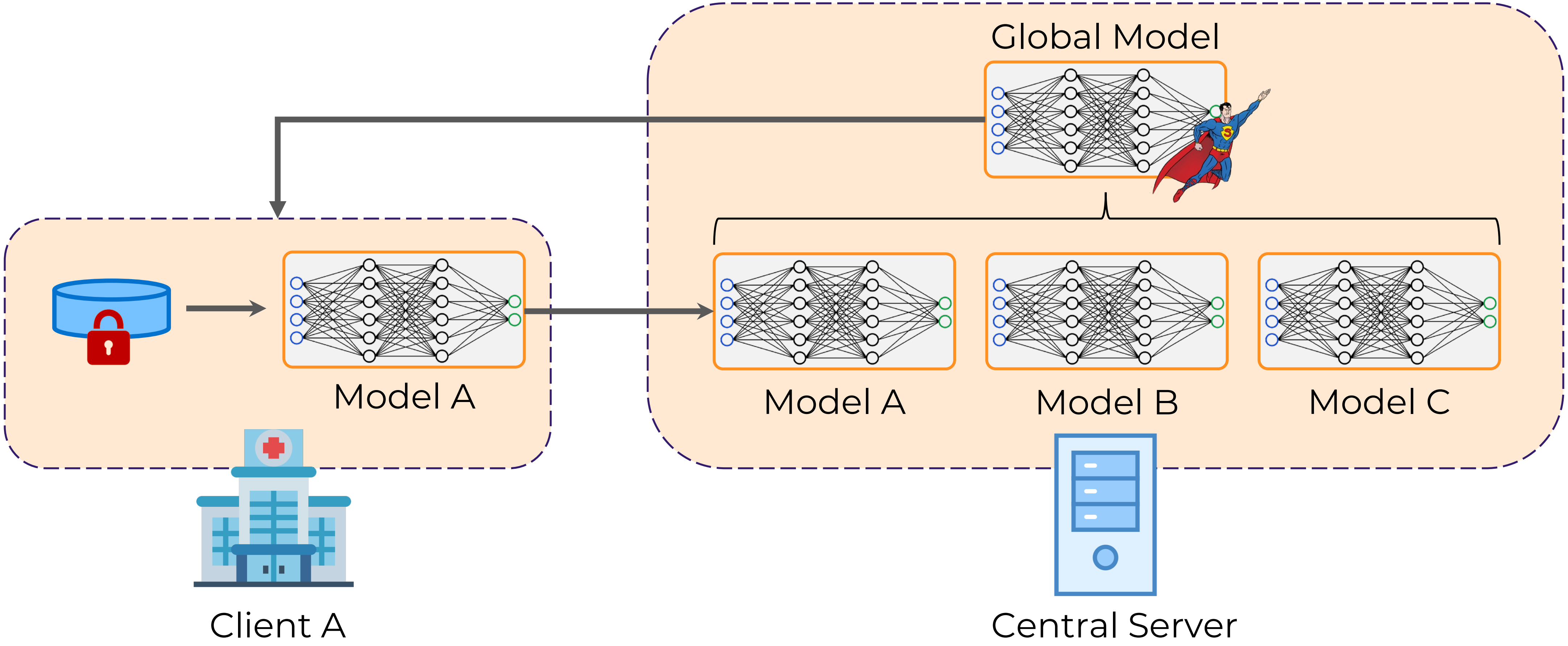
Data Distribution



Data Distribution



Data Distribution



Federated Learning

Data remains distributed across multiple clients, only model updates are shared

PERCEIVE.AI LAB

University of Catania

simone.palazzo@unict.it
giovanni.bellitto@unict.it
federica.proiettosalanitri@unict.it
matteo.pennisi@unict.it