Vision-Language Models
Part I:
Representation learning
CLIP

Introduction

From zero-shot Transfer to representation learning

Remember: Transfer Learning

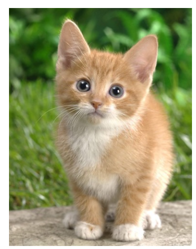
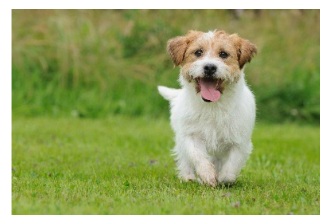
		Source Data (not directly related to the task)	
		labelled	unlabeled
Target Data	labelled	Fine-tuning *Multitask Learning*	Not considered here
	unlabeled	Domain adaptation-adversarial training ***Zero-shot learning***	Not considered here

Zero-shot Learning

- Source data: (x^s, y^s) ➡ Training data
- Target data: $(\emptyset)$ usually same domain

Different tasks

Training time :

x^s:   …

y^s: cat dog …

+ Class Information

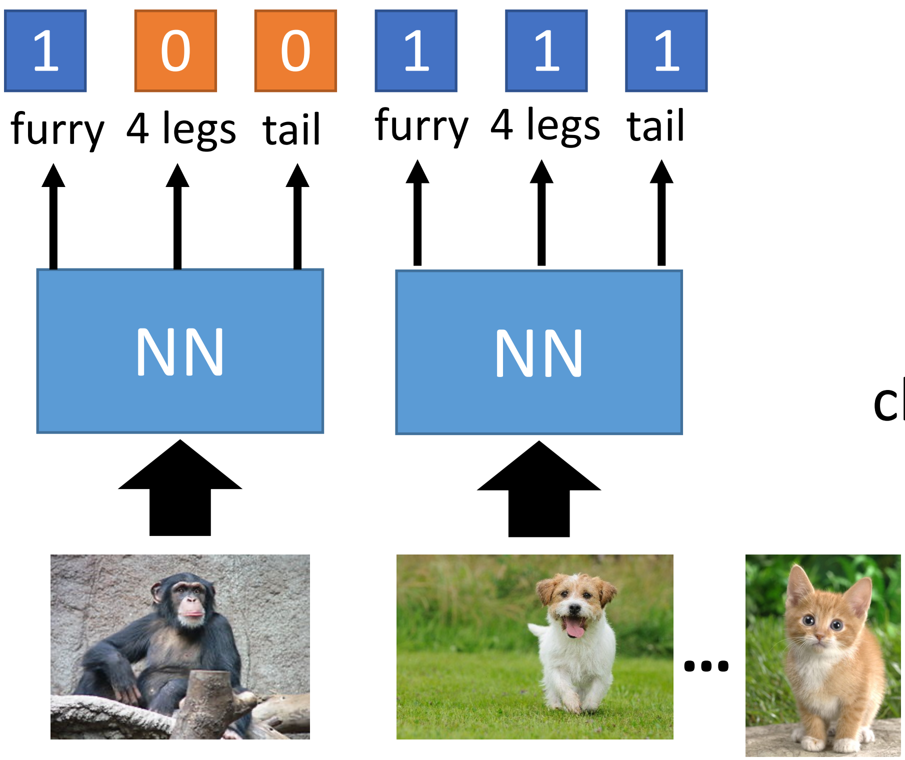
Test time x^t: 

=> Fish class!

Zero-shot Learning

- Representing each class by its attributes

Training



+
Database attributes

class

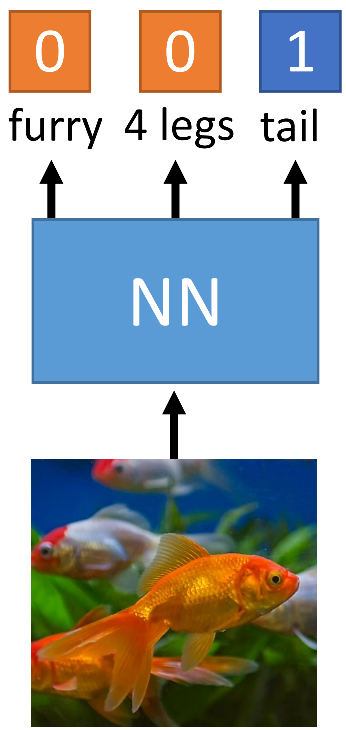
	furry	4 legs	tail	...
Dog	O	O	O	
Fish	X	X	O	
Chimp	O	X	X	
...				

sufficient attributes for one to one mapping

Zero-shot Learning

- Representing each class by its attributes

Testing



Find the class with the most similar attributes

class

attributes

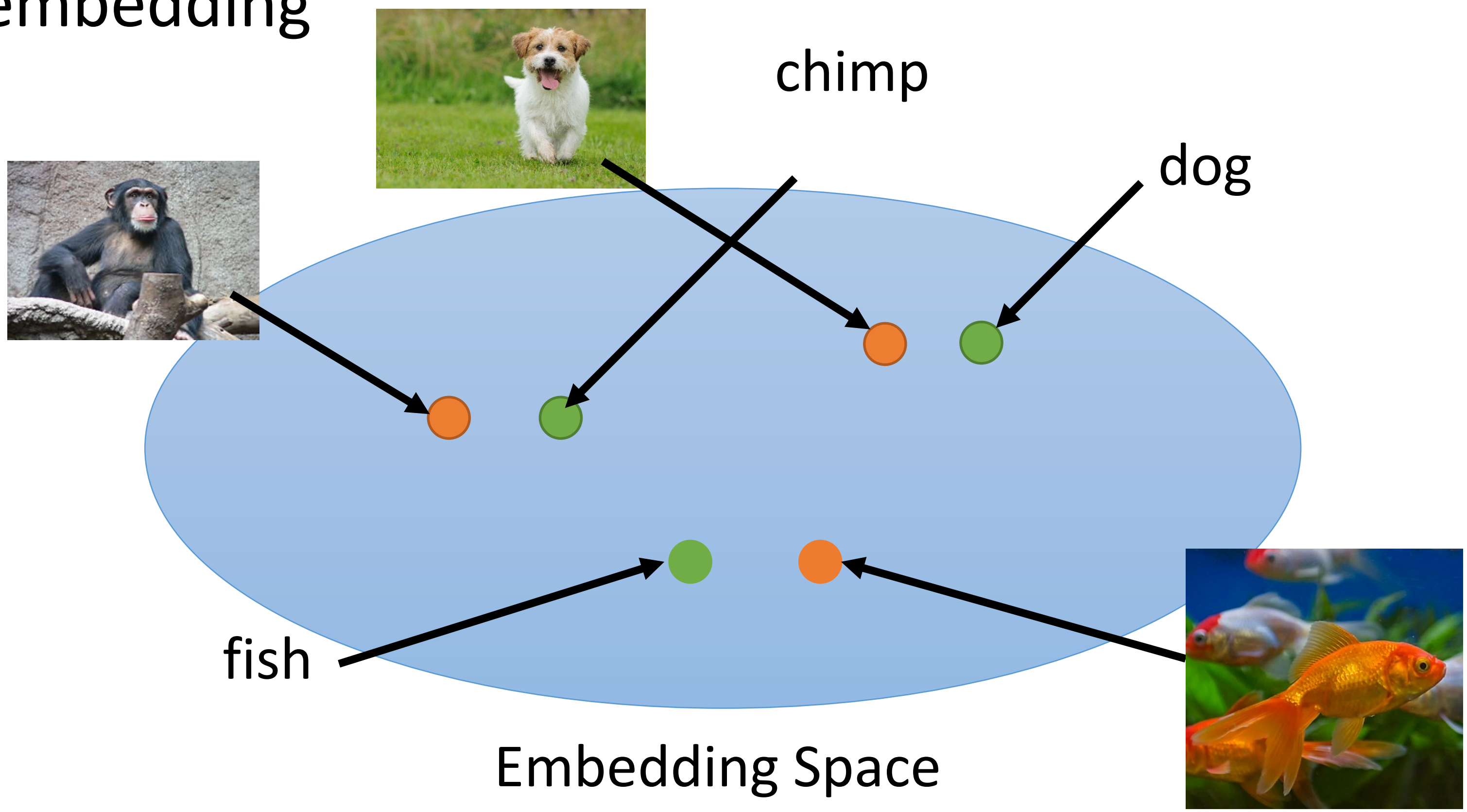
	furry	4 legs	tail	…
Dog	O	O	O	
Fish	X	X	O	
Chimp	O	X	X	
…				

sufficient attributes for one to one mapping

Zero-shot Learning

What if we don't have attribute database

- Attribute embedding + class (word name) embedding



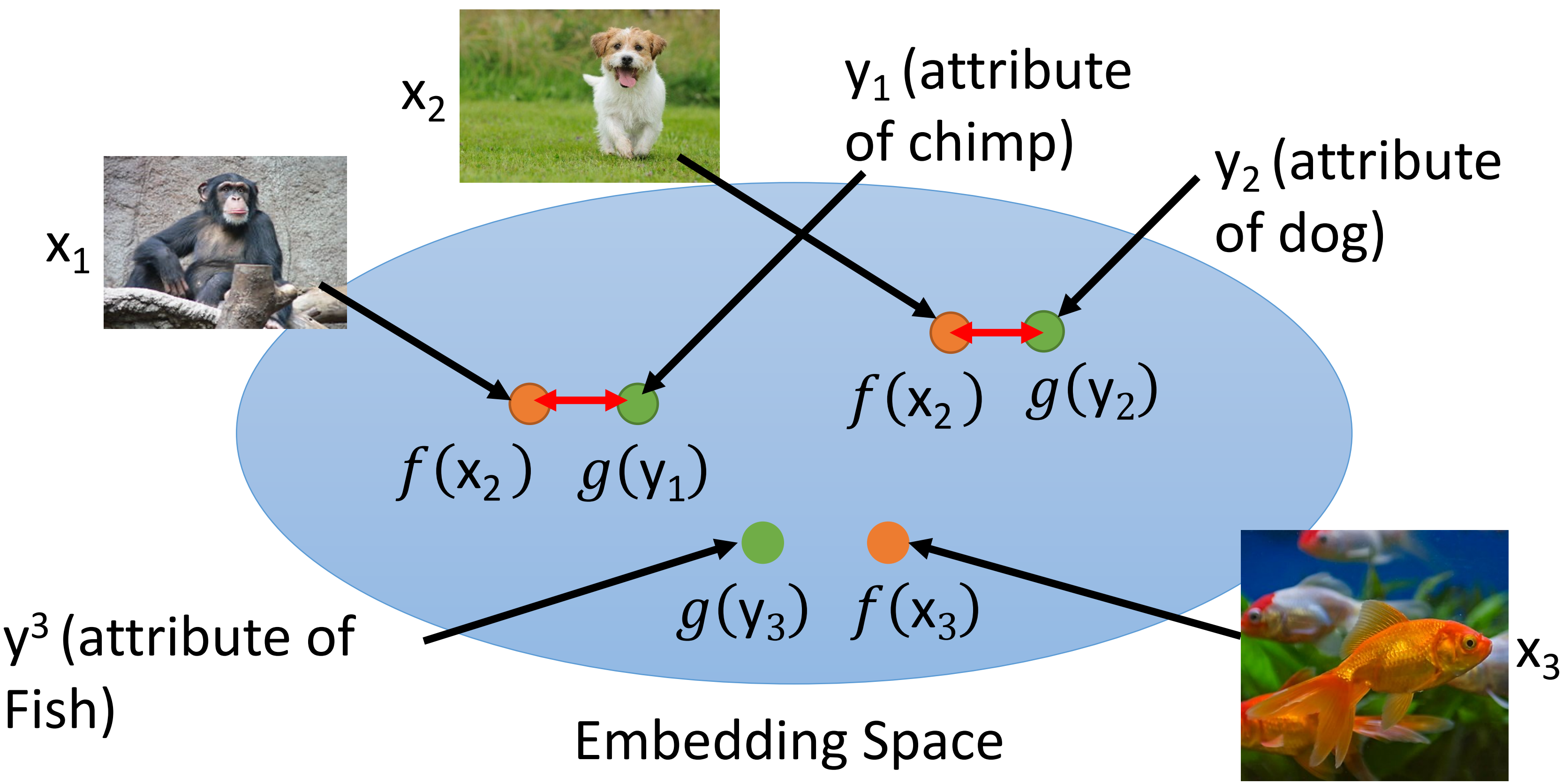
Zero-shot Learning

$f(*)$ and g($*$) can be NN.

Training target:

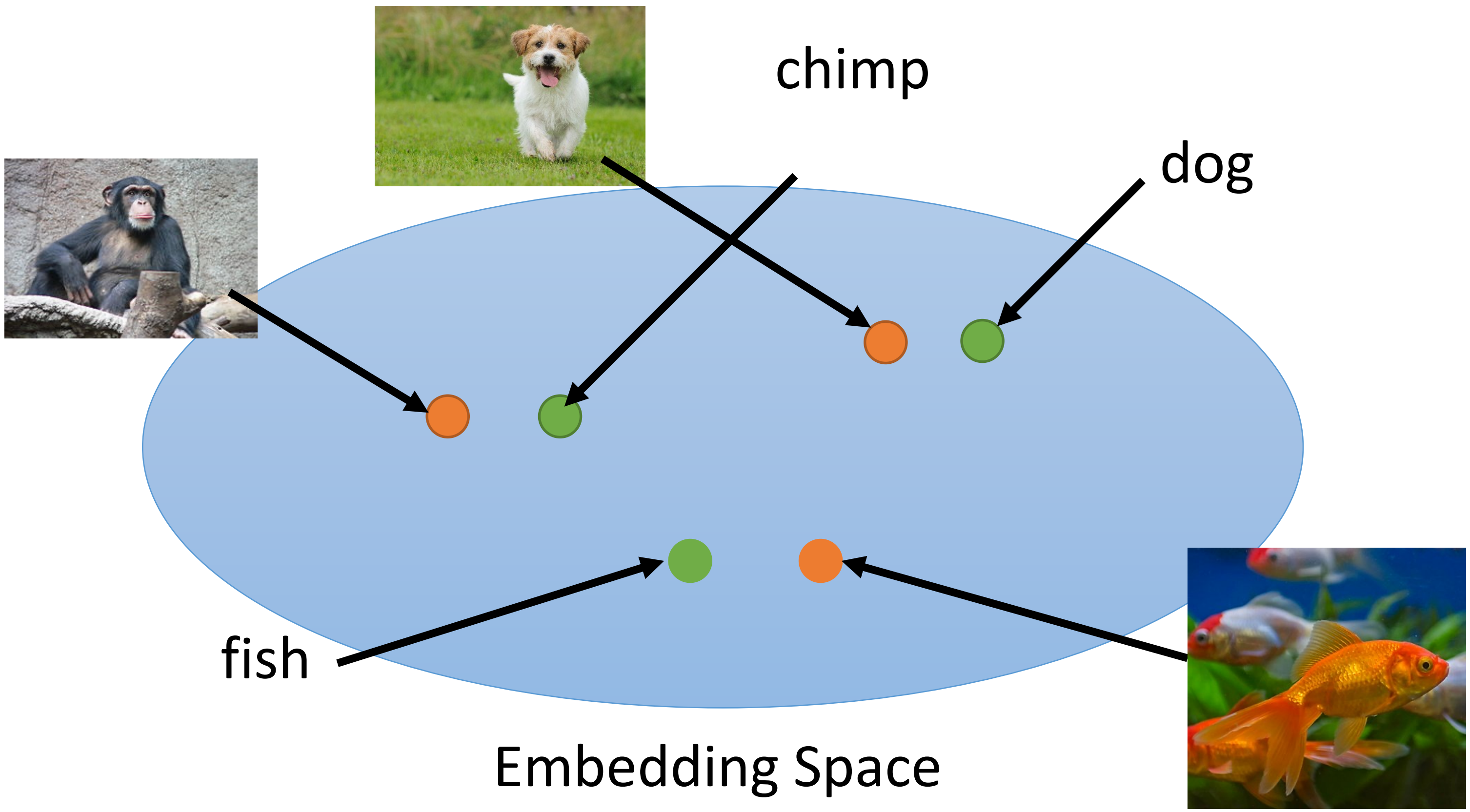
$f(x_n)$ and $g(y_n)$ as close as possible

- Attribute embedding



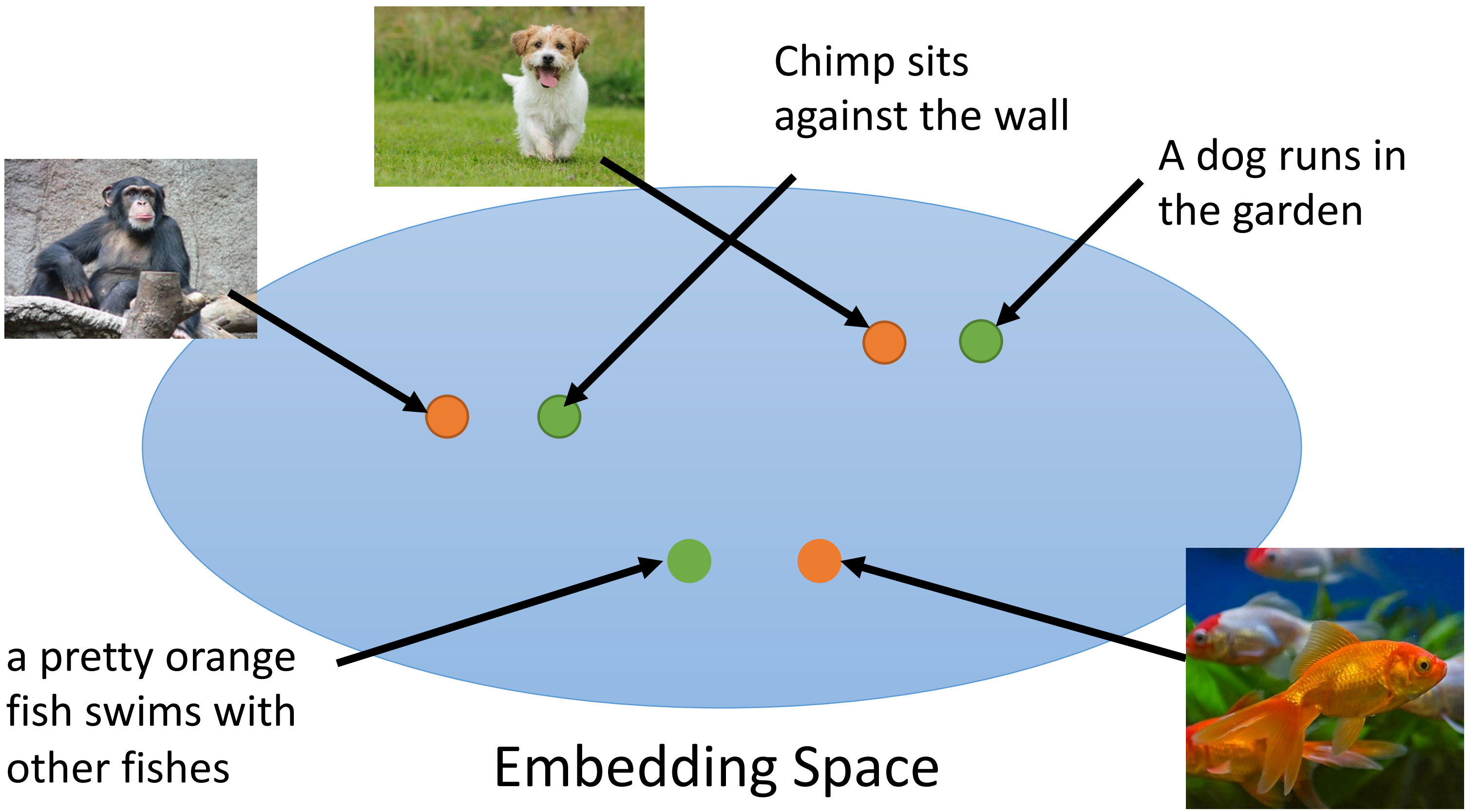
y_i are linked together by a class relationship (e.g. class name embedding as W2v)

More on Vision-Language: Representation Learning



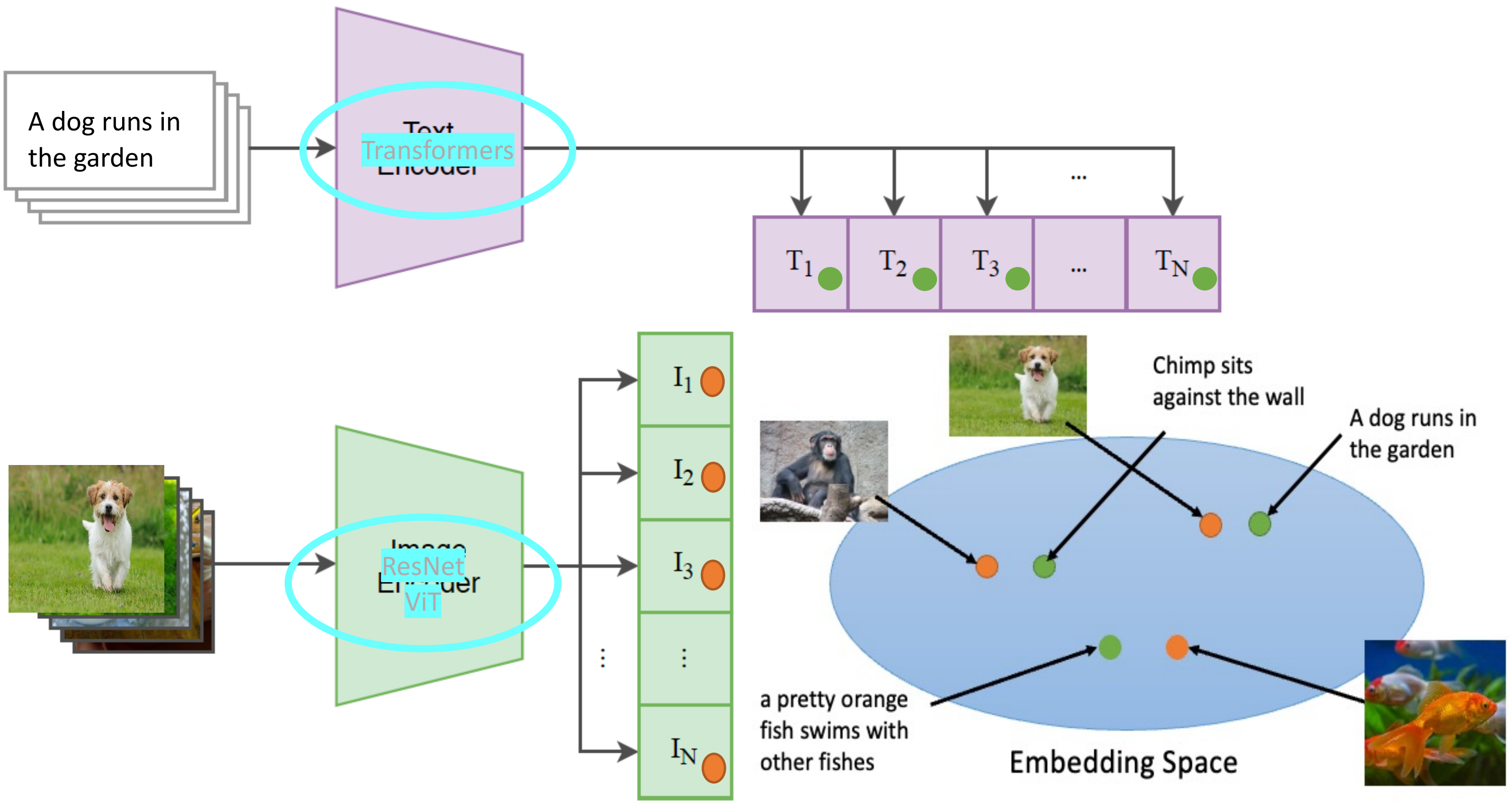
CLIP: Vision + Language Models (VLM)

[Learning transferable visual models from natural language supervision. Radford/Sutskever ICML, 2021]



CLIP: Vision + Language Models (VLM)

Dual architecture: Text encoder + Image encoder



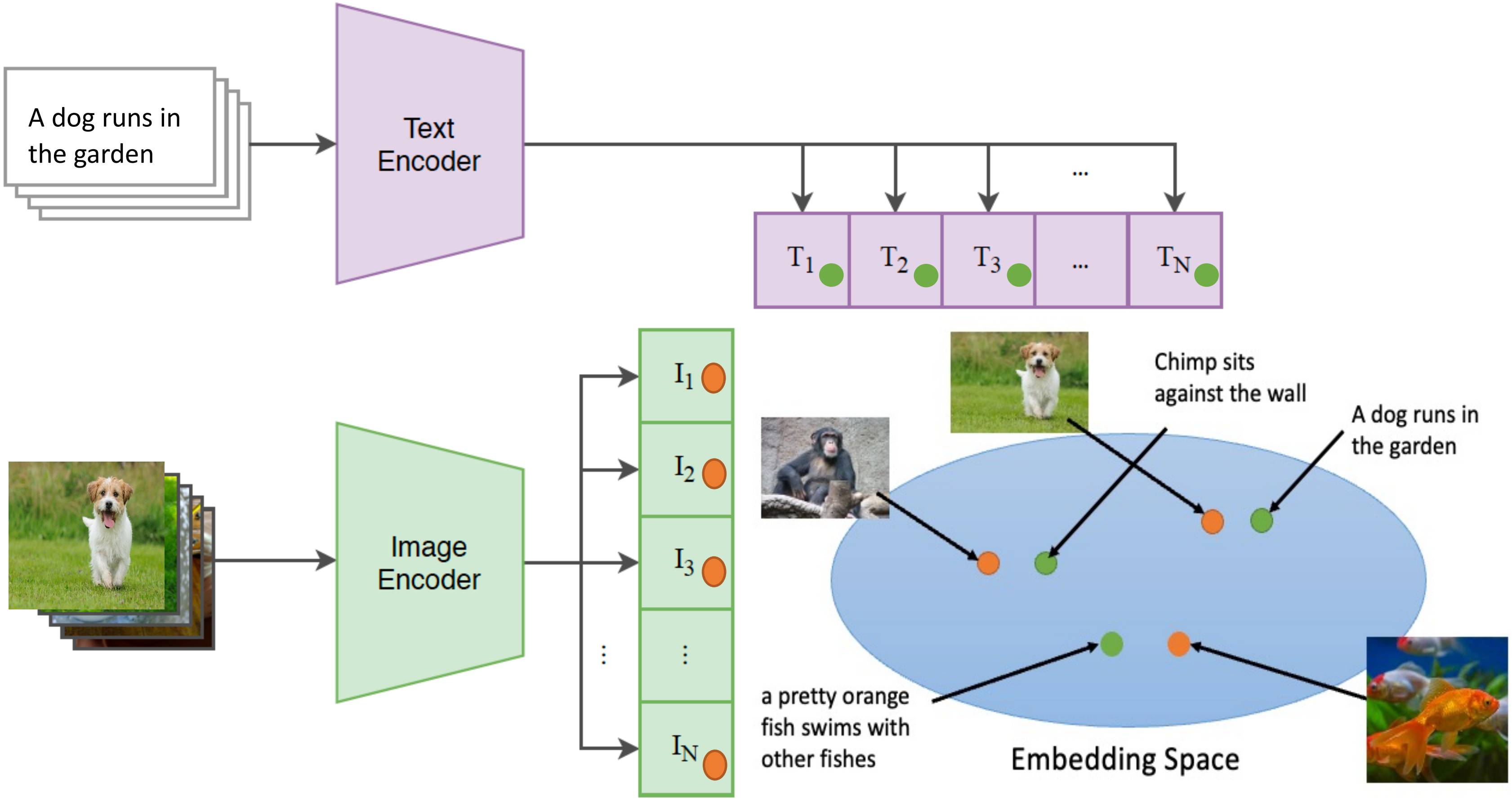
CLIP: Vision + Language Models (VLM)

Learning strategy

Training set: $A = \{(\mathbf{I}_n, \mathbf{T}_n)\}_n$ of image/caption pairs (coherent!)

Massive Text+Image =**400M pairs** to train the model (from the Internet)

Contrastive loss for training: positive pair vs negative one or set



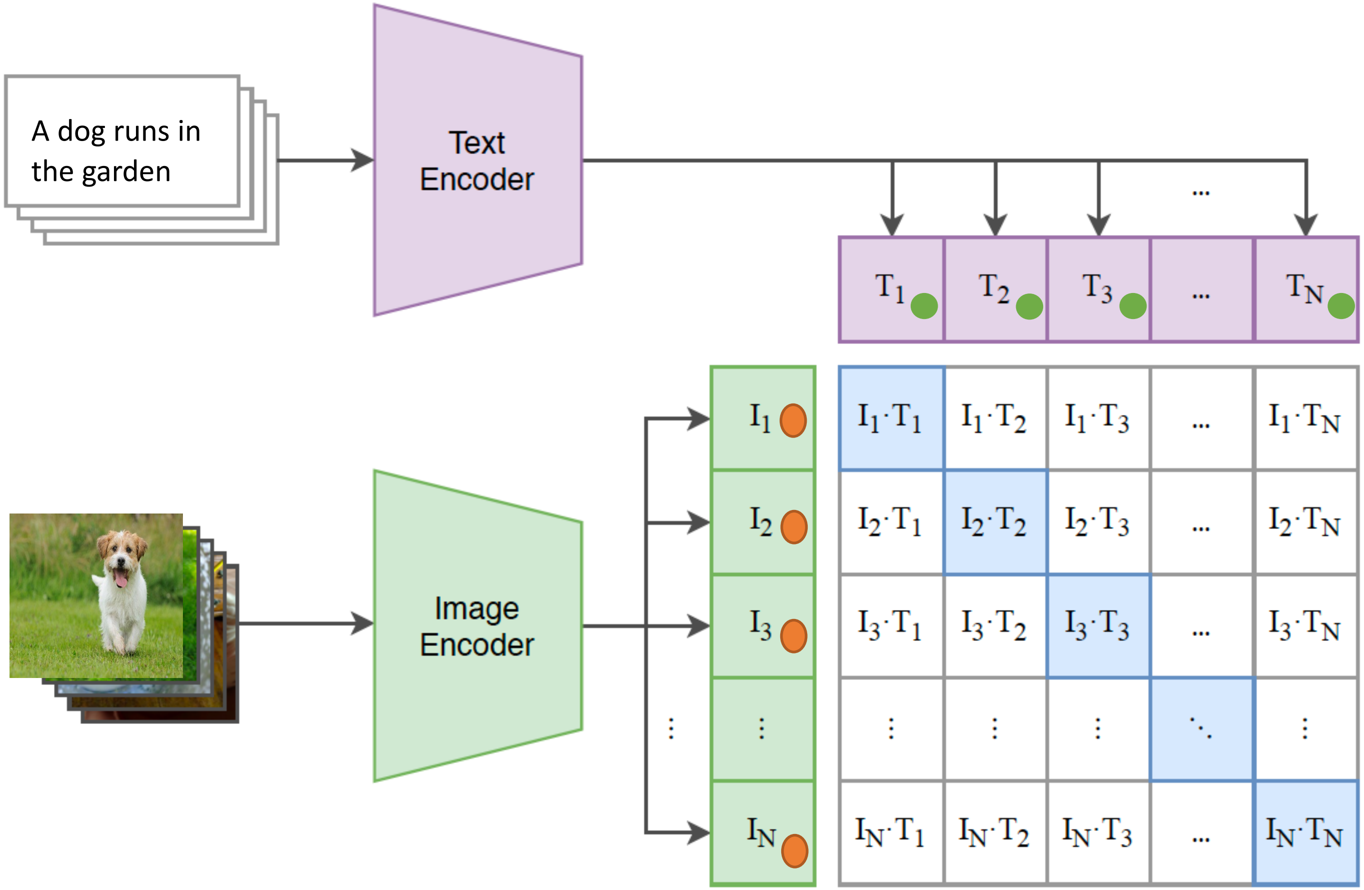
CLIP: Vision + Language Models (VLM)

Learning strategy

Training set: $A = \{(\mathbf{I}_n, \mathbf{T}_n)\}_n$ of image/caption pairs (coherent!)

Massive Text+Image =**400M pairs** to train the model (from the Internet)

Contrastive loss for training: positive pair vs negative one or set



CLIP: Vision + Language Models (VLM)

Learning strategy

Training set: $A = \{(\mathbf{I}_n, \mathbf{T}_n)\}_n$ of image/caption pairs (coherent!)

Massive Text+Image =**400M pairs** to train the model (from the Internet)

Contrastive loss for training: positive pair vs negative pair or more

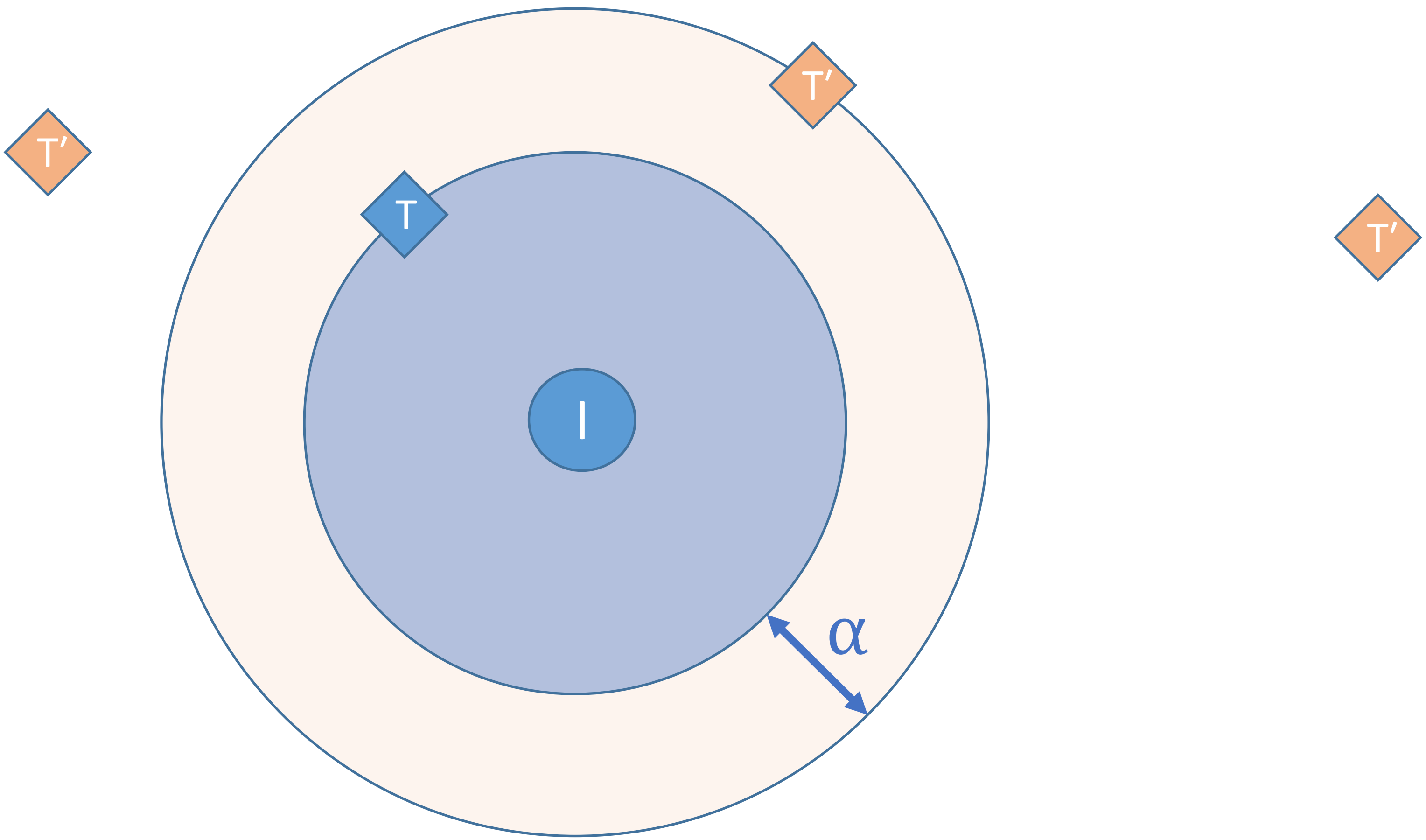
(contrastive) Triplet loss: A variant of the standard margin based loss (SVM)

- Triplet ($\mathbf{I}$, $\mathbf{T}$, $\mathbf{T}'$) (Batch = 3)
- Anchor: $\mathbf{I}$ (E.g image representation)
- Positive: $\mathbf{T}$ (E.g associated caption representation)
- Negative: $\mathbf{T}'$ (E.g contrastive caption representation)
- Margin parameter α

$$\boxed{\mathrm{TripletLoss}(\mathrm{I}, \mathrm{T}, \mathrm{T}') = \max\{0, \alpha + \mathrm{d}(\mathrm{I}, \mathrm{T}) - \mathrm{d}(\mathrm{I}, \mathrm{T}')\}}$$

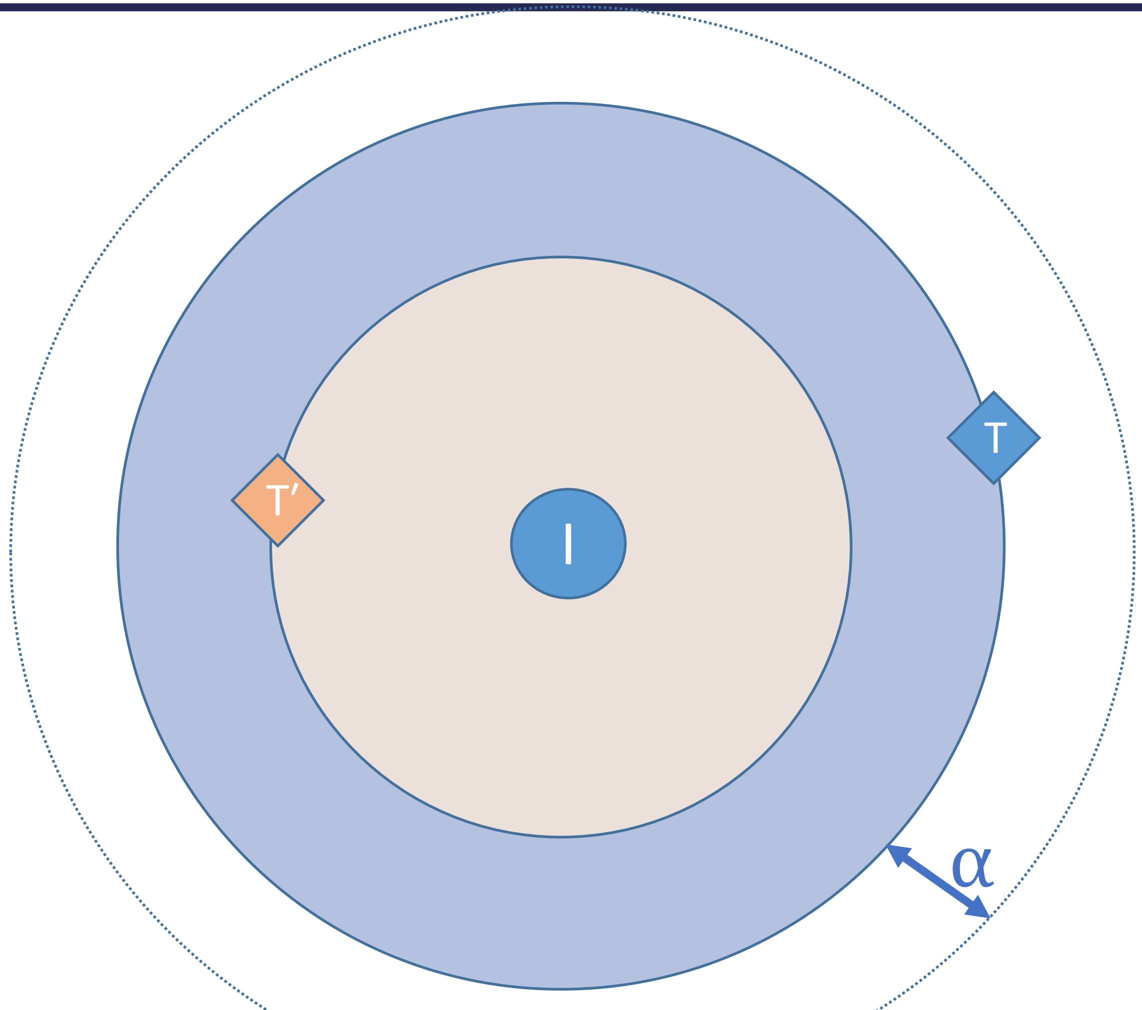
Learning strategy: triplet loss

$$\text{TripletLoss}(I, T, T') = \max\{ {\color{red}0}, \alpha + d(I,T) - d(I,T')\}$$



Learning strategy: triplet loss

$$\mathrm{TripletLoss}(\mathrm{I}, \mathrm{T}, \mathrm{T}') = \max\{0, \alpha + d(\mathrm{I}, \mathrm{T}) - d(\mathrm{I}, \mathrm{T}')\}$$



Learning strategy: triplet loss

Hard negative margin-based loss:

Loss for a **batch** $\mathcal{B} = \{(\mathbf{I}_n, \mathbf{T}_n)\}_{n \in B}$ of image/sentence pairs:

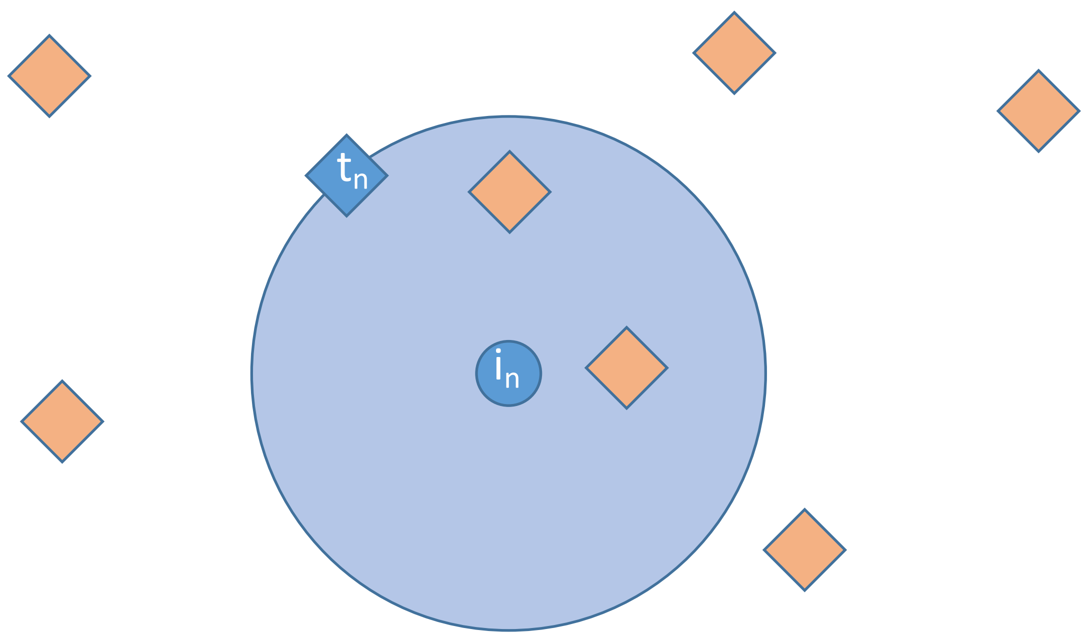
$$\boxed{\mathcal{L}(\boldsymbol{\Theta}; \mathcal{B}) = \frac{1}{|B|} \sum_{n \in B} \left(\begin{array}{l} \max\limits_{m \in C_n \cap B} \operatorname{loss}(\boldsymbol{I}_n, \boldsymbol{T}_n, \boldsymbol{T}_m) \\ + \max\limits_{m \in D_n \cap B} \operatorname{loss}(\boldsymbol{T}_n, \boldsymbol{I}_n, \boldsymbol{I}_m) \end{array} \right)}$$

With C_n (resp. D_n) set of indices of caption (resp. image) unrelated to n-th element

Learning strategy: hard negative triplet loss

Mining hard negative contrastive example:

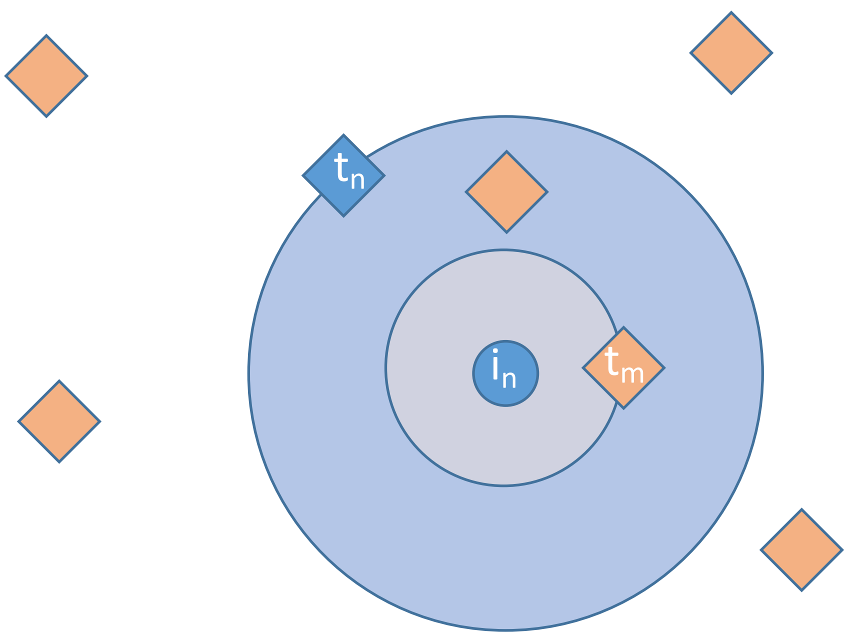
$$\boxed{\mathcal{L}(\boldsymbol{\Theta};\mathcal{B}) = \frac{1}{|B|}\sum_{n\in B}\left(\begin{array}{l}\max\limits_{m\in C_n\cap B} \operatorname{loss}\left(\boldsymbol{I}_n,\boldsymbol{T}_n,\boldsymbol{T}_m\right)\\ +\max\limits_{m\in D_n\cap B} \operatorname{loss}\left(\boldsymbol{T}_n,\boldsymbol{I}_n,\boldsymbol{I}_m\right)\end{array}\right)}$$



Learning strategy: hard negative triplet loss

Mining hard negative contrastive example:

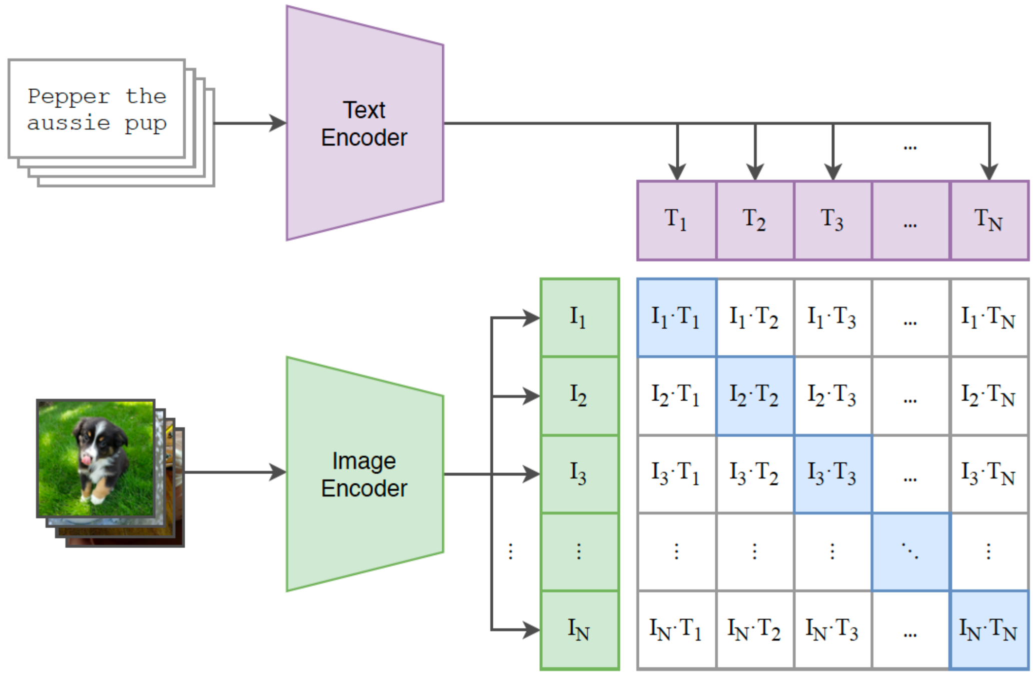
$$\mathcal{L}(\boldsymbol{\Theta};\mathcal{B}) = \frac{1}{|B|}\sum_{n\in B}\left(\max_{m\in C_n\cap B} \operatorname{loss}(\boldsymbol{I}_n,\boldsymbol{T}_n,\boldsymbol{T}_m) + \max_{m\in D_n\cap B} \operatorname{loss}(\boldsymbol{T}_n,\boldsymbol{I}_n,\boldsymbol{I}_m)\right)$$



CLIP: Vision + Language Models (VLM)

Massive Text+Image =**400M pairs** to train the model (from the Internet)

Contrastive loss for training

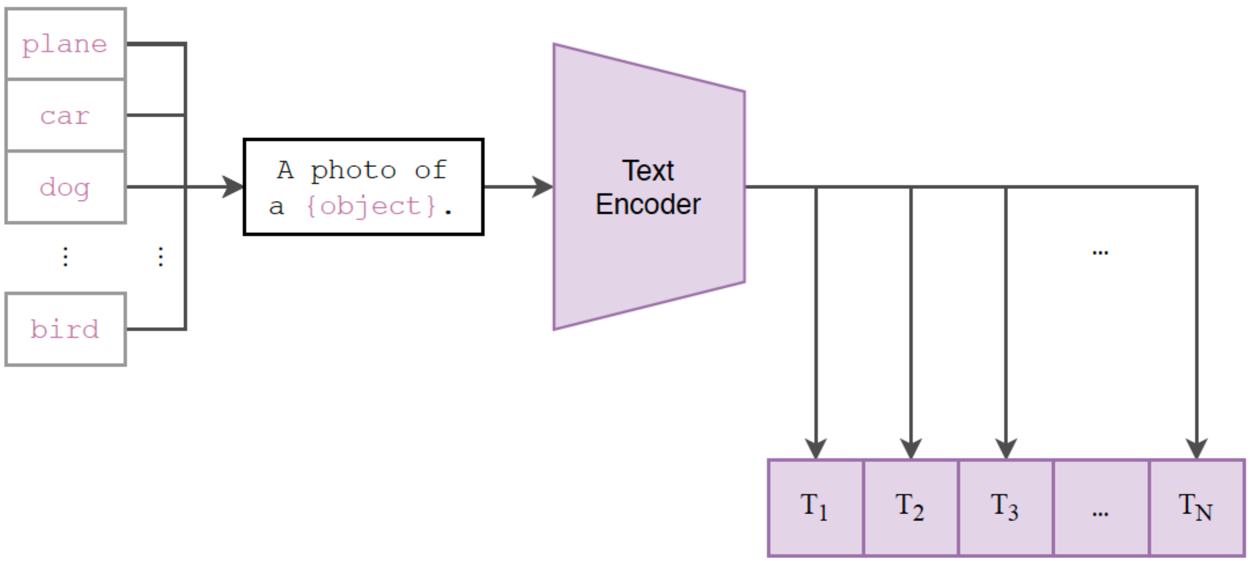


$$\mathcal{L}_{InfoNCE_{CLIP}} = -\sum_i \log\left(\frac{exp(\frac{sim(I_i, T_i)}{\tau})}{\sum_{k=1}^{N} exp(\frac{sim(I_i, T_k)}{\tau})}\right)$$

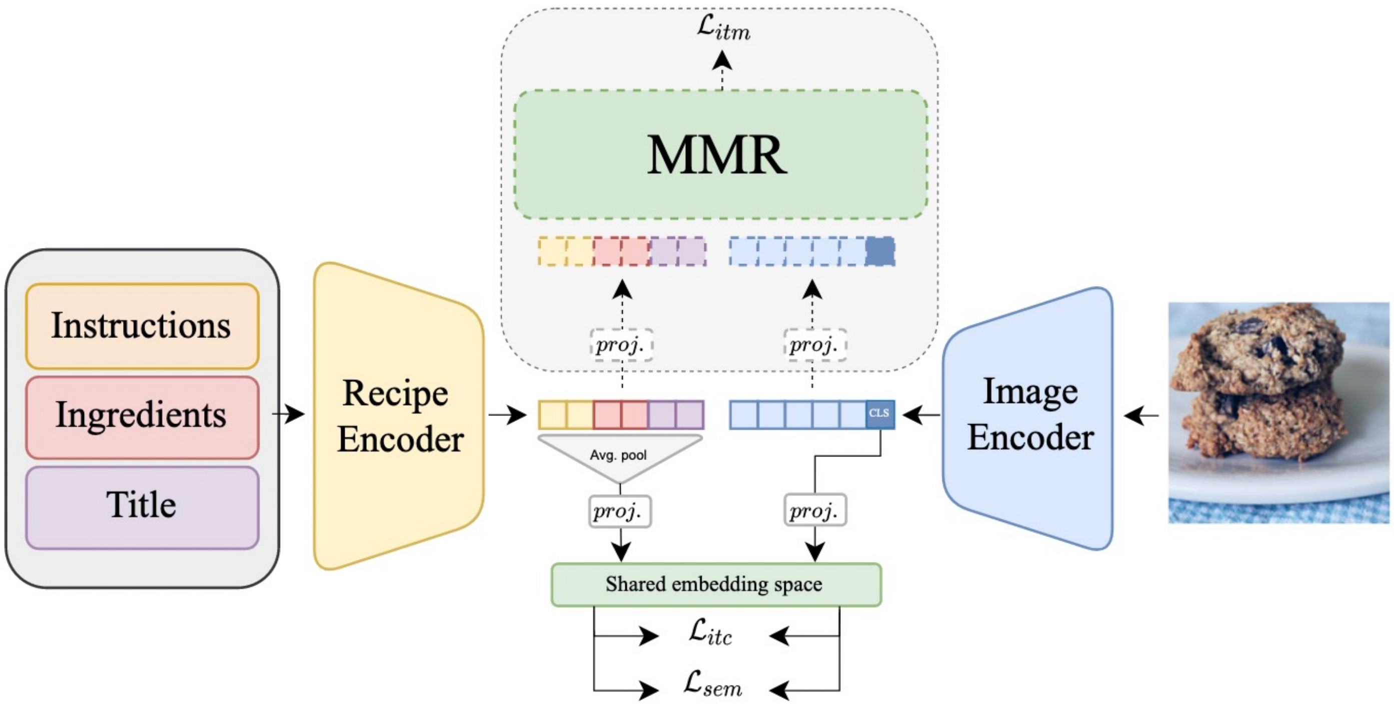
CLIP: Vision + Language Models (VLM)

Pre-trained encoders = **dual encoders** (Text/Image)

used for Zero-shot classifier, and other downstream tasks



A lot of variants ....



Title query	Ingredient query	Instruction query
Mint Chocolate Chip Frosting.	1 cup Unsalted Butter, 2 Tablespoons Heavy Cream, 2 drops Green Food Coloring, .. Chocolate..	Add sugar, cream, peppermint, and food coloring... ...scoop the frosting and place on top of your cupcakes Source: Chocolate Cupcakes with Mint Chocolate Chip ...
Honey-Grilled Chicken.	1 broiler-fryer chicken, halved, 34 cup butter, melted, 14 cup honey..	Place the halved chicken in a large, shallow container... ...Combine the remaining ingredients, stirring sauce well Grill chicken, skin side up..
The Best Kale Ever.	1/2 cup Kale, 1 teaspoon Olive Oil, 1/4 teaspoons Red Pepper Flakes..	Wash and cut kale off the stems... Heat olive oil on medium heat and add garlic Add in kale and red pepper flakes..

Top 5 retrieved images

Vision-Language Models

Part II:

VLMs using LLMs

Outline

1. **From classification to detection, segmentation, …**
2. Vision-Language Models in the era of LLMs

(Visual) Transformers

ViT (Vision Transformers) architecture
=> **Self attention encoder** modules for classification

Published as a conference paper at ICLR 2021

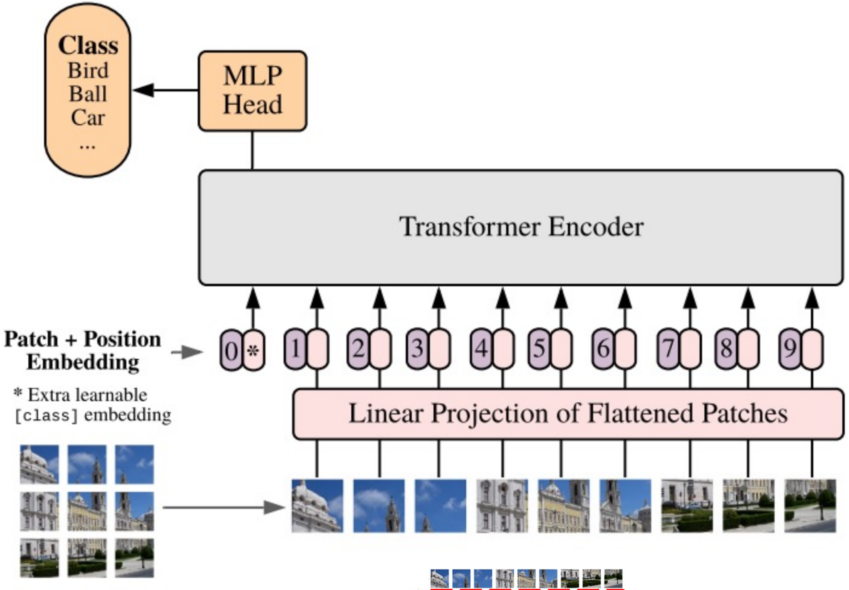
AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE

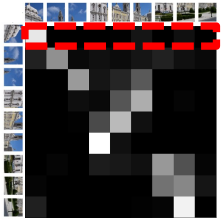
Alexey Dosovitskiy*,†, Lucas Beyer*, Alexander Kolesnikov*, Dirk Weissenborn*, Xiaohua Zhai*, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, Neil Houlsby*,†

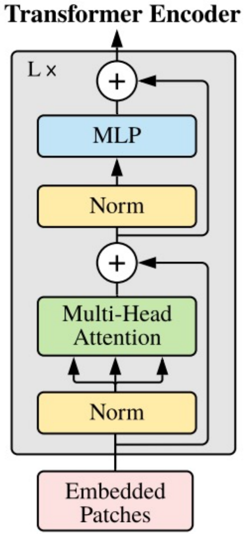
*equal technical contribution, †equal advising

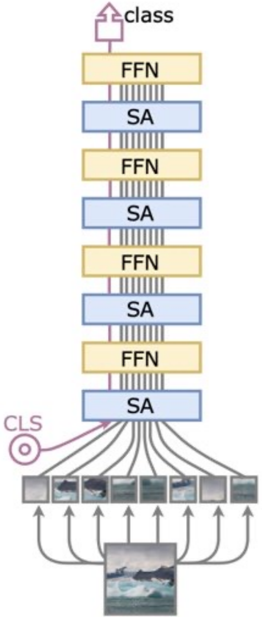
Google Research, Brain Team

{adosovitskiy, neilhoulsby}@google.com









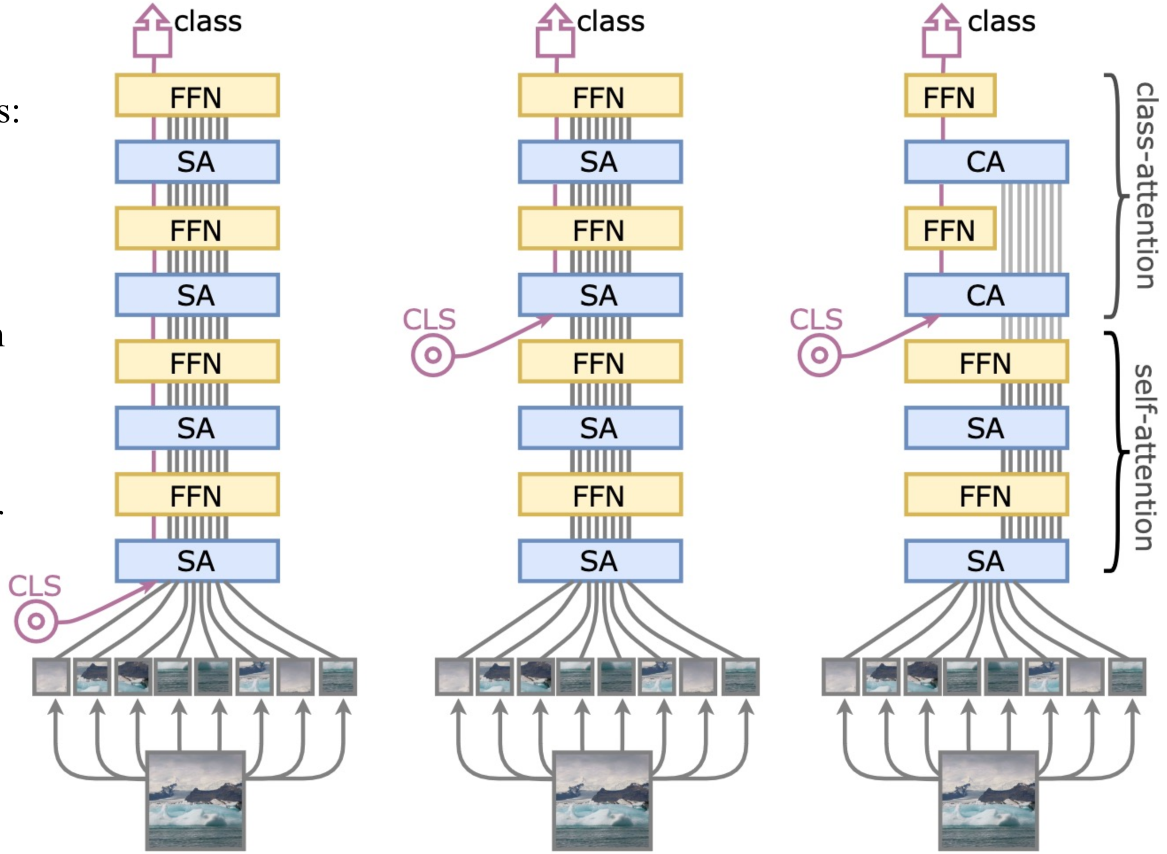
Class Activation architecture

In ViT class embedding CLS token inserted along with the patch embeddings:

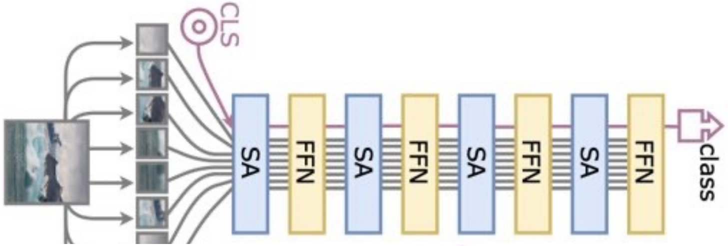
- helping the attention process
- preparing the vector to be fed to the classifier

CaiT freezes the patch embeddings when inserting CLS:

- last part of the network (2 layers) dedicated to summarizing the information to be fed to the classifier
- save compute

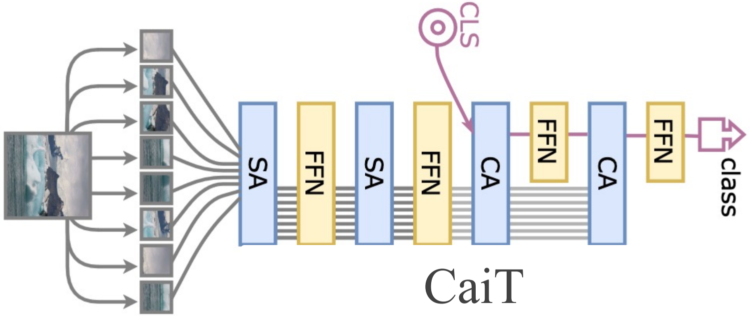


(Visual) Transformers for detection, segmentation, …



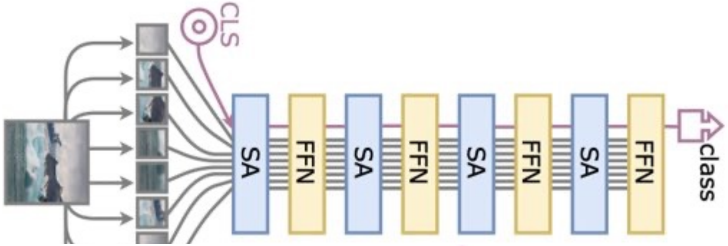
Design output for classification, detection, …

- CLS token for classification
- CaiT strategy: CLS to decode the embeddings



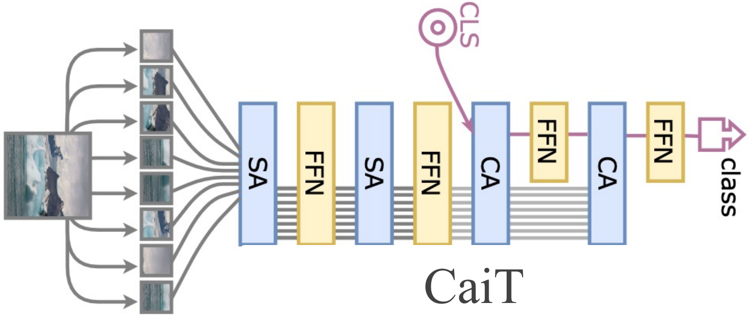
CaiT

(Visual) Transformers for detection, segmentation, …



Design output for classification, detection, …

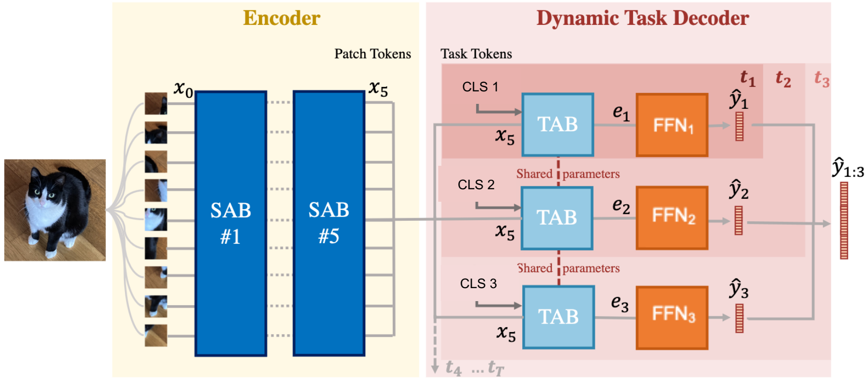
- CLS token for classification



- CaiT strategy: CLS to decode the embeddings

CaiT

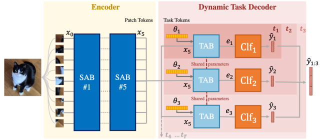
- Extension to incremental classification task learning:



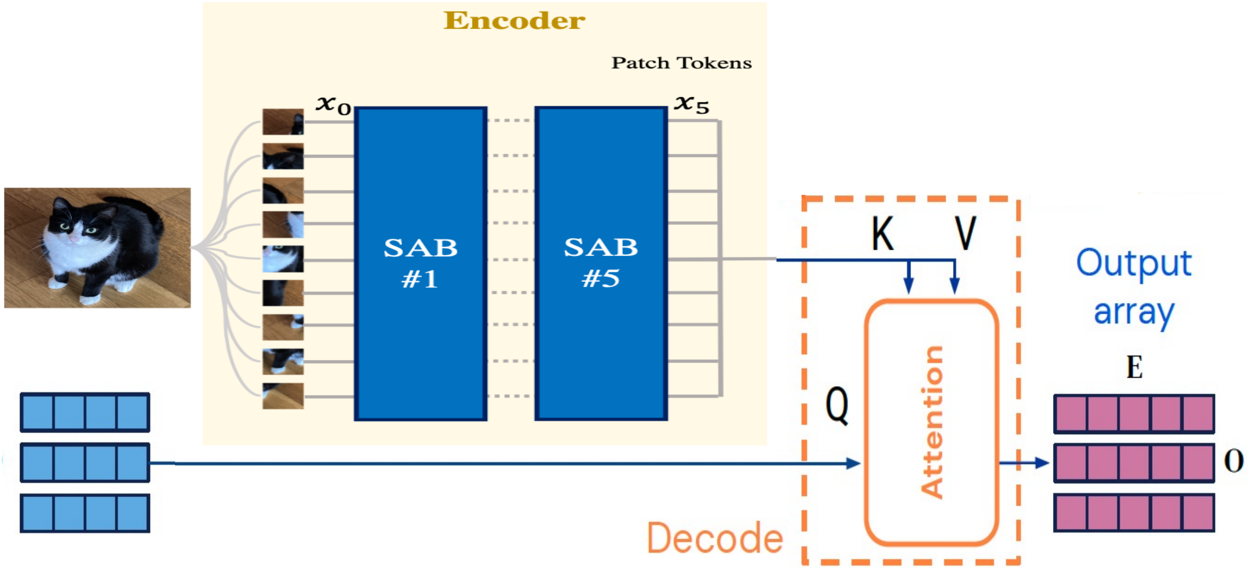
TAB: Task Attention Block

And for other type of output as detection?

(Visual) Transformers for detection, segmentation, …



To summarize:

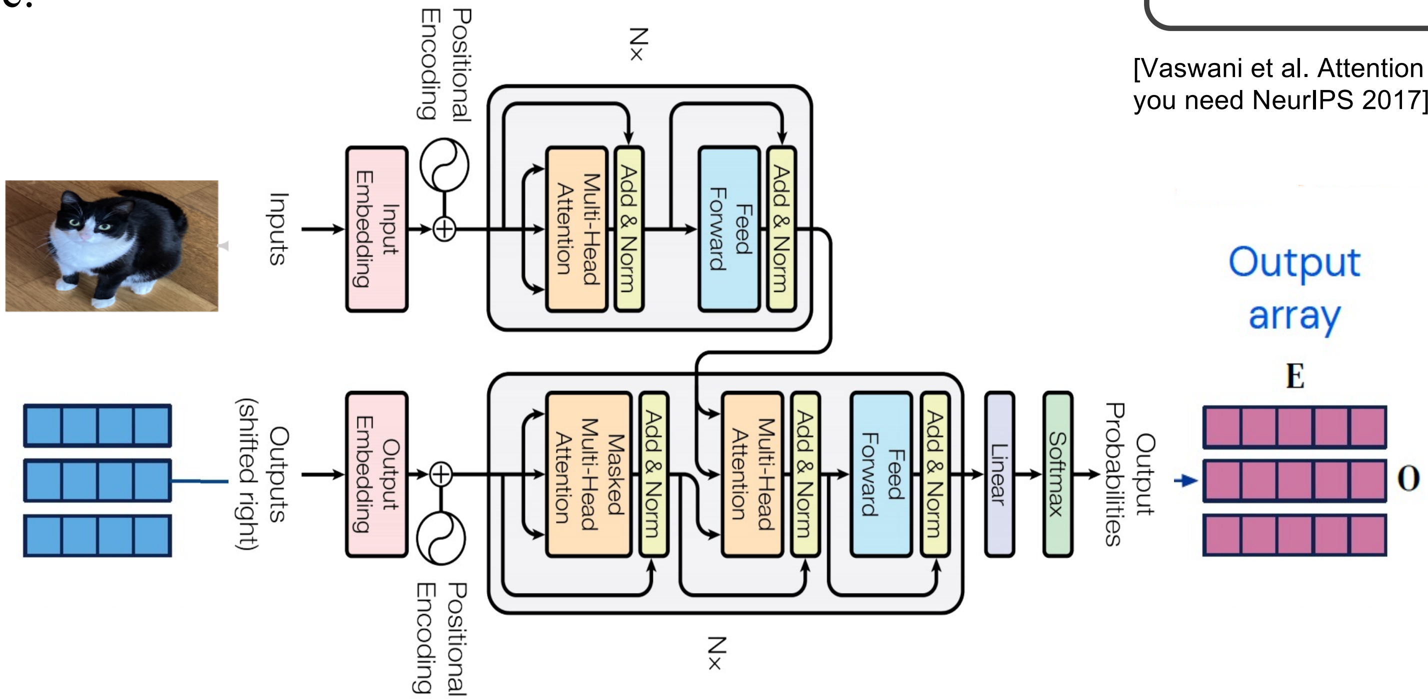


(Visual) Transformers for detection, segmentation, …

Transformers

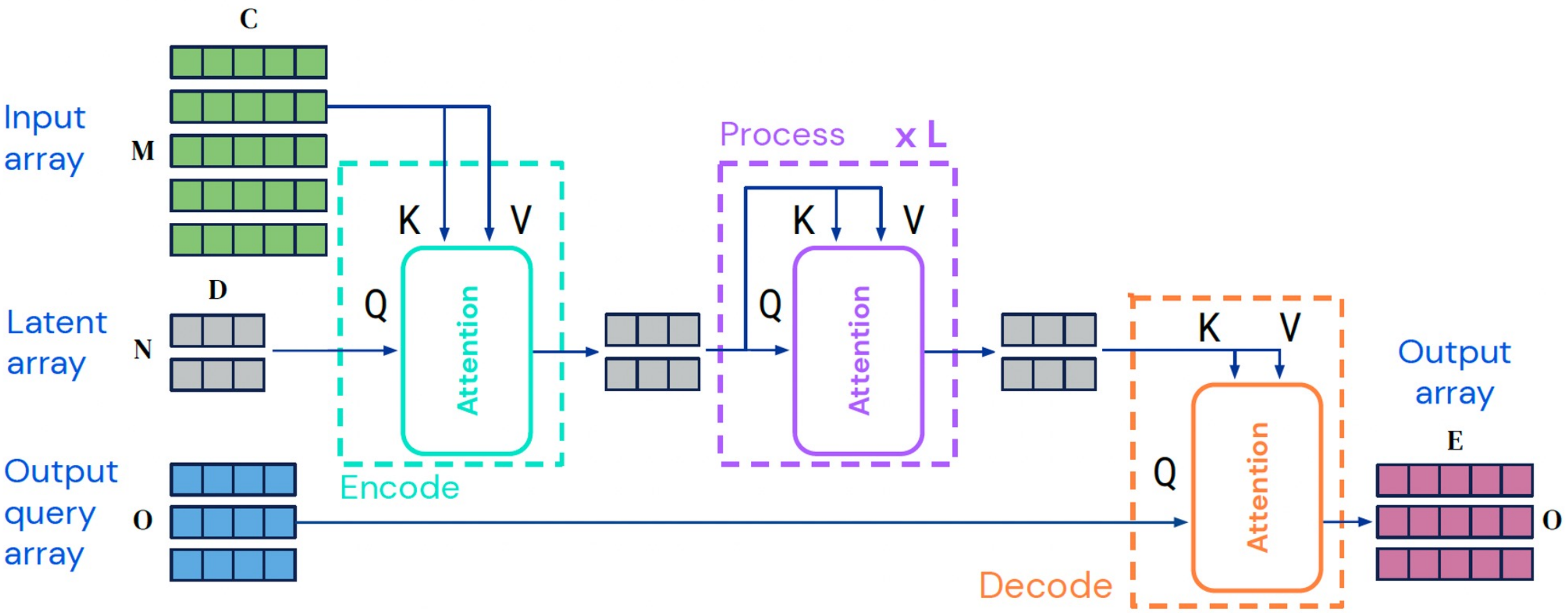
[Vaswani et al. Attention is all you need NeurIPS 2017]

To summarize:

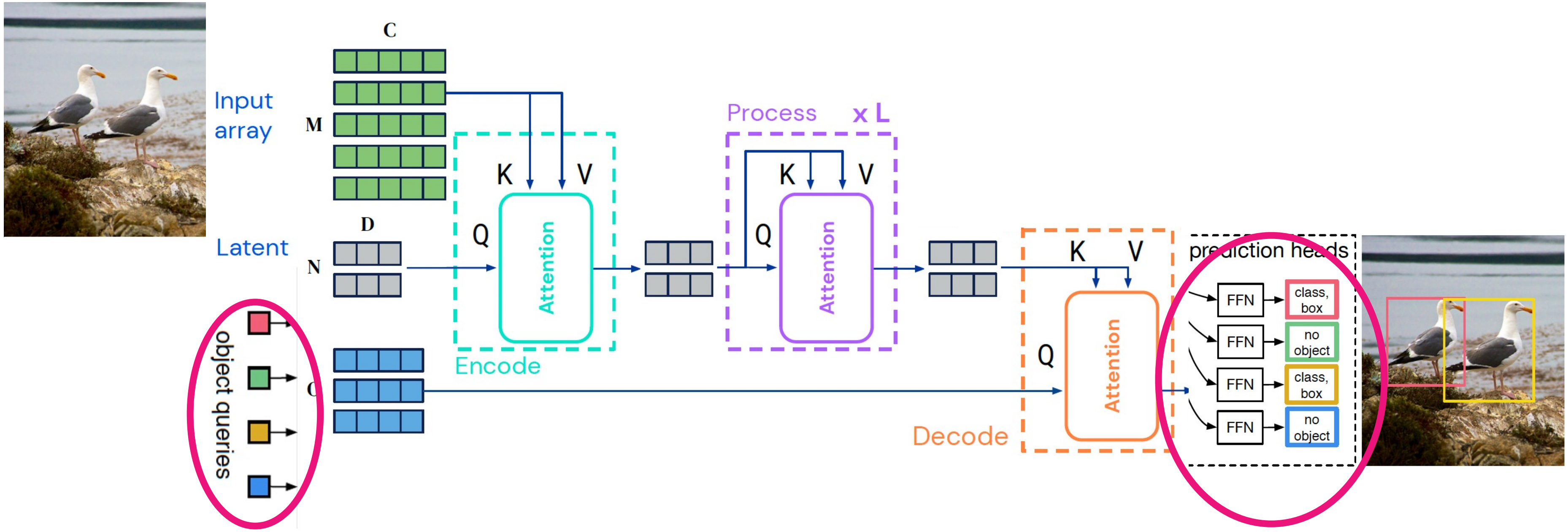


(Visual) Transformers for detection, segmentation, …

Just to complete the big picture
[Perceiver IO A General Architecture for Structured Inputs & Outputs ICLR22]

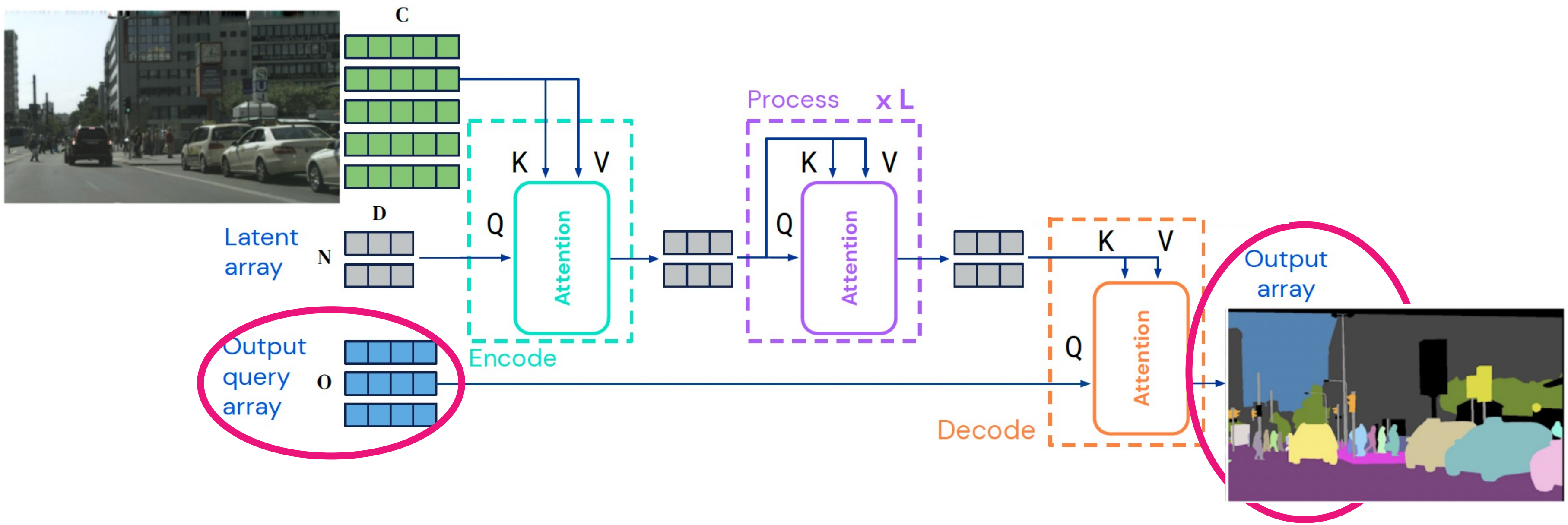


Output query array / Output array defines the downstream task: detection



(Visual) Transformers for detection, segmentation, …

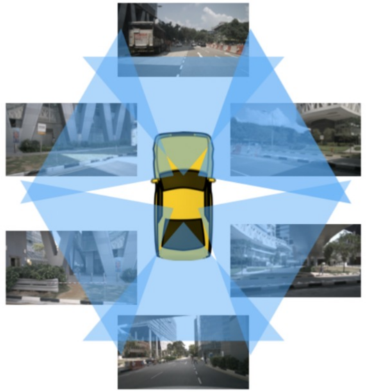
Output query array / Output array defines the downstream task: segmentation …

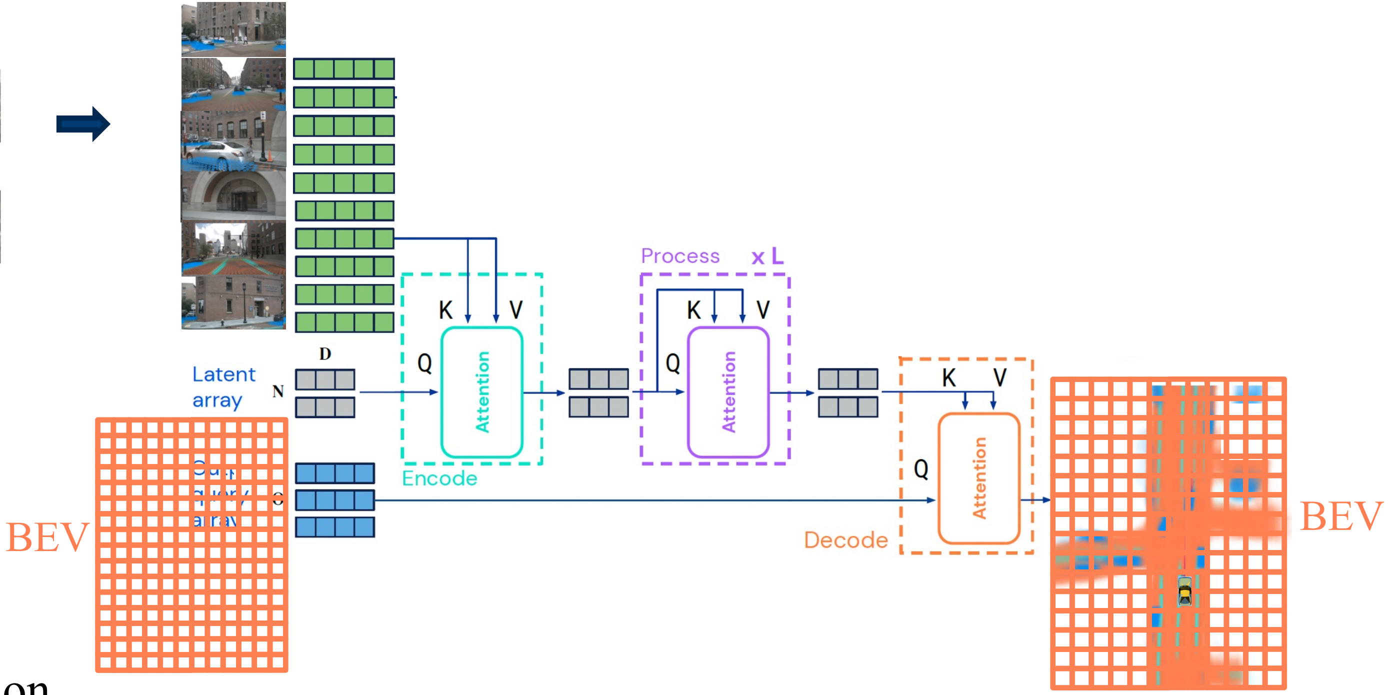


(Visual) Transformers for detection, segmentation, …

Input array = N cameras

Output array = Bird Eye View (BEV) representation



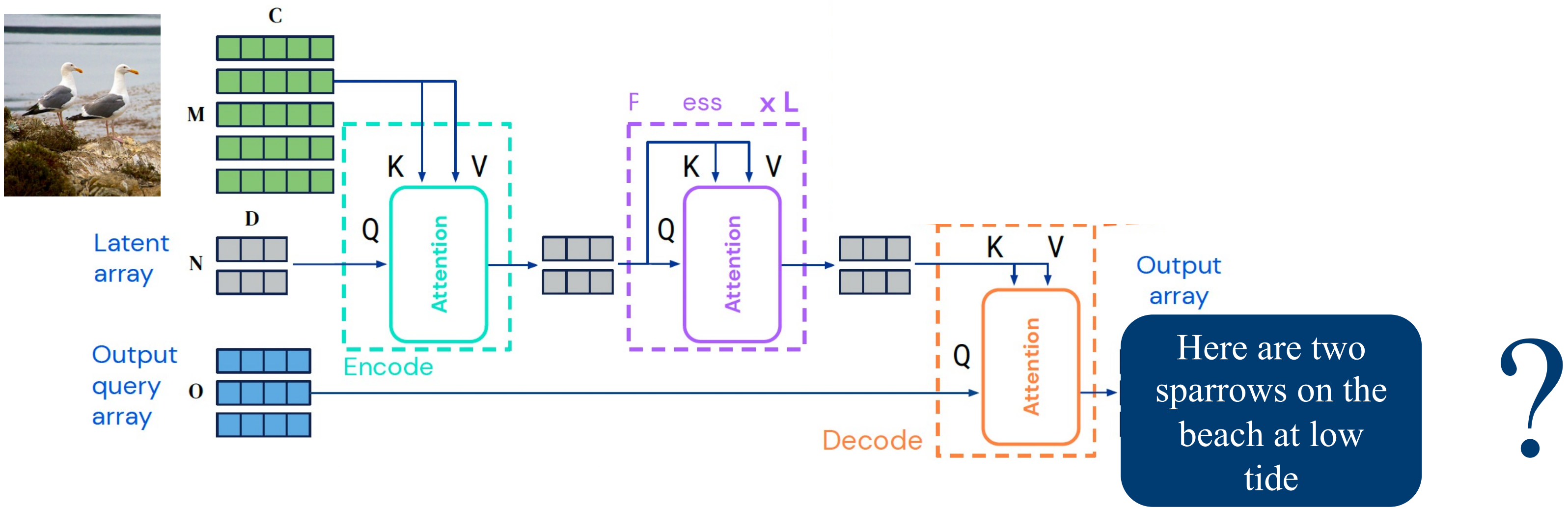


Merging by attention

Many Foundation models for Autonomous driving based on this framework

(Visual) Transformers for detection, segmentation, …

From Image to sentences!

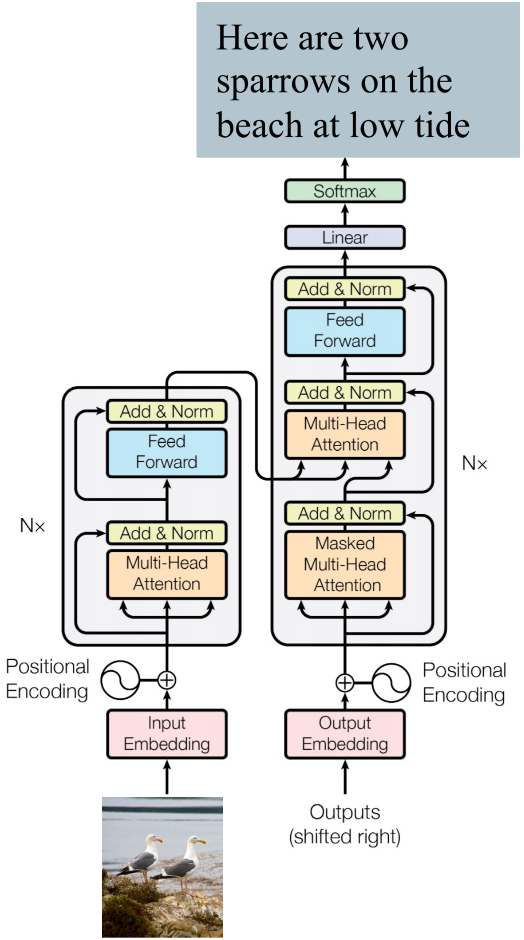


Outline

1. From classification to detection, segmentation, …
2. **Vision-Language Models in the era of LLMs**

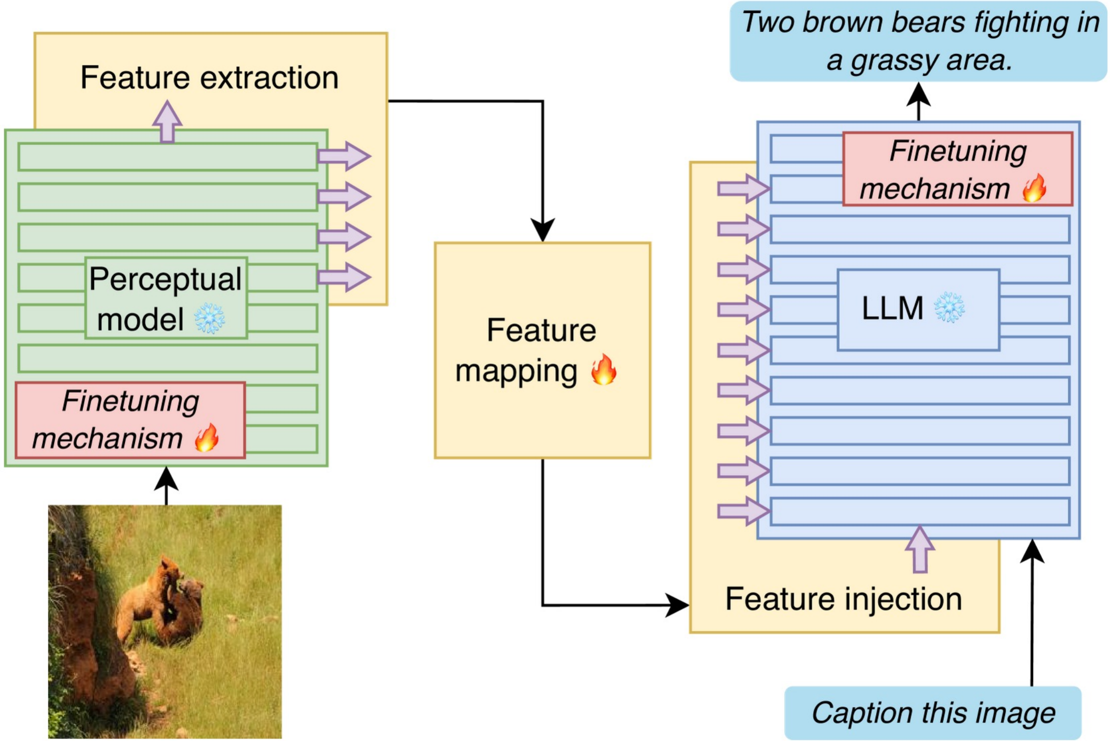
   - Unimodal models with connection

   - One model for all



Vision Encoder + LLM Decoder

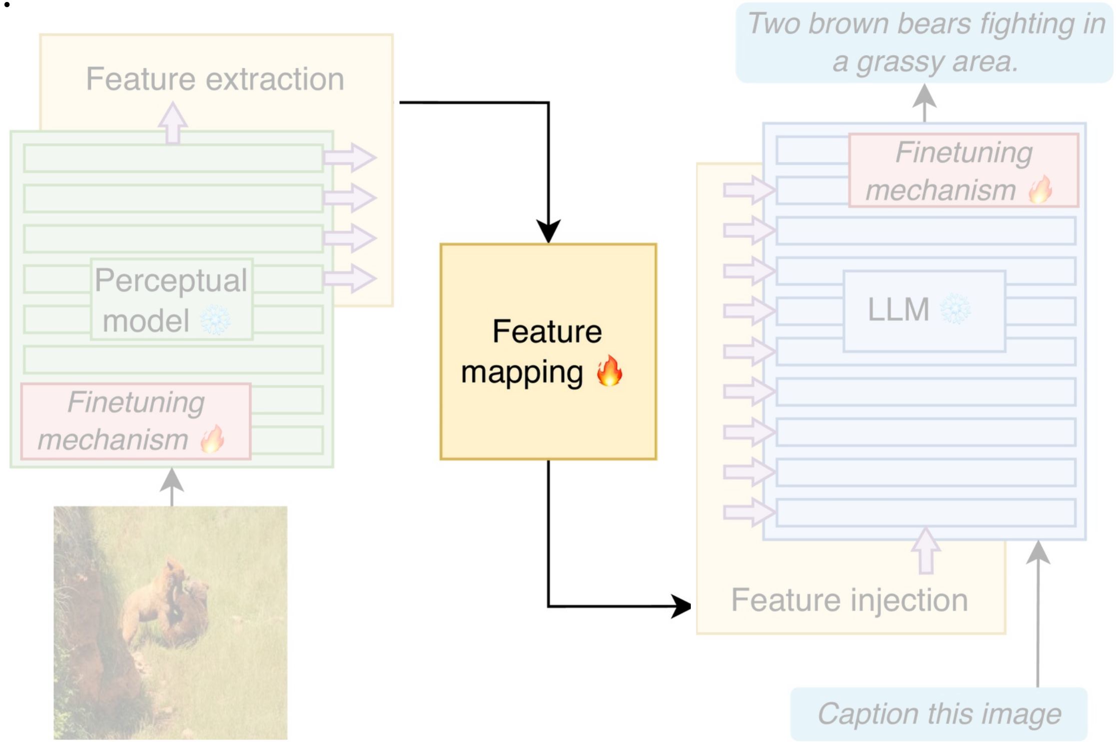
Image as input, textual caption as output



Why this modeling? Because the best LLM ever designed (and the plug&play update if a new LLM is released)

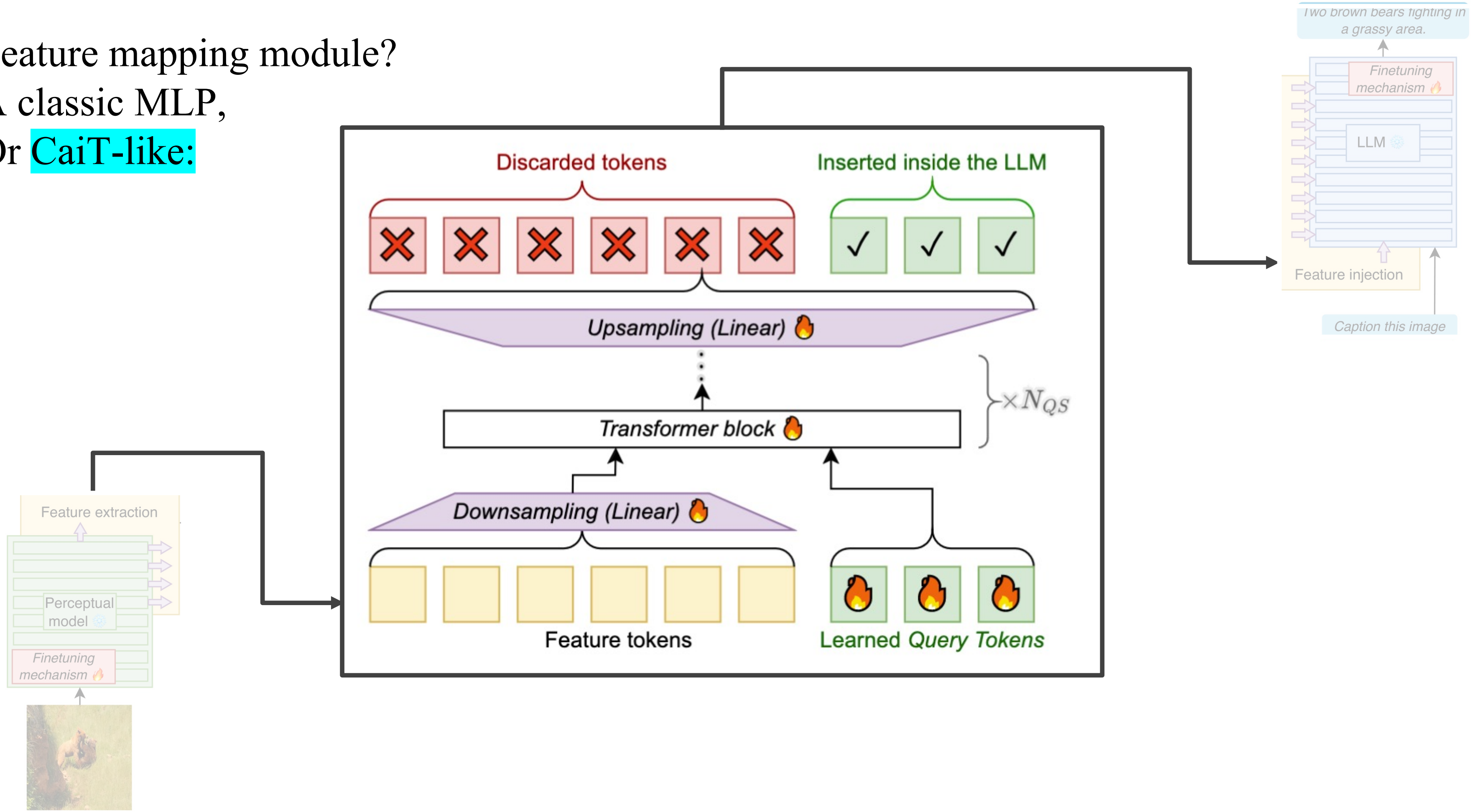
Vision Encoder + LLM Decoder

Feature mapping module?
A classic MLP, or:



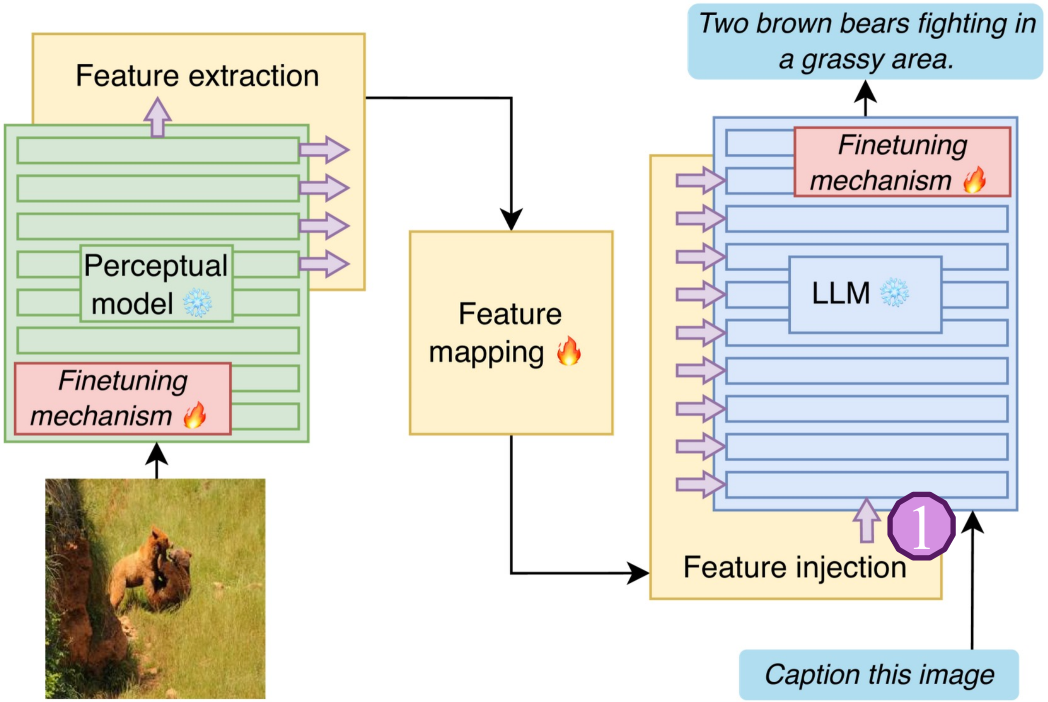
Vision Encoder + LLM Decoder

Feature mapping module?
A classic MLP,
Or CaiT-like:



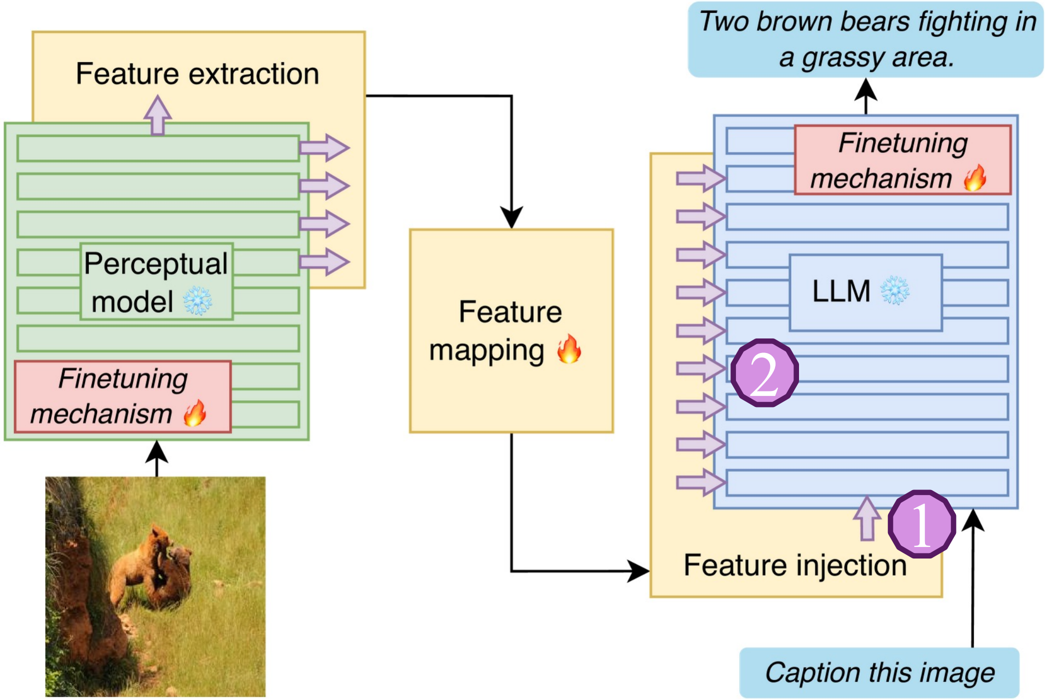
Vision Encoder + LLM Decoder

After feature mapping, feature **injection**!



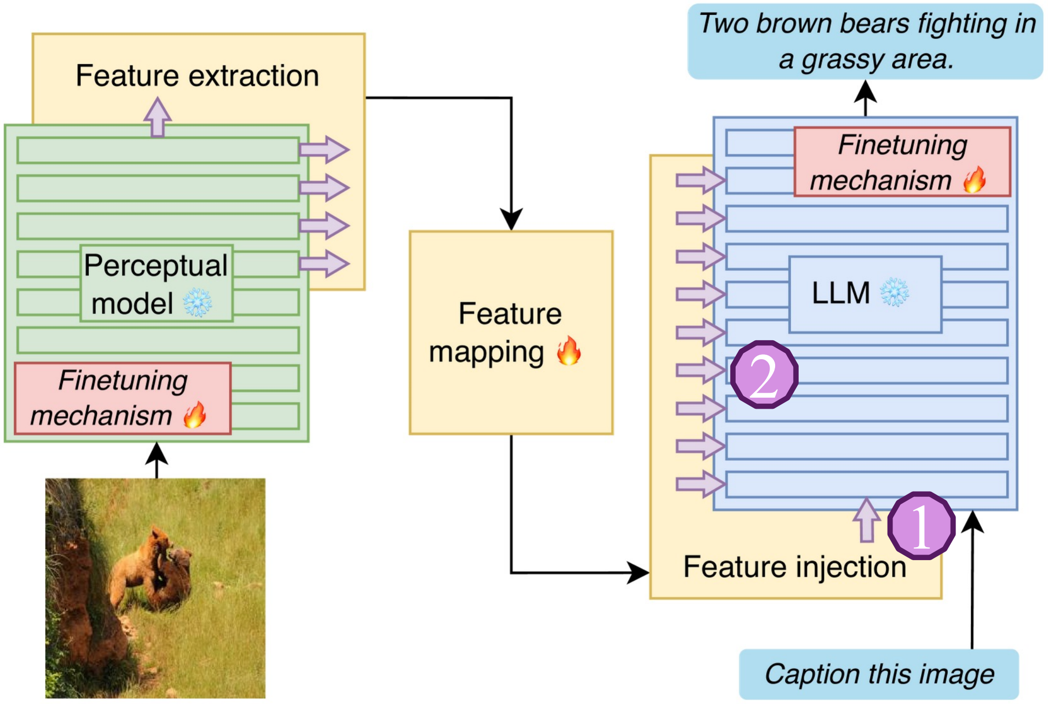
Vision Encoder + LLM Decoder

After feature mapping, feature **injection**!

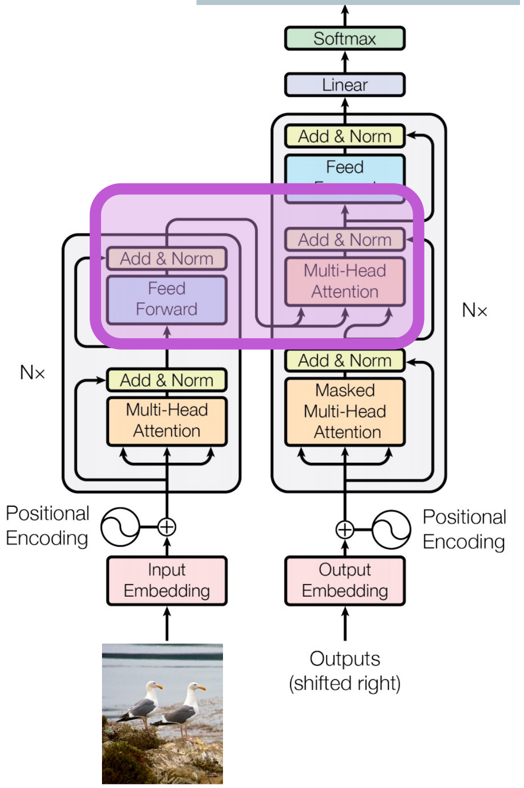


Vision Encoder + LLM Decoder

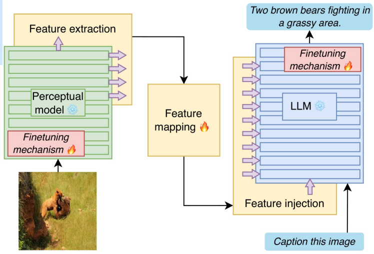
After feature mapping, feature **injection**!



Here are two sparrows on the beach at low tide

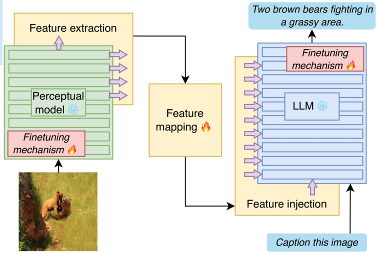


Vision Encoder + LLM Decoder



Method	Backbones		Adaptation mechanism				# Tr.
	LLMs	Perceptual Enc.	Feature extraction	Feature mapping	Feature injection	Fine-tuning mechanisms	params.
Flamingo [1]	Chinchilla [33]	NFNet [5]	Tokens from last layer	Perceiver Resampler (Transformer)	GATED XATTN-DENSE (Cross-attention)	–	10B
BLIP-2 [43]	OPT [92], FlanT5 [13]	CLIP [65]	Tokens from last layer	Q-Former	1st layer token injection	–	1.2B
MAGMA [22]	GPT-J 6B [86]	CLIP [65] / NFNet [5]	Tokens from last layer	MLP	1st layer token injection	fine-tuning of perceptual model	243M
MAPL [58]	GPT-J 6B [86]	CLIP-L [65]	Tokens from last layer	QPMapper (d_{embed}=256, 4 layers)	1st layer token injection	–	3.4M
PromptFuse [46]	BART [42]	ViT [19]	Tokens from last layer	*nothing*	–	prompt tuning	15K
LiMBeR [60]	GTP-J 6B [86]	CLIP [65]	Tokens from last layer	Linear projection	1st layer token injection	–	12.5M
eP-ALM [72]	OPT-2.7B/6.7B [92]	ViT [77], AST [27], TimeSformer [4]	CLS tokens from n last layers	(Shared) linear projection	Token injection in intermediate layers	prompt tuning	4.2M
LLaMA-Adapter [25, 91]	LLaMA[82]	CLIP [65]	Tokens from last layer	Linear projection	Token injection in intermediate layers	inner-layer prompt tuning, bias tuning, norm tuning	14M
Frozen [84]	GPT-like [66]	NFNet [5]	Pooled output tokens	*nothing*	1st layer token injection	Fine-tune the NFNet	40.3M
ClipCap [61]	GPT-2[66]	CLIP [65]	Tokens from last layer	Transformer	1st layer token injection	–	43M
VL-Adapter [79]	BART [42], T5 [67]	CLIP [65]	Tokens from last layer	Linear projection	1st layer token injection	Adapters	5.8M
AnyMAL [62]	Llama 2-70B-chat [83],	CLIP [65], CLAP [23]	Tokens from last layer	Perceiver Resampler, or linear projection	1st layer token injection	LoRA [34]	–
DePALM$^{\text{QP,inner}}$	OPT-6.7B [92], LLaMA [82]	CLIP-L [65], DINOv2 [63], MAViL [36] TimeSformer [4]	Tokens from n last layers	QPMapper	Token injection in intermediate layers	prompt tuning	18.1M
DePALM							17.9M
DePALM$^{\text{R-rand,L0}}$, DePALM$^{\text{R-linear,L0}}$, DePALM$^{\text{R-QPMapper,L0}}$, DePALM$^{\text{R-avgpool,L0}}$			Tokens from last layer	Linear projection + Resampler	1st layer token injection		21M, 88M 18M, 21M
DePALM$^{\text{c-attn}}$			Tokens from n last layers	Projection + Small Transformer	Gated cross-attention		17.9M

Vision Encoder + LLM Decoder



Parameter efficient approaches:

Leave the LLM and backbone frozen,

Train the mapping on (very) limited training sets to obtain very good results

Simple design choices works best!

ie. passing all perceptual tokens at the input to the LLM

compress perceptual to a few “summary tokens”

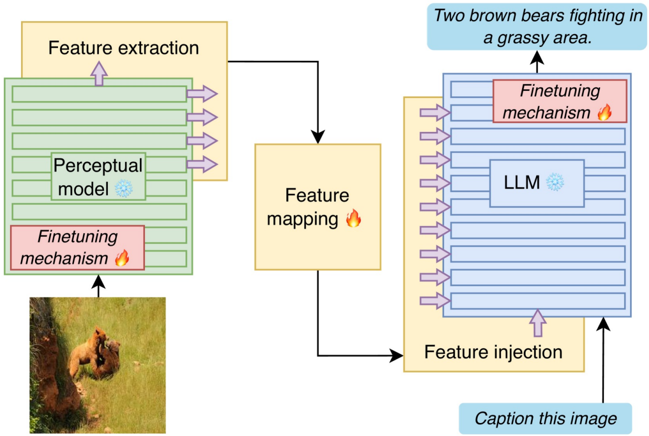
4 times faster to train and on par results

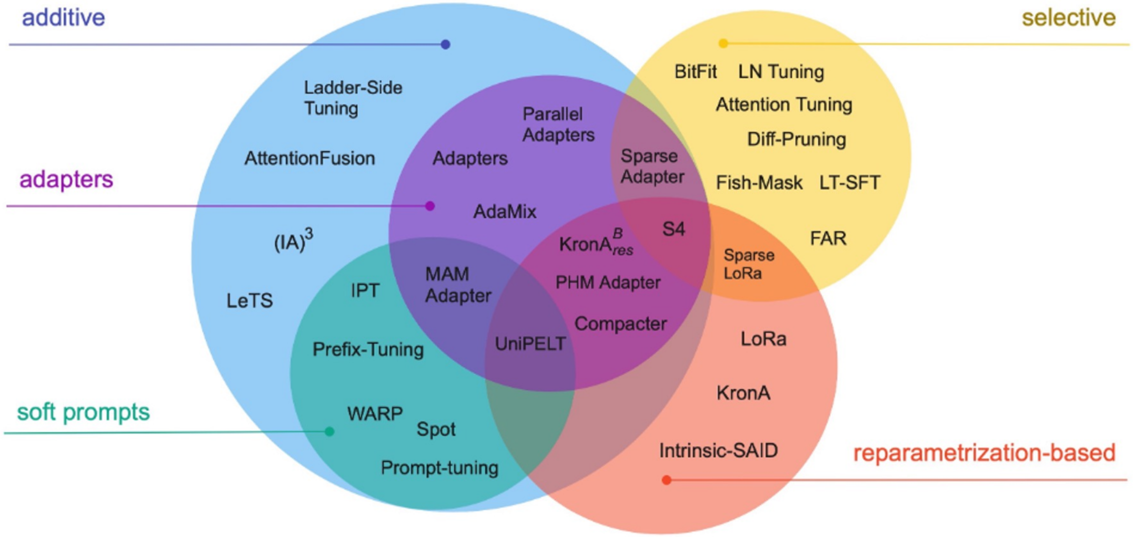
Vision Encoder + LLM Decoder

Many things to do on top of (pretrained) foundations models (if/when available)

Leverage **unimodal** models to build efficient **multimodal** models works well

Efficient finetuning: parameter efficiency, data efficiency, …





Vision Encoder + LLM Decoder

How to get the best VLM?

Relax the **efficiency** constraint

1/ Build a huge multimodal dataset

Vision Encoder + LLM Decoder

How to get the best VLM?

Relax the **efficiency** constraint

1/ Build a huge multimodal dataset

Image-Text Pairs

Tottenham vs Chelsea Live Streaming

Tottenham Spurs vs Chelsea Live Streaming

Multimodal Document

The match between Tottenham Spurs vs Chelsea will kick off from 16:30 at Tottenham Hotspur Stadium, London.

The derby had been played 54 times and the Blues have dominated the Spurs. Out of 54 matches played, Chelsea has won 28 times and Spurs had only won 7 times. The remaining 19 matches had ended in draw.

However, in recent 5 meetings, Spurs had won 3 times where Chelsea had won the other two times. …

+Add synthetized data …

Dataset	Images	% unique images	Docs	Tokens	Open
KOSMOS-1	-	-	71M	-	✗
M3W	185M	-	43M	-	✗
mmc4-ff	385M	60.6%	79M	34B	✓
mmc4	**585M**	-	103M	43B	✓
OBELICS	353M	**84.3%**	**141M**	**115B**	✓

Table 1: General statistics of OBELICS and the current largest alternatives.

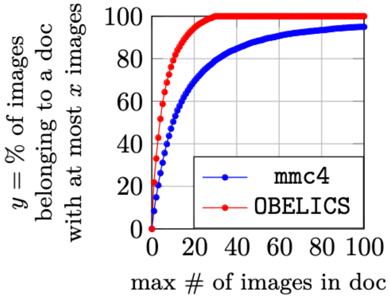


Figure 3: Distribution of images.

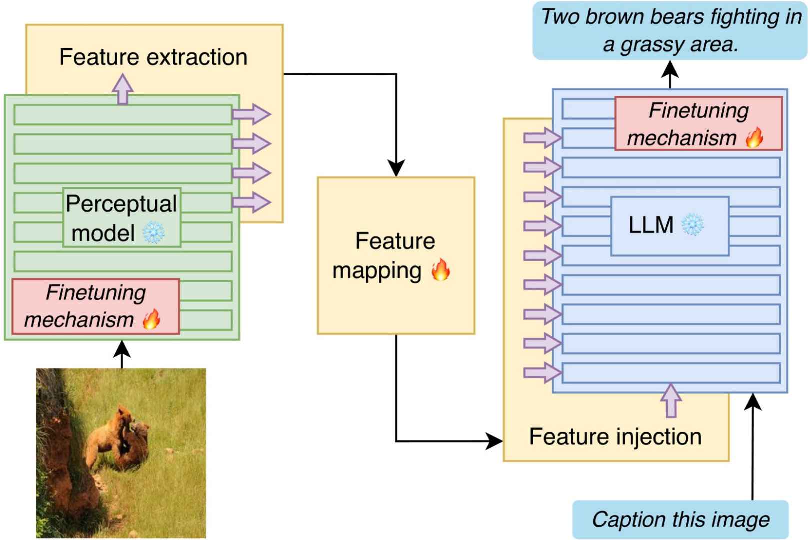
Vision Encoder + LLM Decoder

How to get the best VLM?

Relax the **efficiency** constraint

1/ Build a huge multimodal dataset

2/ Train your best model:



Best architecture?

Vision encoder

Feature Mapping to the LLM input space

Visual tokens (64 in our standard configuration)

interleaved with the input sequence of text embeddings

LLM

Vision Encoder + LLM Decoder

Evaluation very important, not easy for Generative models

Quantitative results:

Model	Size	Archi.	# tokens per image	VQAv2	TextVQA	OKVQA	COCO
OpenFlamingo	9B	CA	-	54.8	29.1	41.1	96.3
Idefics1	9B	CA	-	56.4	27.5	47.7	97.0
Flamingo	9B	CA	-	58.0	33.6	50.0	99.0
MM1	7B	FA	144	63.6	46.3	51.4	**116.3**
Idefics2-base	8B	FA	**64**	**70.3**	**57.9**	**54.6**	116.0

Model	Size	# tokens per image	MMMU	MathVista	TextVQA	MMBench
LLaVA-NeXT	13B	2880	36.2/-	35.3	67.1	70.0
DeepSeek-VL	7B	576	36.6/-	36.1	64.4	73.2
MM1-Chat	7B	720	37.0/35.6	35.9	72.8	72.3
Idefics2	8B	**64**	**43.5/37.9**	**51.6**	70.4	**76.8**
Idefics2	8B	320	43.0/37.7	51.4	**73.0**	76.7

Qualitative results:

Prompt

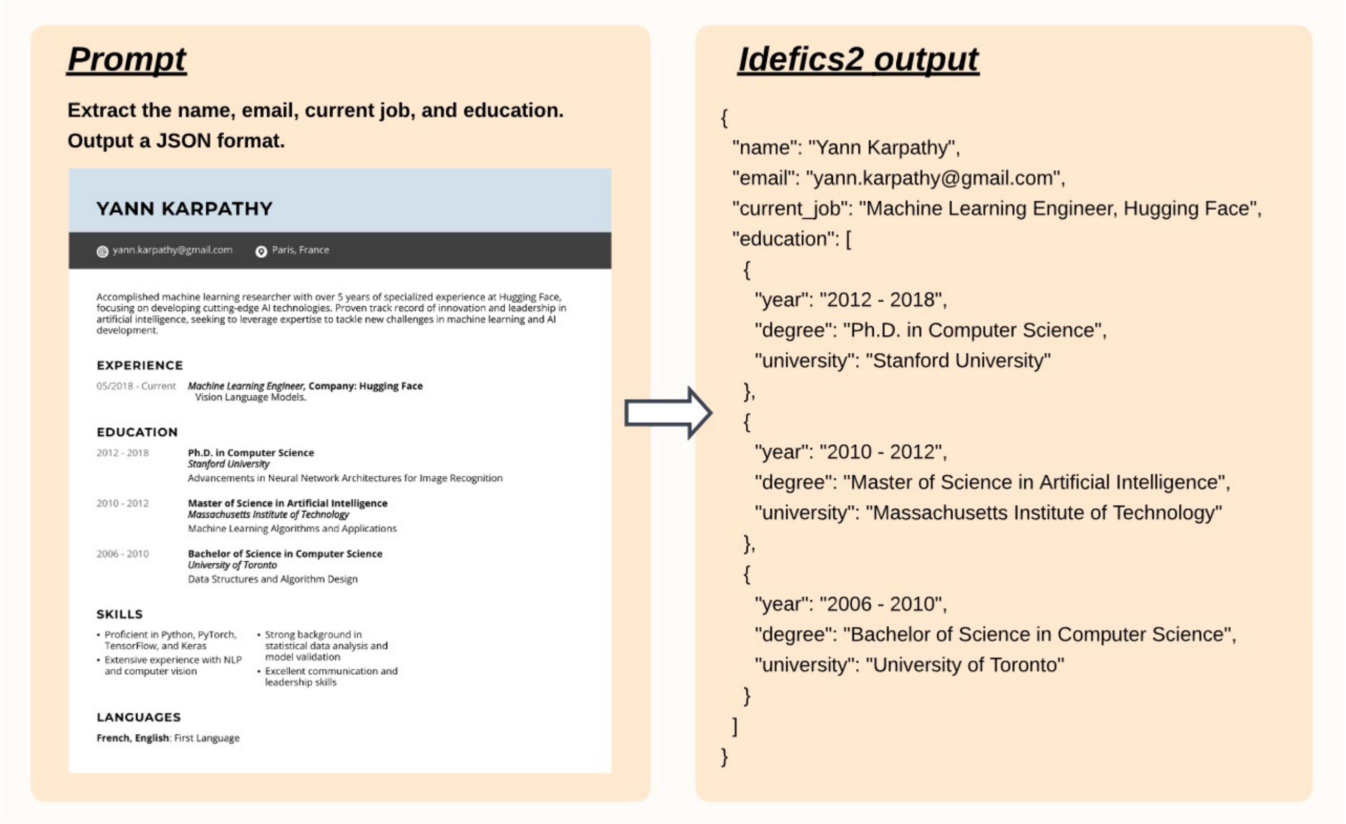
Describe the image

Idefics2 output

The image shows two golden retriever puppies sitting in a field of flowers. They are sitting next to each other, looking at the camera, and appear to be very happy. The puppies are adorable, and their fur is a beautiful golden color. The flowers surrounding them are yellow and add a vibrant touch to the scene.

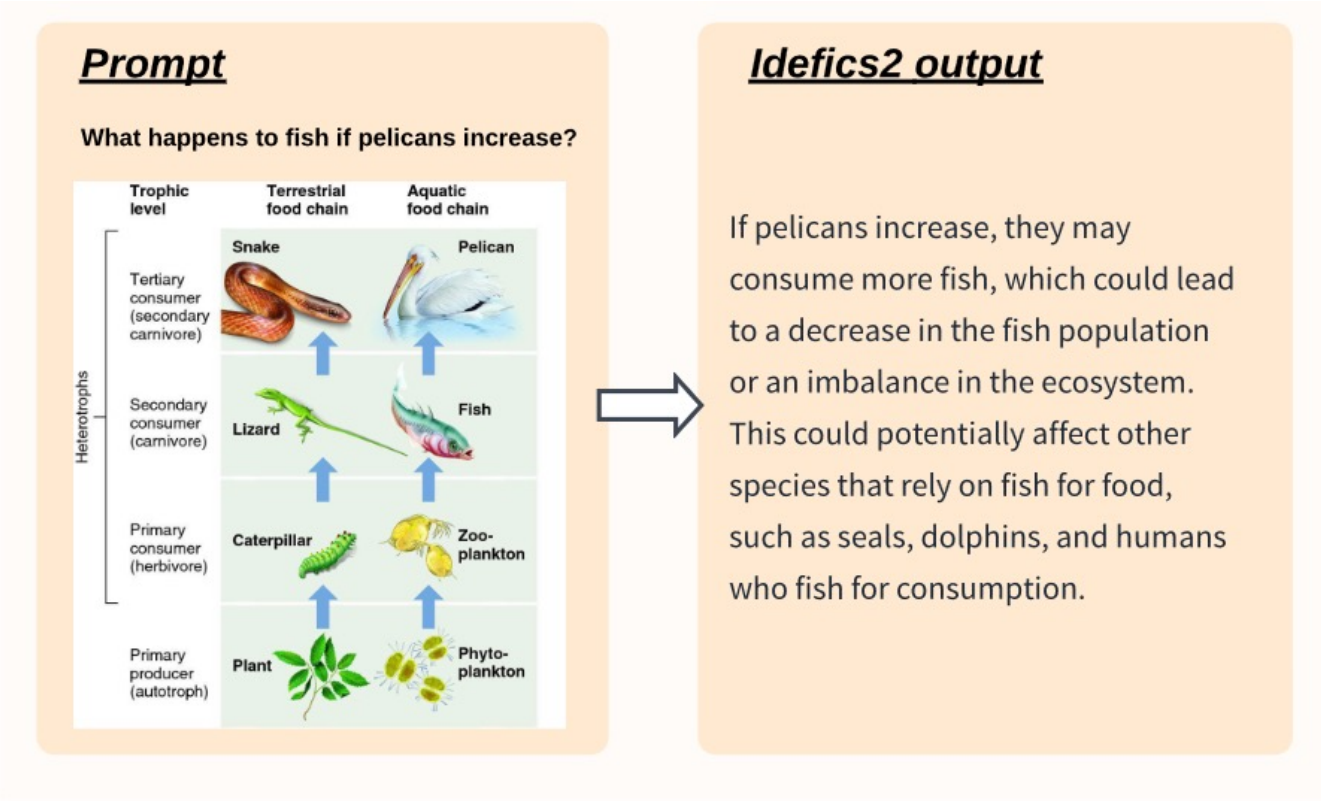
Vision Encoder + LLM Decoder

Qualitative results:



Vision Encoder + LLM Decoder

Qualitative results:

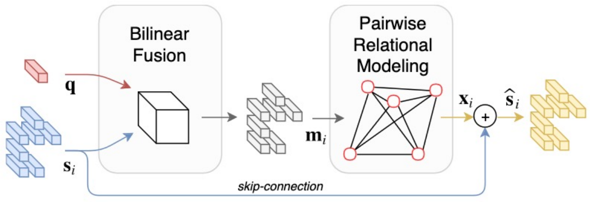


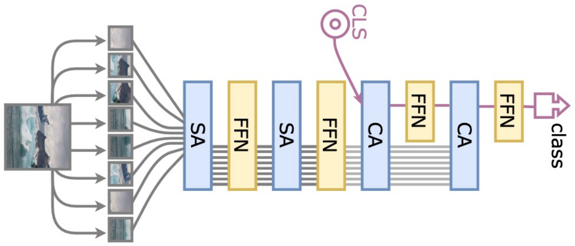
Conclusion -- Vision-Language framework

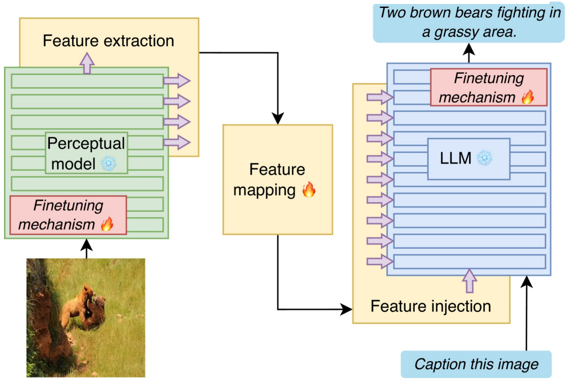
- Architectures/models

  Transformers: the end?

  Vision-Language interaction/representation







- Learning VLM: data, loss, optim., evaluation, generalization …

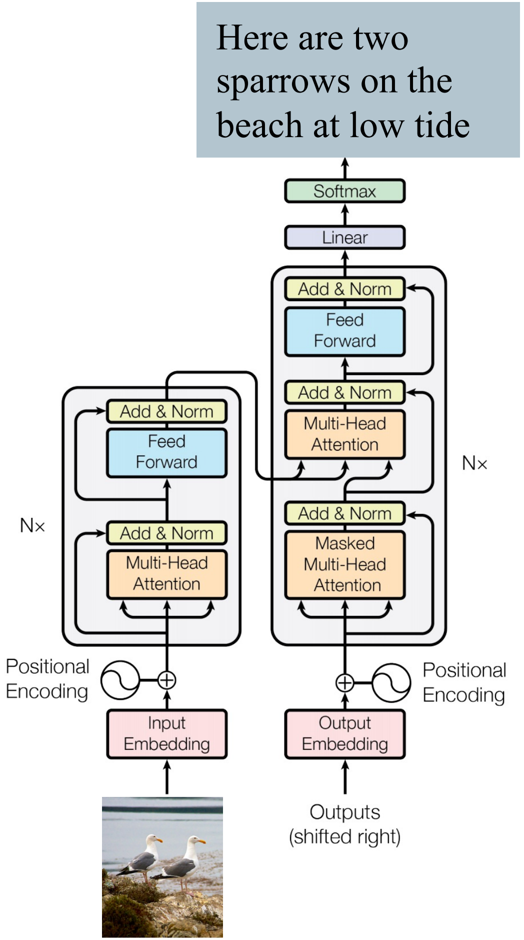
- Monitoring, explainability for VLM

And what models if output different from text only?

Outline

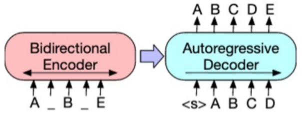
1. From classification to detection, segmentation, …
2. Vision-Language Models in the era of LLMs

   - Unimodal models with connection
   - **One model for all**

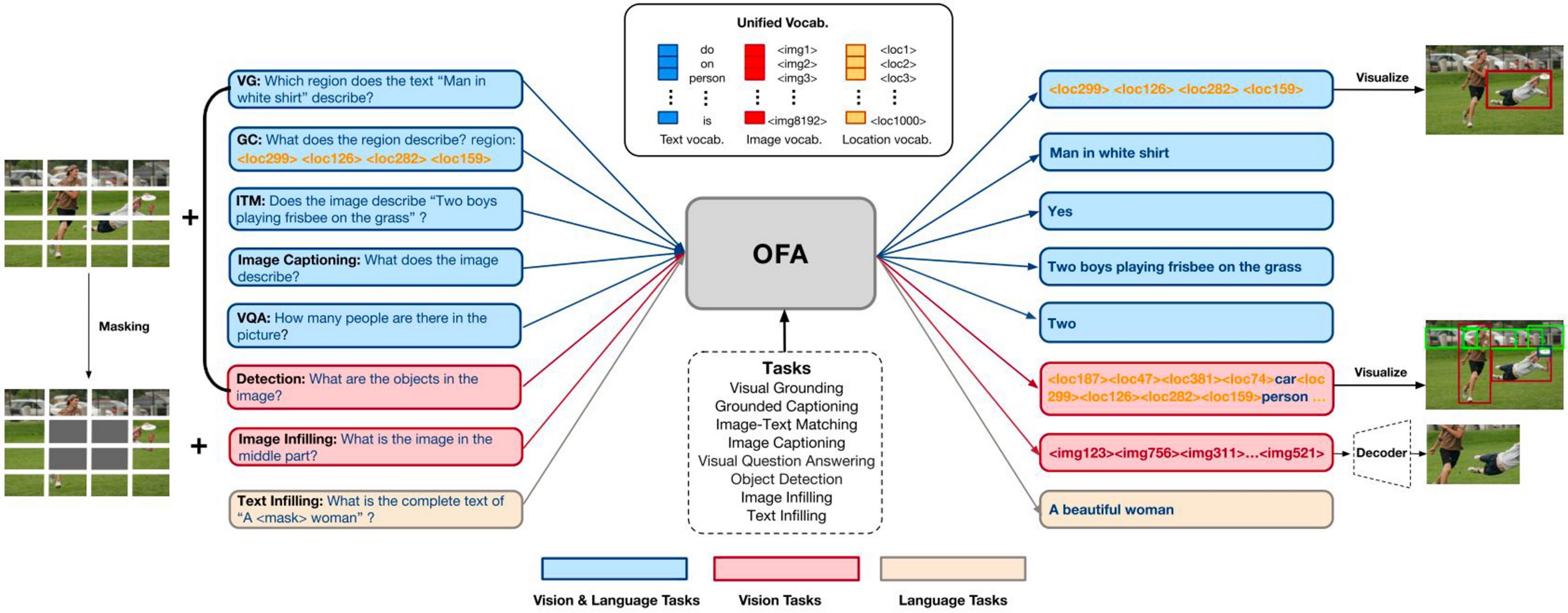


1M4all

One model with:
many inputs / many outputs / many tasks



(c) BART: Inputs to the encoder need not be aligned with decoder outputs, allowing arbitary noise transformations. Here, a document has been corrupted by replacing spans of text with mask symbols. The corrupted document (left) is encoded with a bidirectional model, and then the likelihood of the original document (right) is calculated with an autoregressive decoder. For fine-tuning, an uncorrupted document is input to both the encoder and decoder, and we use representations from the final hidden state of the decoder.



NeurIPS 2023/24

4M: Massively Multimodal Masked Modeling

David Mizrahi[1,2*] Roman Bachmann[1*] Oğuzhan Fatih Kar[1]
Teresa Yeo[1] Mingfei Gao[2] Afshin Dehghan[2] Amir Zamir[1]
[1]Swiss Federal Institute of Technology Lausanne (EPFL) [2]Apple