# Multimodality

CS6120 Natural Language Processing

Northeastern University

Si Wu

Some slides are adapted from Yejin Choi's lecture on multimodality

# Logistics

- Friday we will have guest lecturer Lucy Li.
- Make sure you sign up for presentation:
    - Any date any time is fine. You can sign up for the 1:25pm session.
    - Only sign up once.
    - All group members need to speak during the presentation.
    - This is graded.
- Next Tuesday is the final lecture on topics you all suggested. Go to Ed to suggest things you want to learn.
- Today's topic: multimodality.

# How Humans Interact with the World

We perceive, understand, and interact with the world using multiple senses:

**Sight**

👁️

**Hearing**

👂

**Touch**

✋

**Smell**

👃

**Taste**

👄

# How Humans Interact with the World

- Each sense provides a different type of information about our environment.
- Our brains naturally combine these sensory inputs to build a complete understanding of what’s happening around us..

Tastes **sweet**!

Smells **crisp, fruity, sweet, refreshing ?**

It is **red**

The skin is **waxy,** and the shape is **round**

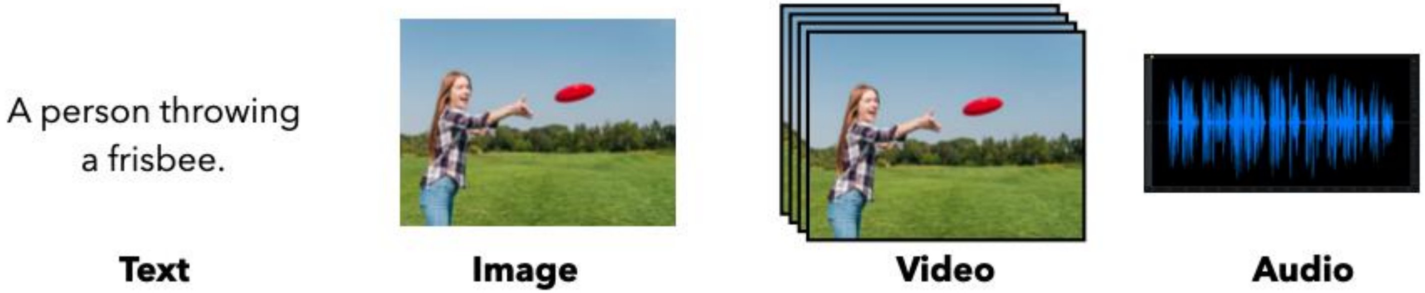
What is it?

# From Human Senses to the Idea of “Modality”

- Each **sense** can be thought of as a **modality**
  - **Modality:** a specific channel through which information is received.
    - Visual
    - Aural
    - Gestural
    - Linguistic, and many more.
- Combining different modalities creates richer, more accurate understanding
  - e.g., seeing someone speak + hearing their voice, is better than just hearing someone’s voice over the phone.
- It’s natural that we want to process information mimicing how we experience the world: using information from different modalities.

# Digital Modalities

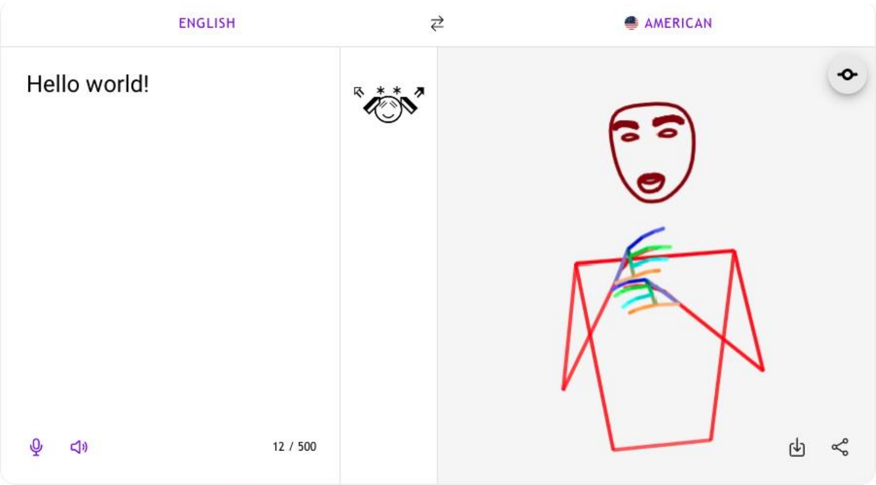
- Even in digital formats, there are many different information channels, which parallels human modalities:
  - Text
  - Image
  - Audio: speech, music
  - Video: combines visual and audio info
  - Other sensor data: motion, touch, location, etc.



- Just as humans combine senses, digital models can combine these modalities to understand context more deeply

# Multimodal AI

- Research has shown that generally, utilizing different modalities can improve the performance
- In some research areas, it’s natural to use different modalities
  - Robotics
  - Sign language processing: images or video, along with text



ENGLISH ⇄ AMERICAN

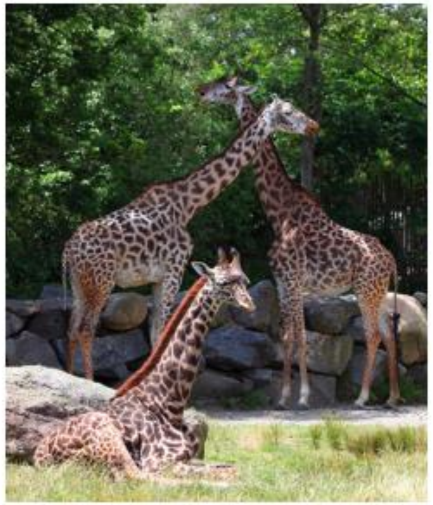
Hello world!

Try sign translate to experience state-of-the art-sign language translation technology.

https://research.sign.mt/

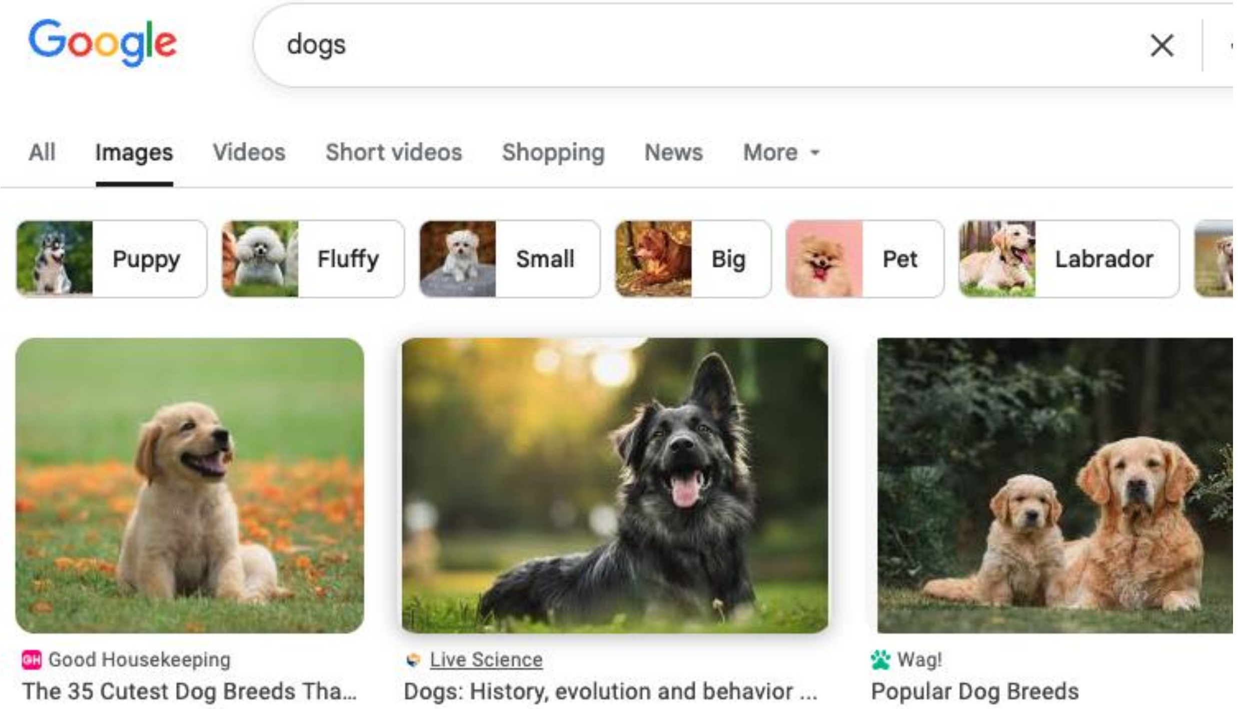
# Examples of multimodal tasks

## Visual reasoning, VQA



**How many giraffes are standing up?**

## Text-to-image retrieval



Good Housekeeping
The 35 Cutest Dog Breeds Tha...

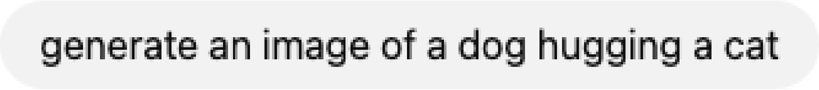
Live Science
Dogs: History, evolution and behavior ...

Wag!
Popular Dog Breeds

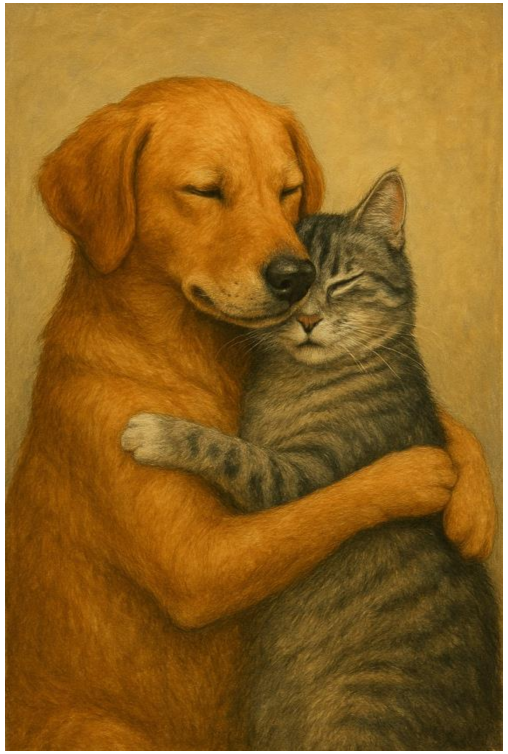
First figures from
https://arxiv.org/abs/2508.17298

# Examples of multimodal tasks

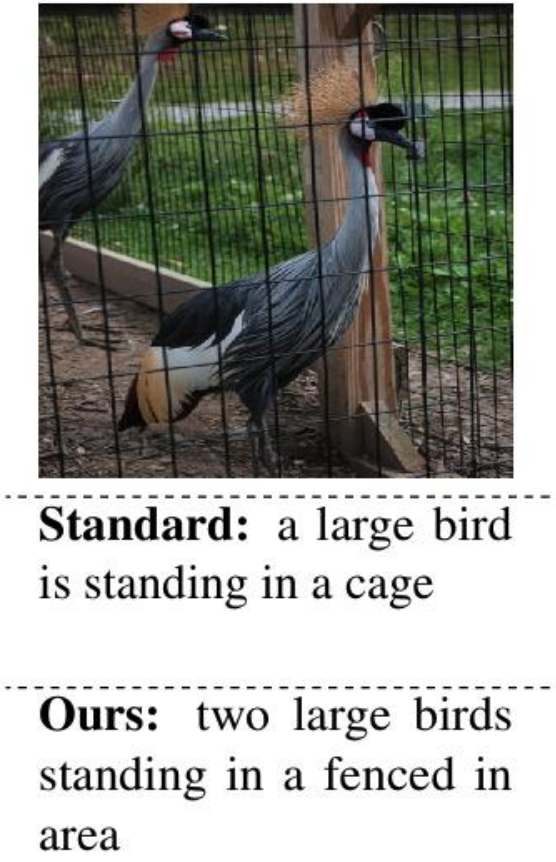
## Text to image generation



generate an image of a dog hugging a cat



## Image captioning



**Standard:** a large bird is standing in a cage

**Ours:** two large birds standing in a fenced in area

figure on image captioning is from https://arxiv.org/abs/1906.05963

# The structure of this lecture

- In this lecture, we will only focus on **text** and **image**.
- We will discuss the following topics
  - Firstly, how to **align the text and image space**?
    - A specific example is CLIP
  - Then, we will look at some image+text understanding examples
    - VQA
  - Finally, we will brief talk about **text to image generation**

# The structure of this lecture

**Image & Text Alignment**

A person throwing
a frisbee.

**Image + Text Understanding**

What is the object
being thrown?

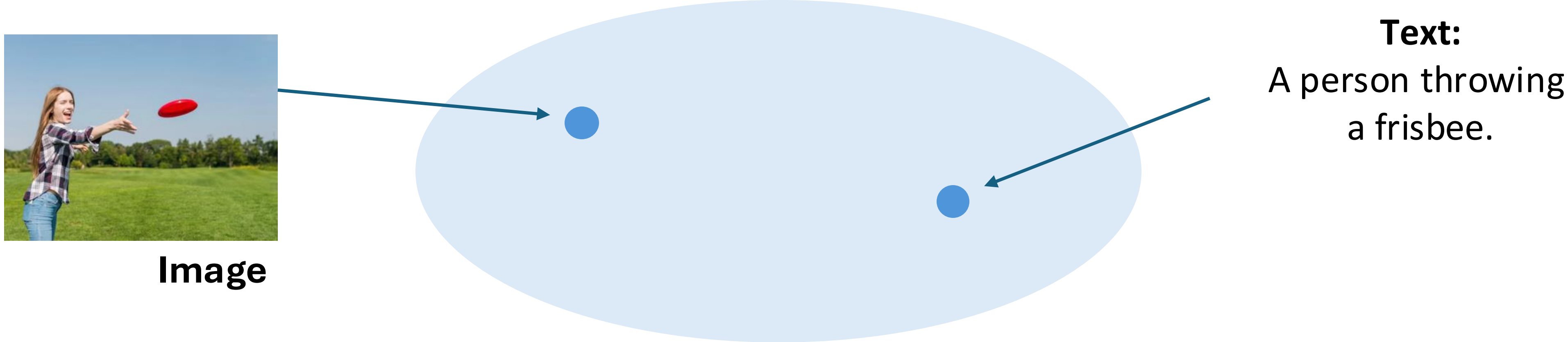
A frisbee

**Text to Image Generation**

A person throwing
a frisbee.

# Steps of Image-Text Alignment

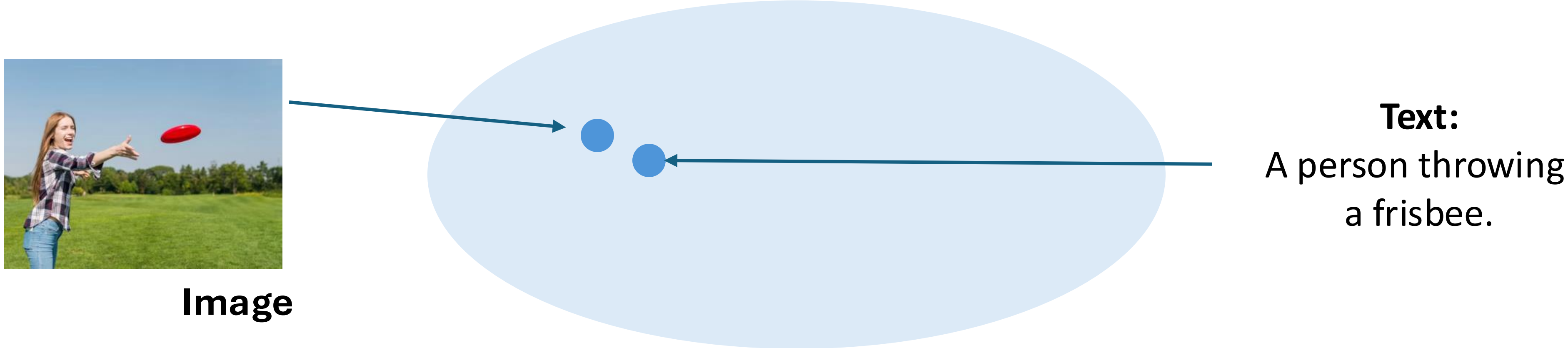
- **Step1:** Encode different modalities into shared embeddings.



Image

**Text:**
A person throwing
a frisbee.

- **Step2:** Bring embeddings of image and text that shared the same meaning closer together.

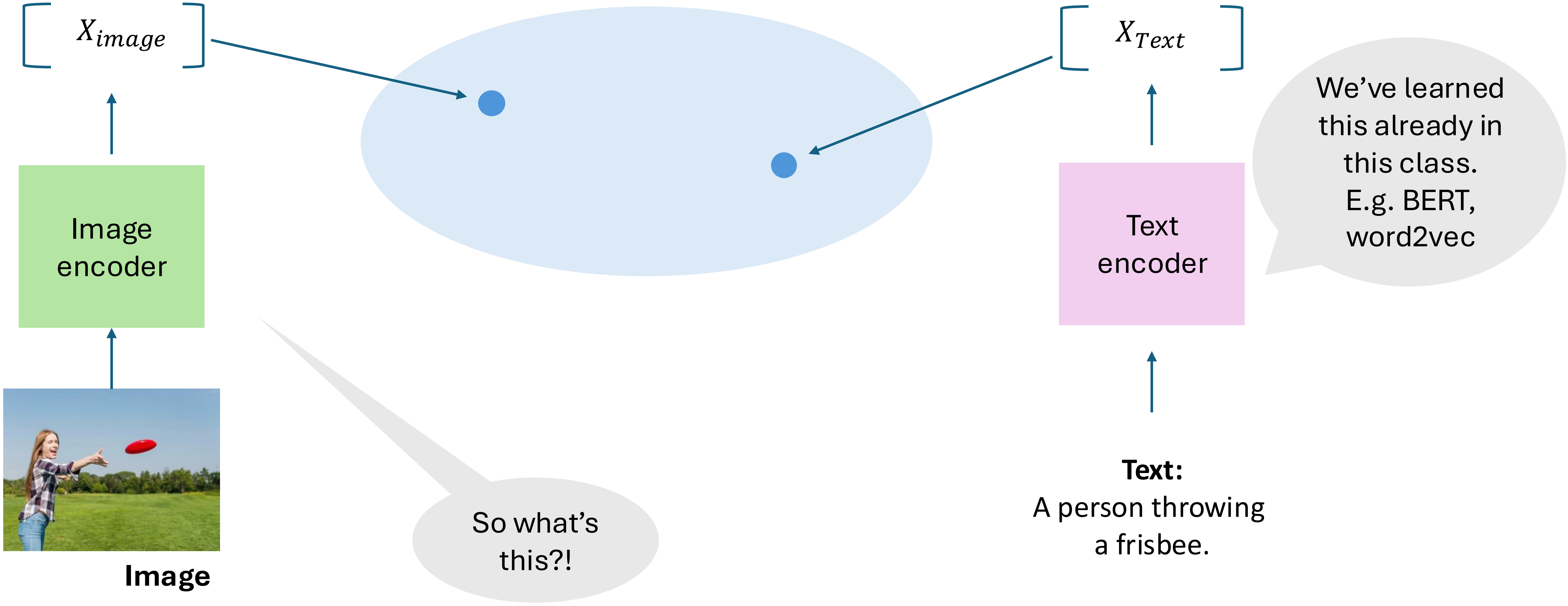


Image

**Text:**
A person throwing
a frisbee.

# Step 1: encode different modalities

- **Step1:** Encode different modalities into shared embeddings.



$X_{image}$

Image encoder

**Image**

So what’s this?!

$X_{Text}$

Text encoder

**Text:**
A person throwing a frisbee.

We’ve learned this already in this class. E.g. BERT, word2vec

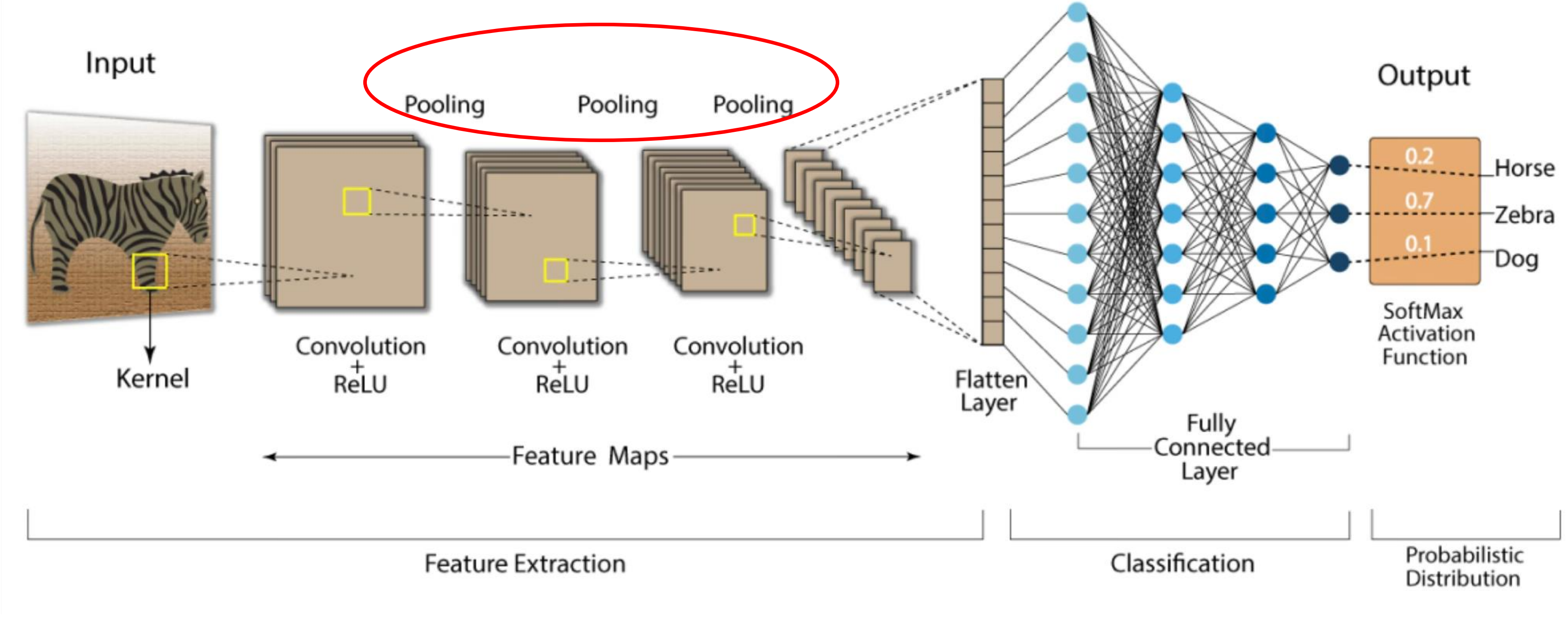
# Vision Encoder: Convolutional Neural Networks

- A classic image encoder uses **Convolutional neural network (CNN)**
- Remember images are different from text. Images are made of pixels.
  - Each pixel has a color
  - An image has height and width, e.g. 16 x 16
  - An images has multiple color channels: RGB, e.g. (3,16,16)

# Vision Encoder: Convolutional Neural Networks

- Briefly, if you didn’t learn about **Convolutional neural network (CNN)** before…

**Pooling**: Reduce dimensionality of the convoluted features for efficient computation



Here’s a good resource for CNN
https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks

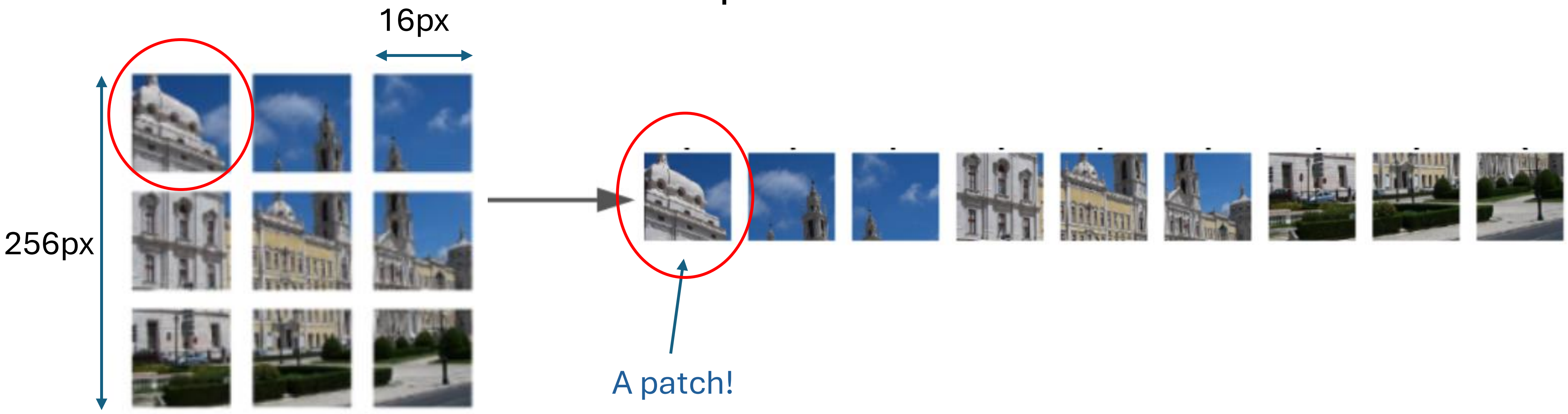
# The Vision Transformer: Image Encoding via Patch Tokens

- Alternatively, we can use a transformer model to encode images like in NLP…
- But images have height and width, and are made of rows of pixels, how to process them?

# The Vision Transformer: Image Encoding via Patch Tokens

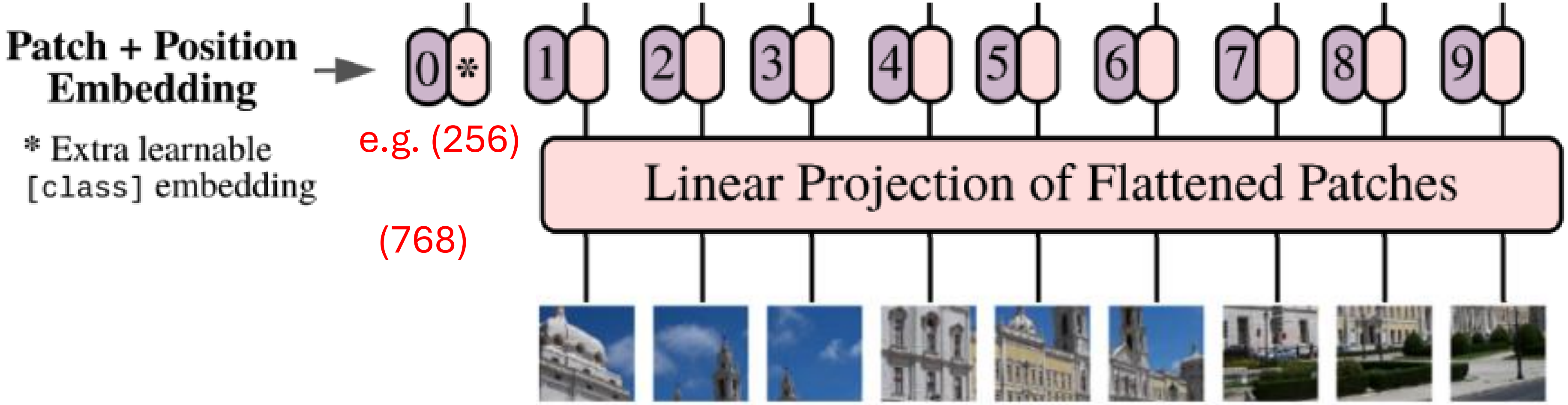
First, we split an image into fixed sized "**patches**", for example, the size could be 16 x 16 px, while the original image is 256 x 256 px

--> each patch is a "**token**" in the NLP world



# The Vision Transformer: Image Encoding via Patch Tokens

Once the original image is turned into patches or "visual tokens", we can process each patch using, for example CNN (modern ViT) or linear projection (original ViT), to encode a patch


Patch + Position Embedding
* Extra learnable [class] embedding
0
*
1
2
3
4
5
6
7
8
9
Linear Projection of Flattened Patches

e.g. (256)

(768)

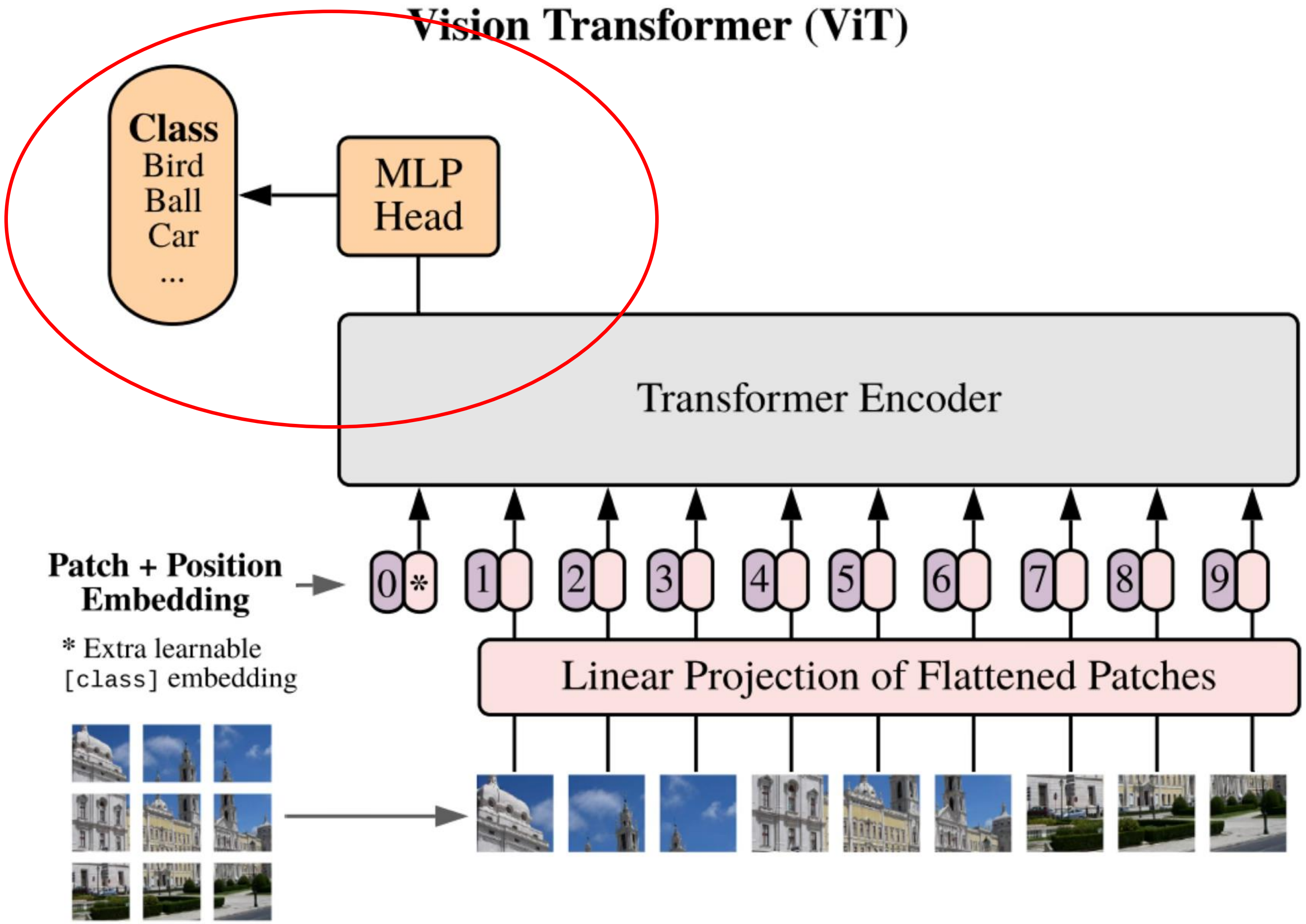
Wx+b

(3,16,16)

# The Vision Transformer: Image Encoding via Patch Tokens

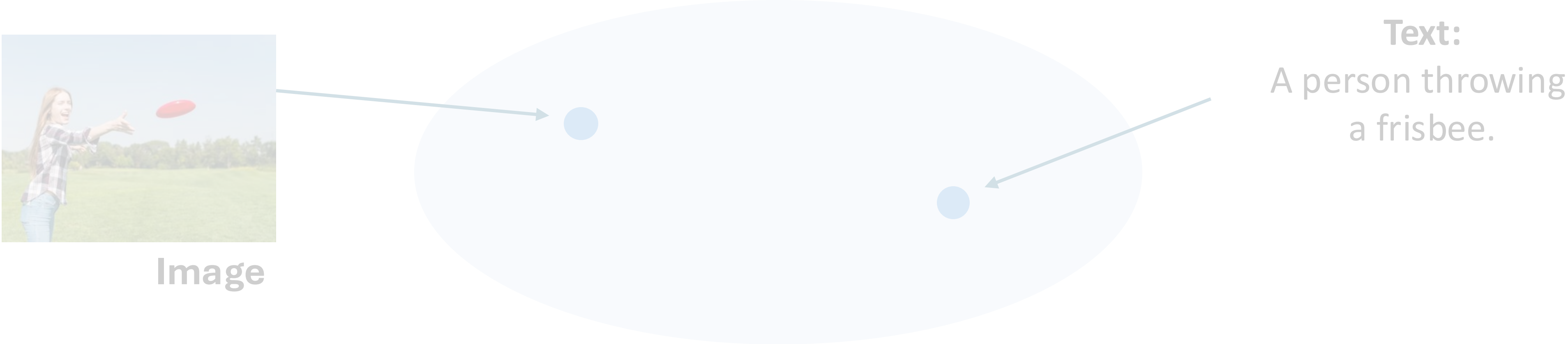
Once you have the embedding of these patches, you can continue like how you would with text.

For example, here we are doing a classification task, so we have a task-specific head, the MLP head.



# Steps of Image-Text Alignment

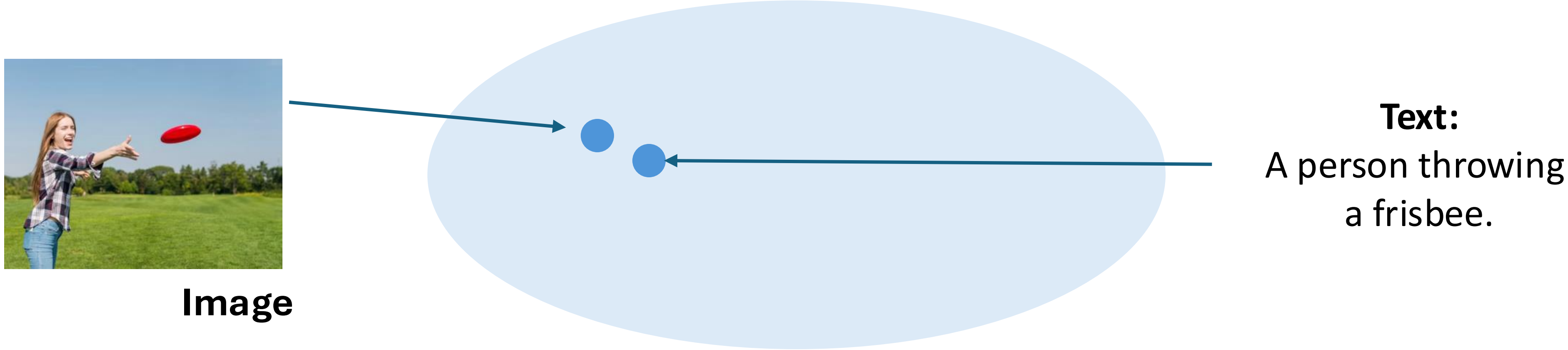
- **Step1:** Encode different modalities into shared embeddings.



Image

**Text:**
A person throwing
a frisbee.

- **Step2:** Bring embeddings of image and text that shared the same meaning closer together.



Image

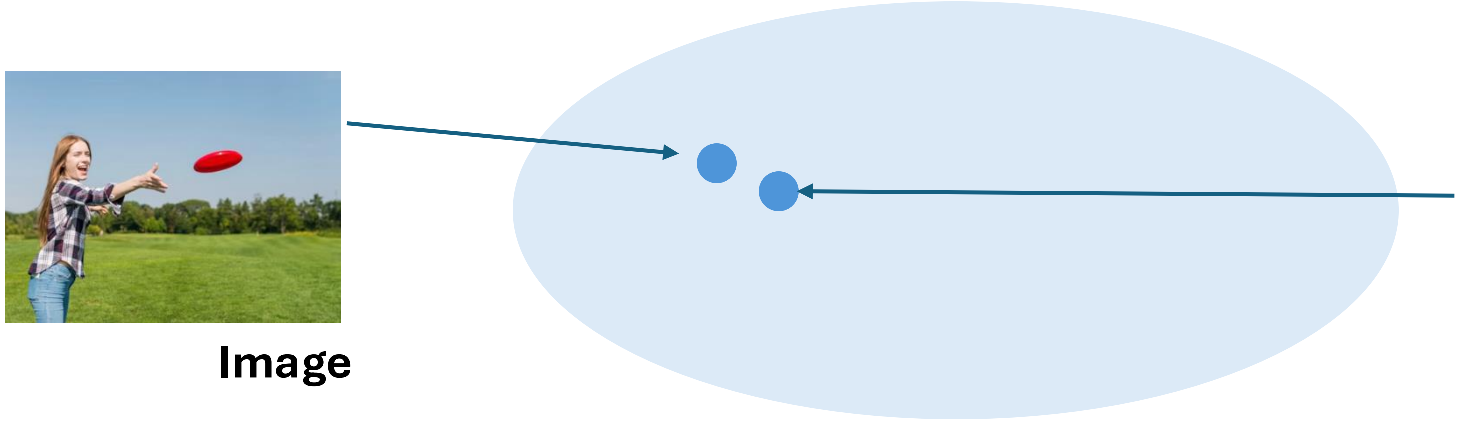
**Text:**
A person throwing
a frisbee.

# Step 2: bring matched image-text pairs closer

**Step2:** Bring embeddings of image and text that shared the same meaning closer together.

→Direct translation of this: **minimize the loss, maximize similarity**!

But how? → **Contrastive learning!**



**Image**

**Text:**
A person throwing
a frisbee.

# Contrastive learning

- **Contrastive Learning**: learn the shared embedding by contrasting **positive** and **negative** pairs of instances
  - **Positives**: matched image-text pairs
  - **Negatives**: image-text from mismatched instances

+

**Text:**
A person throwing
a frisbee.

The idea is to bring the pairs with the same meaning closer, and push the mismatched apart

Positive pair!

Negative pair!

−

**Text:**
A person eating an apple.

# Recall Cosine similarity from previous lectures

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\| \|\mathbf{B}\|} = \frac{\sum\limits_{i=1}^{n} A_i B_i}{\sqrt{\sum\limits_{i=1}^{n} A_i^2} \cdot \sqrt{\sum\limits_{i=1}^{n} B_i^2}},$$

- Recall:
  - Because these are high dimensional vectors, we don't really work on (Euclidean) "**distance**" but we use cosine similarity
  - Using the **angle** between the two vectors, not the actual distance
- So here:
  - **Increase** cosine similarity between **matching** image-text pairs
  - **Decrease** cosine similarity between **mismatch** image-text pairs

**Text:**
A person throwing
a frisbee.

Sim(Positive pair) > Sim(Negative pair)

**Text:**
A person eating an apple.

# Loss functions for step 2

- There are multiple options here:
  - **Triplet loss**:
    - (Anchor, Positive, Negative)
  - Cross-entropy loss
    - We will get into it when we talk about CLIP

# Triplet loss

- Triplet loss learns to
  - minimize the distance between the anchor and positive
  - maximize the distance between the anchor and negative
  - If the negative is already far enough, loss = 0

with the following equation:
Anchor, Positive, Negative

$$L = \sum_{i=1}^{m} \max\left( \underbrace{\| f(A^{(i)}) - f(P^{(i)}) \|_2^2}_{} - \underbrace{\| f(A^{(i)}) - f(N^{(i)}) \|_2^2}_{} + \enclose{circle}{\alpha}, 0 \right)$$

The margin parameter specifies the minimum required difference between (A, P) and (A,N).

# Triplet loss

Or more simply, triplet loss enforces:

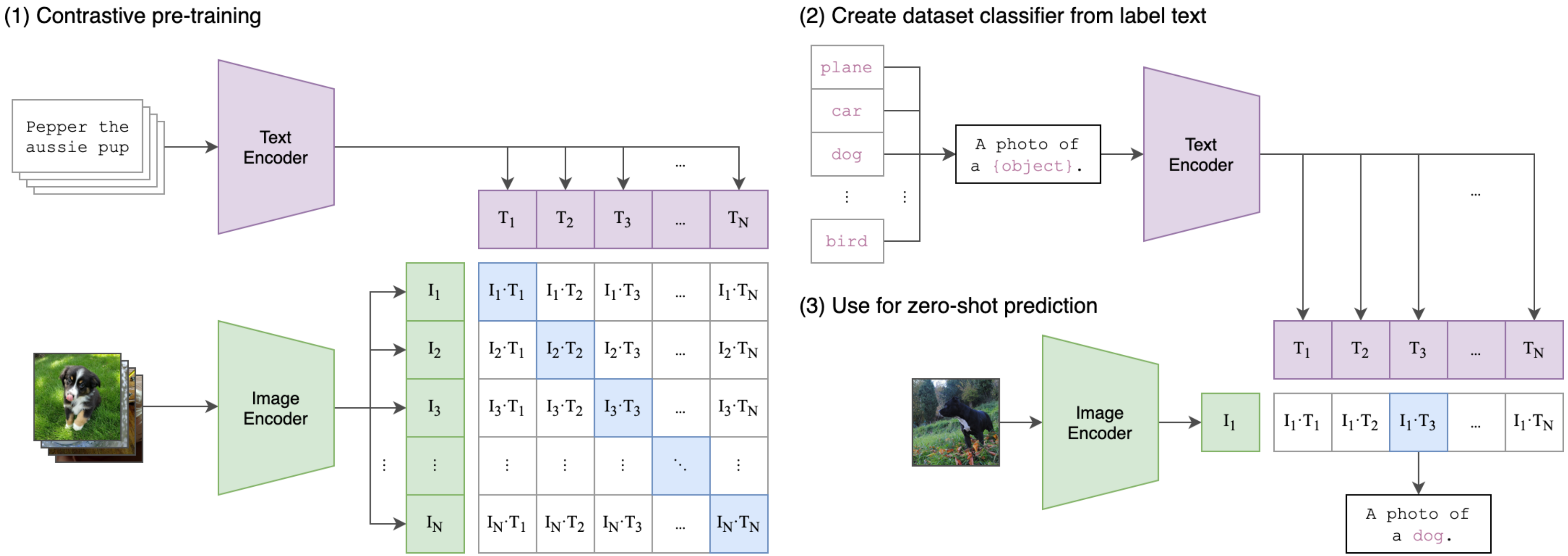
$$d(A,N) \quad >= \quad d\,(A,P) + margin$$

$$\downarrow$$

$$Loss = \max(0, \quad d(A,P) - d(A,N) + margin)$$

# A specific example: CLIP

- **C**ontrastive **L**anguage-**I**mage **P**re-training (CLIP)
- It uses a different loss → InfoNCE-style loss



(1) Contrastive pre-training

(2) Create dataset classifier from label text

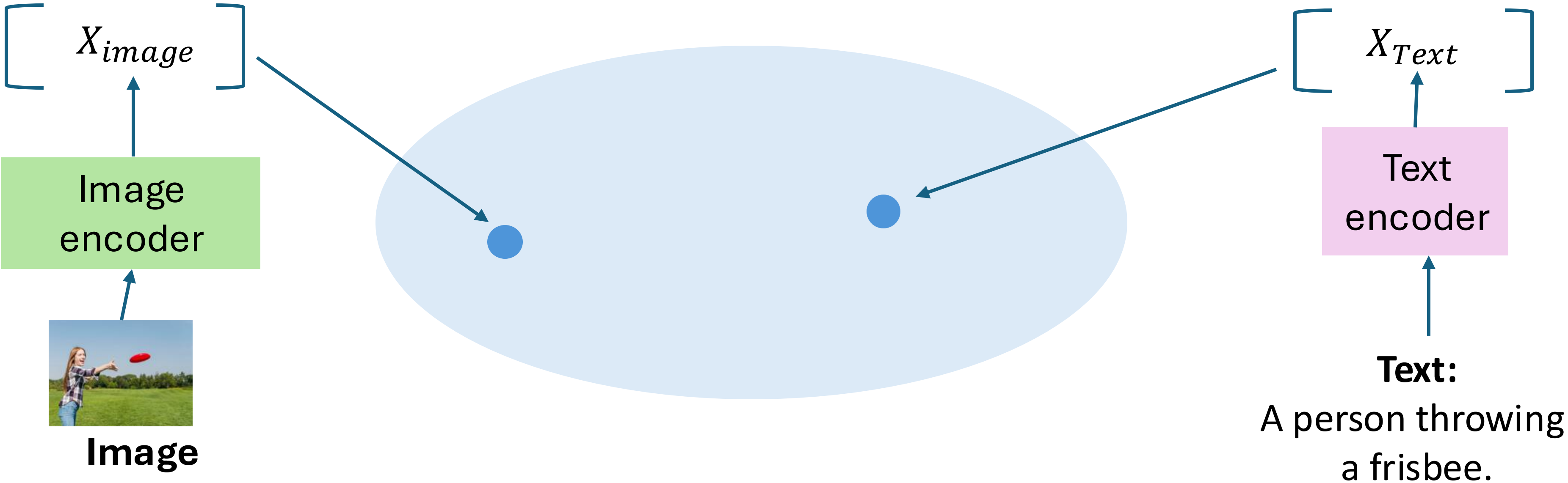
(3) Use for zero-shot prediction

|  | $T_1$ | $T_2$ | $T_3$ | ... | $T_N$ |
|---|---|---|---|---|---|
| $I_1$ | $I_1 \cdot T_1$ | $I_1 \cdot T_2$ | $I_1 \cdot T_3$ | ... | $I_1 \cdot T_N$ |
| $I_2$ | $I_2 \cdot T_1$ | $I_2 \cdot T_2$ | $I_2 \cdot T_3$ | ... | $I_2 \cdot T_N$ |
| $I_3$ | $I_3 \cdot T_1$ | $I_3 \cdot T_2$ | $I_3 \cdot T_3$ | ... | $I_3 \cdot T_N$ |
| ⋮ | ⋮ | ⋮ | ⋮ | ⋱ | ⋮ |
| $I_N$ | $I_N \cdot T_1$ | $I_N \cdot T_2$ | $I_N \cdot T_3$ | ... | $I_N \cdot T_N$ |

|  | $T_1$ | $T_2$ | $T_3$ | ... | $T_N$ |
|---|---|---|---|---|---|
| $I_1$ | $I_1 \cdot T_1$ | $I_1 \cdot T_2$ | $I_1 \cdot T_3$ | ... | $I_1 \cdot T_N$ |

# CLIP: step 1

- Same as what we said so far, CLIP will map text and images to the same space.
- Image encoder: ResNet or Vision Transformer variant
- Text encoder: transformer
- Then each encoder output go through a linear projection to ensure they are of the same dimension



$X_{image}$

Image encoder

**Image**

$X_{Text}$

Text encoder

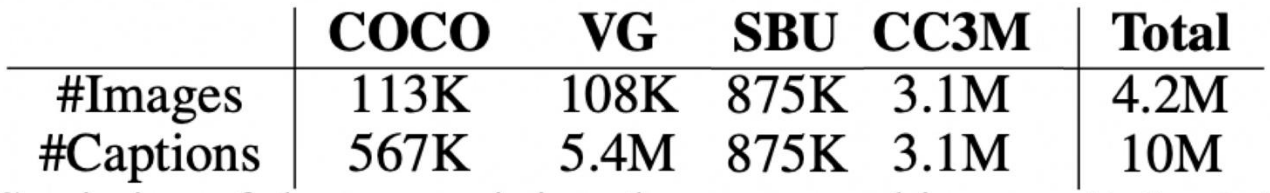
**Text:**
A person throwing
a frisbee.

# CLIP: step 2

- Then image and text pairs are divided into batches
- For each batch, we compute a similarity matrix ($N_{image}$x $N_{text}$)
- The loss has two parts:
    - **For each images**, you bring its paired text closer (and push all other texts away)
    - **For each text**, you bring its paired image closer (and push all other images away)
    - We use cosine similarity (normalized dot product) for the two vectors.
    - It is a symmetric InfoNCE-style cross-entropy contrastive loss

# Image-Text Training Dataset

- Previous Image-Text Pre-Training Dataset
  - Leverage filtered, carefully annotated dataset for academic research
  - 10M was considered as “large-scale” pre-training



| | **COCO** | **VG** | **SBU** | **CC3M** | **Total** |
|---|---|---|---|---|---|
| #Images | 113K | 108K | 875K | 3.1M | 4.2M |
| #Captions | 567K | 5.4M | 875K | 3.1M | 10M |

Table 3.2: Statistics of the pre-training datasets used in a typical academic setting.

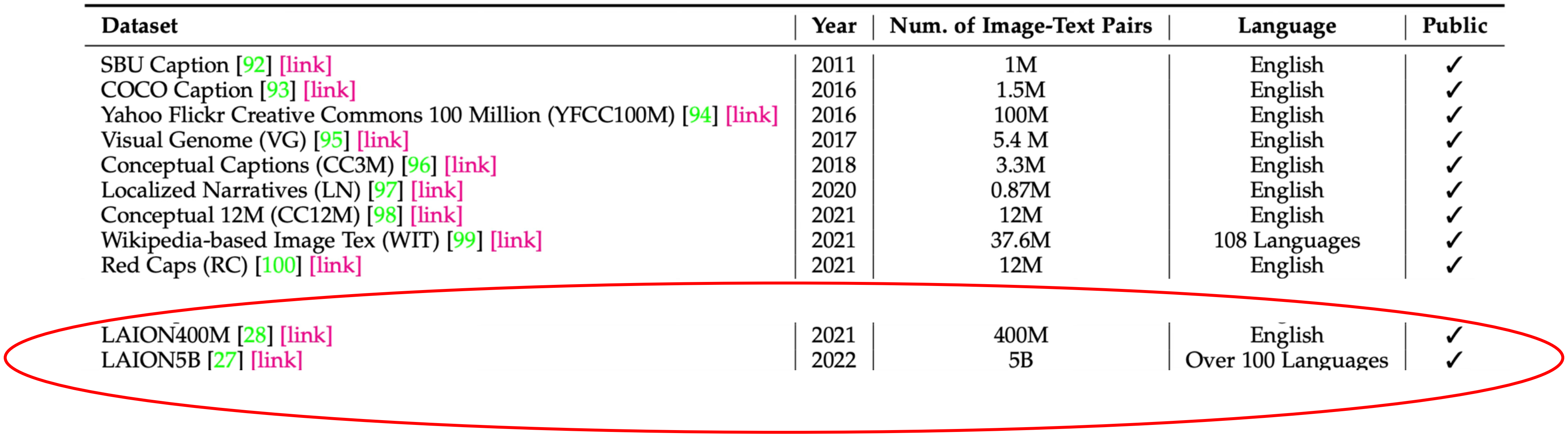
# Image-Text Training Dataset

- Previous Image-Text Pre-Training Dataset
  - Leverage filtered, carefully annotated dataset for academic research
  - 10M was considered as “large-scale” pre-training

- **CLIP: 400M** Image-Text pairs crawled from web
  - Unfiltered, highly varied, and highly noisy data
  - Covers much more diverse concepts and images

# Image-Text Training Dataset

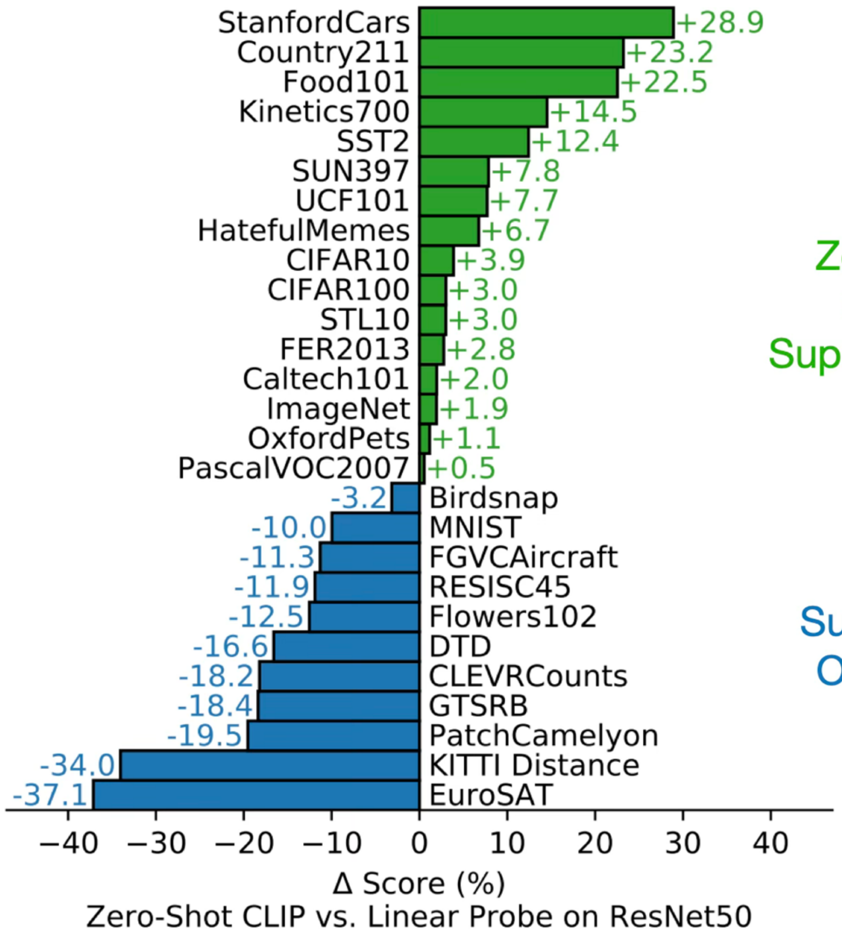
- **CLIP Training Data:** 400M Image-Text Pairs crawled from the web
  - Wasn’t open to public for training
- **LAION Dataset**: 400M/5B Image and alt-text attributes
  - OpenCLIP (UW): Open-sourced CLIP training on LAION dataset



| Dataset | Year | Num. of Image-Text Pairs | Language | Public |
|---|---|---|---|---|
| SBU Caption [92] [link] | 2011 | 1M | English | ✓ |
| COCO Caption [93] [link] | 2016 | 1.5M | English | ✓ |
| Yahoo Flickr Creative Commons 100 Million (YFCC100M) [94] [link] | 2016 | 100M | English | ✓ |
| Visual Genome (VG) [95] [link] | 2017 | 5.4 M | English | ✓ |
| Conceptual Captions (CC3M) [96] [link] | 2018 | 3.3M | English | ✓ |
| Localized Narratives (LN) [97] [link] | 2020 | 0.87M | English | ✓ |
| Conceptual 12M (CC12M) [98] [link] | 2021 | 12M | English | ✓ |
| Wikipedia-based Image Tex (WIT) [99] [link] | 2021 | 37.6M | 108 Languages | ✓ |
| Red Caps (RC) [100] [link] | 2021 | 12M | English | ✓ |
| LAION400M [28] [link] | 2021 | 400M | English | ✓ |
| LAION5B [27] [link] | 2022 | 5B | Over 100 Languages | ✓ |

# Text Supervision Enables Strong Zero-Shot Performance in Vision Tasks

- Large-Scale Training on Noisy Image-Text Data → Great Zero-Shot Performance
- Zero-Shot CLIP is competitive with fully supervised Resnet50 in Image Classification
    - Linear Probe: Train linear layer on top of fixed, pre-trained embeddings.



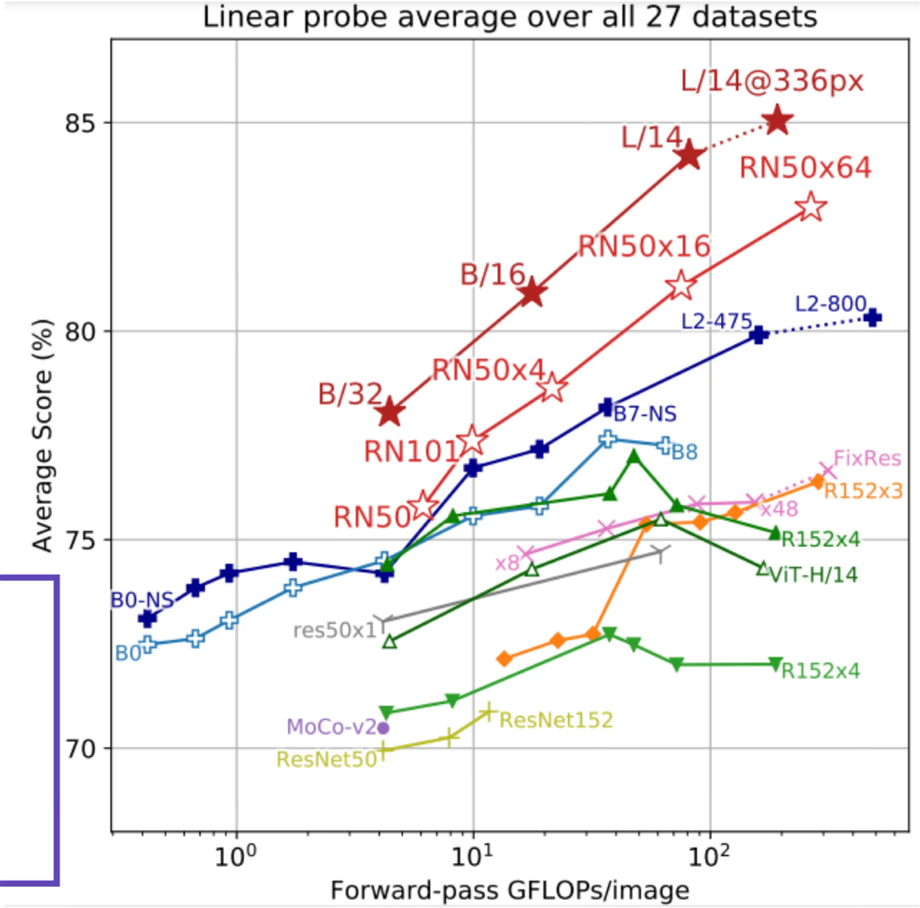
Zero-Shot CLIP vs. Linear Probe on ResNet50

Zero-Shot CLIP Outperforms Supervised baseline.

Supervised baseline Outperforms CLIP.

# CLIP vs Unimodal Visual Representations

- Linear Probe performance v.s. computer vision models
- CLIP provides visual representations  with better transferability



# Language + vision

**Image & Text Alignment**

A person throwing a frisbee.

**Image + Text Understanding**

What is the object being thrown?

A frisbee

**Text to Image Generation**

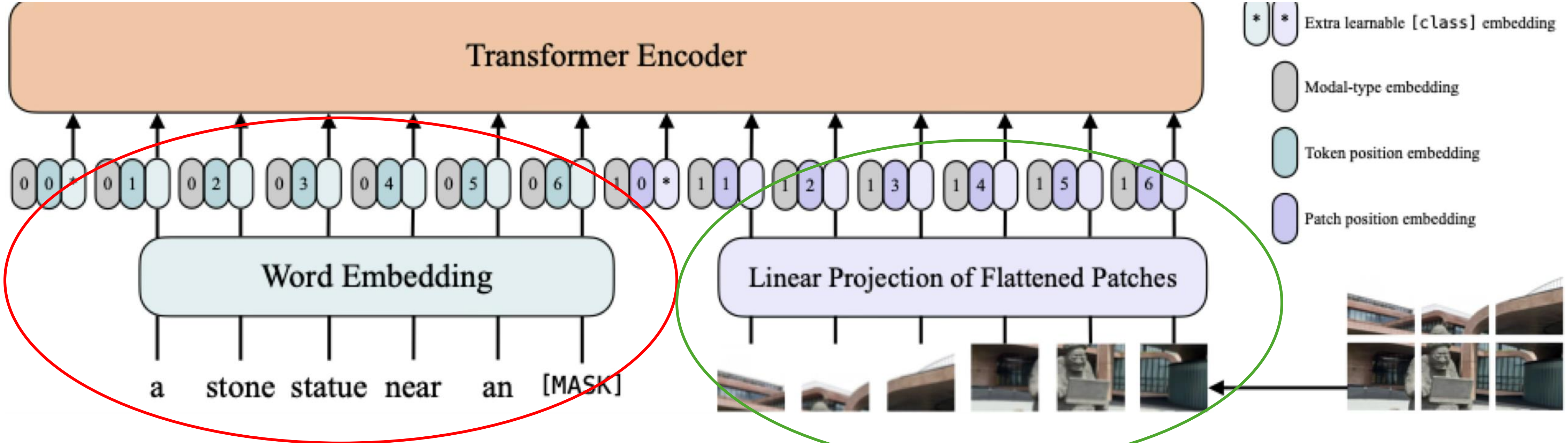
A person throwing a frisbee.

# CLIP is great but

- Simply putting different modalities in the same space is not going to good enough for more complex task that questions details in the image
    - Such as VQA and other visual reasoning tasks
    - For example: CLIP is not powerful enough to answer “how many apples are in this picture”
- So what if we mask the details? Like in mask language modeling?
    - VisualBERT and ViLT doesn’t mask images/pixels but only the text side, but some other variants also mask image patches.

# One example:
# ViLT: The Vision-Language Transformer

Your input is concatenating:
**[cls, image patches, text tokens]**



# Potential ViLT pretraining objectives

- **Masked Language Modeling** (MLM): Predict labels of masked text tokens.

For example, you can mask

a [MASK] is barking

so that the model learns to use visual context to fill in words, without masking any images

# Potential ViLT pretraining objectives

- **Image-Text Matching** (ITM): Classify if image-text pairs are aligned
  - To teach the model whether an image and a text belong together
  - If it’s a true pair (IMAGE, TEXT) = 1,
    0 otherwise

# Potential ViLT pretraining objectives

- **Word Region/Patch Alignment** (WPA): Align image regions/patches with text tokens
  - Align specific words to specific image patches

For example: model needs to learn:

Dog -- dog patches
grass -- grass image patches

The loss is contrastive

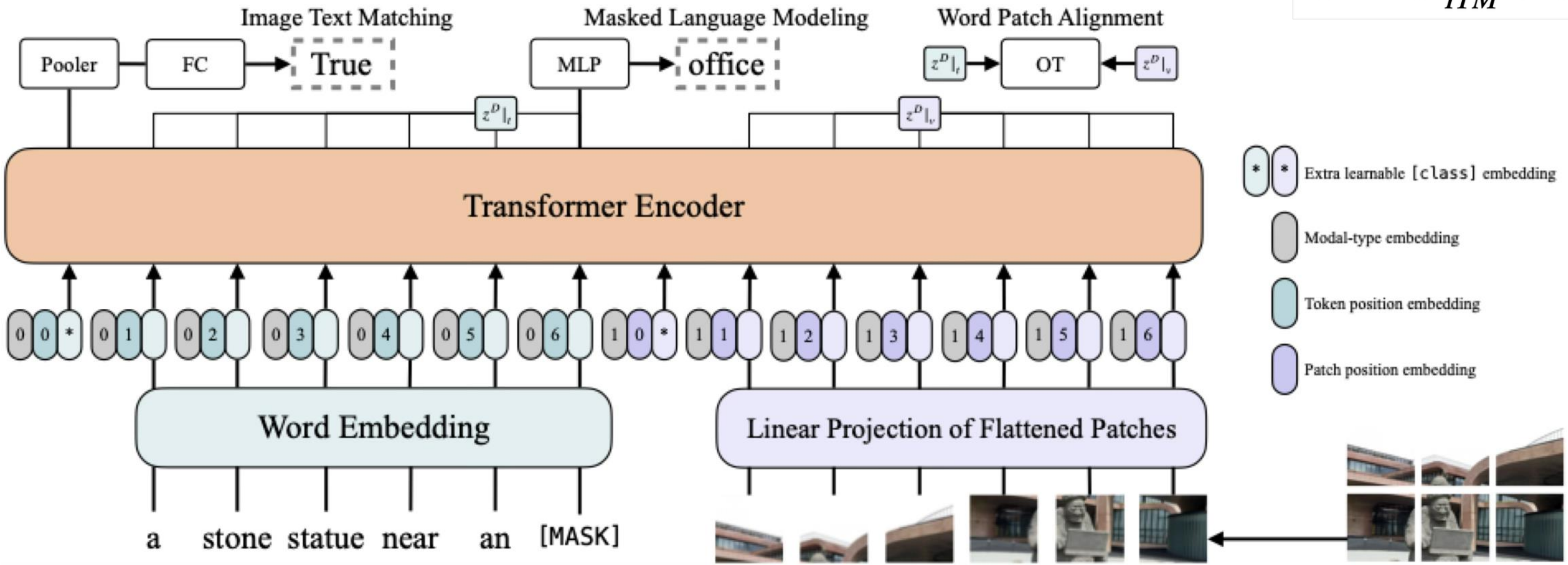
# ViLT: The Vision-Language Transformer pretraining objectives:

**ITM**
- Classify 0/1 if image and text are matching
- Negative pairs are sampled randomly every batch

**MLM**
- Predict the masked text tokens
- Without masking the images

**WPA**
- Align image patches and word tokens together.

$$\mathscr{L} = \mathscr{L}_{ITM} + \mathscr{L}_{MLM} + \mathscr{L}_{WPA}$$

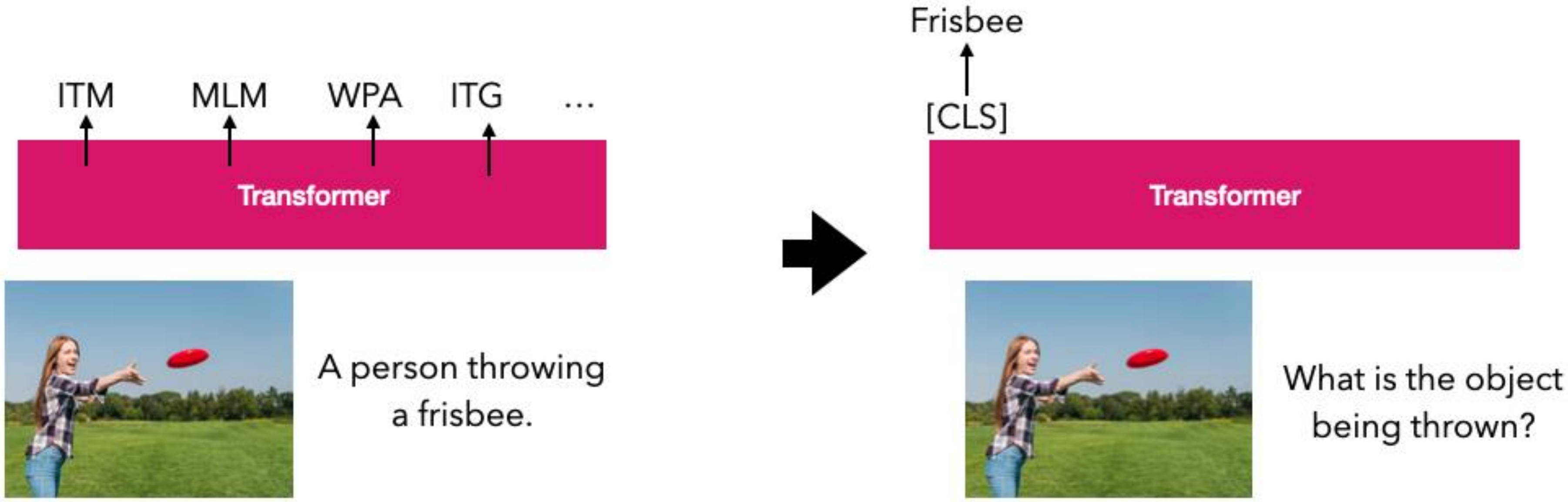


# Potential ViLT pretraining objectives

- Other pretraining objectives:
  - **Image to Text Generation** (ITG): Generate the next text tokens.
  - **Masked Image Modeling** (MIM): Predict/Regress masked image patches
  - **Region Prediction**: Predict object labels of provided regions.
  - Many more….

# Pre-Training to Downstream Tasks

- Similar to BERT, finetune **[CLS] token for classification tasks**.

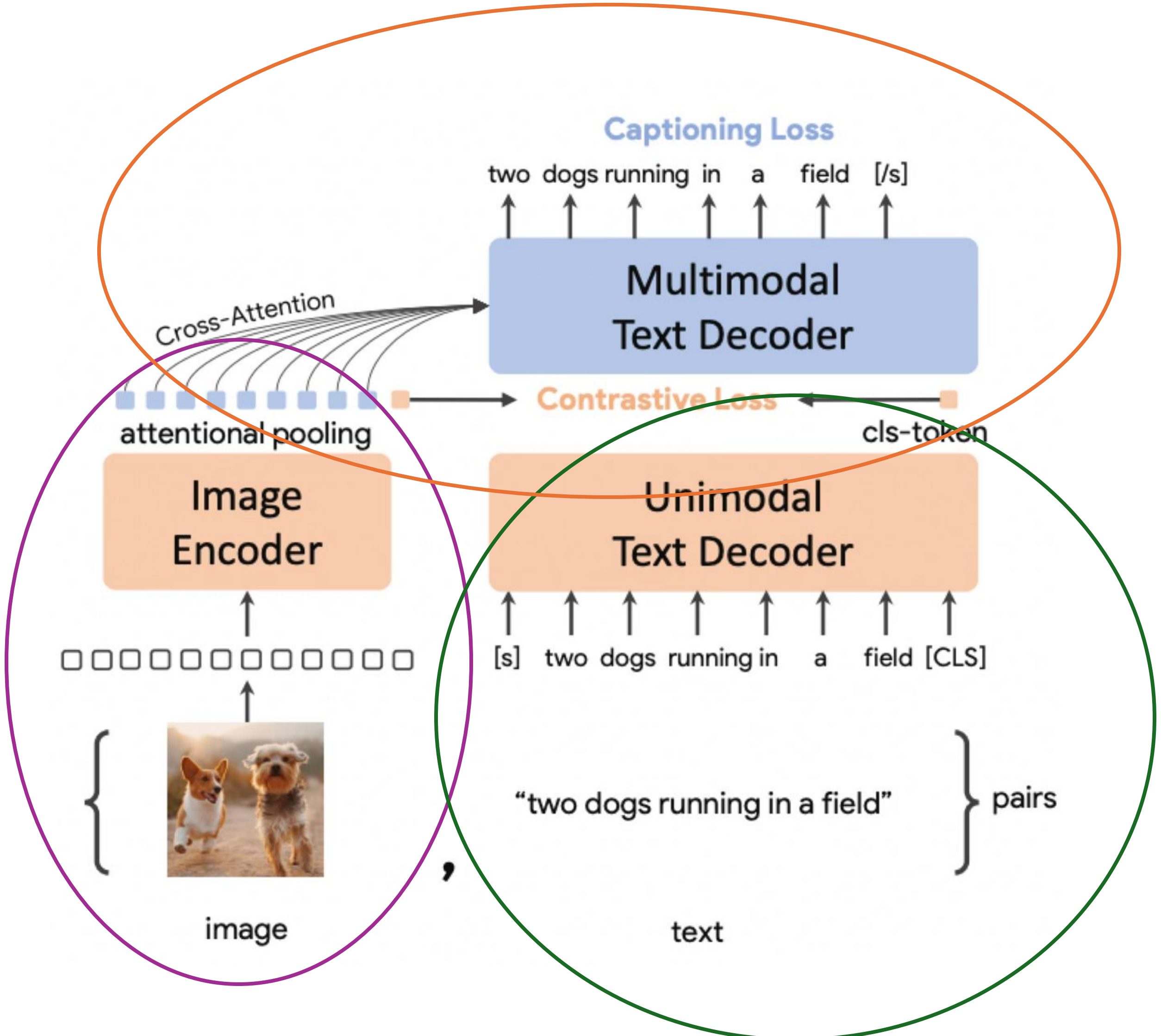

ITM
MLM
WPA
ITG
...
Transformer
A person throwing a frisbee.
Frisbee
[CLS]
Transformer
What is the object being thrown?

**Pre-Training to Learn Image-Text Alignment**

**Finetune to Downstream Tasks (e.g. VQA)**

# An example downstream task: image captioning

- CoCA: Contrastive Captioning



# Text-to-Image Generation

# Language + vision

### Image & Text Alignment

↕

A person throwing a frisbee.

### Image + Text Understanding

What is the object being thrown?

↓

A frisbee

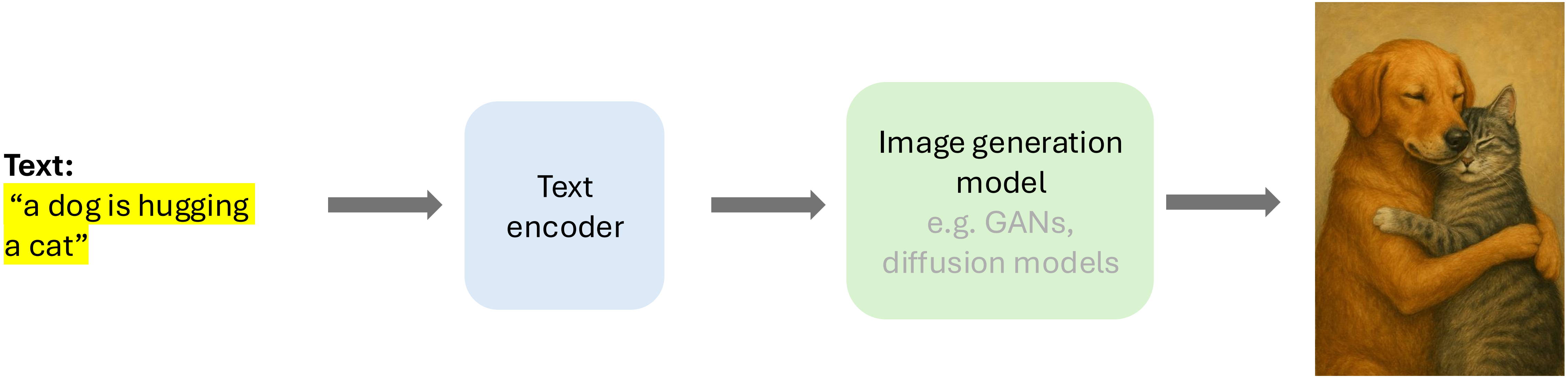
### Text to Image Generation

A person throwing a frisbee.

↓

# T2I models

- There are many different architectures
- One example:



**Text:**
“a dog is hugging a cat”

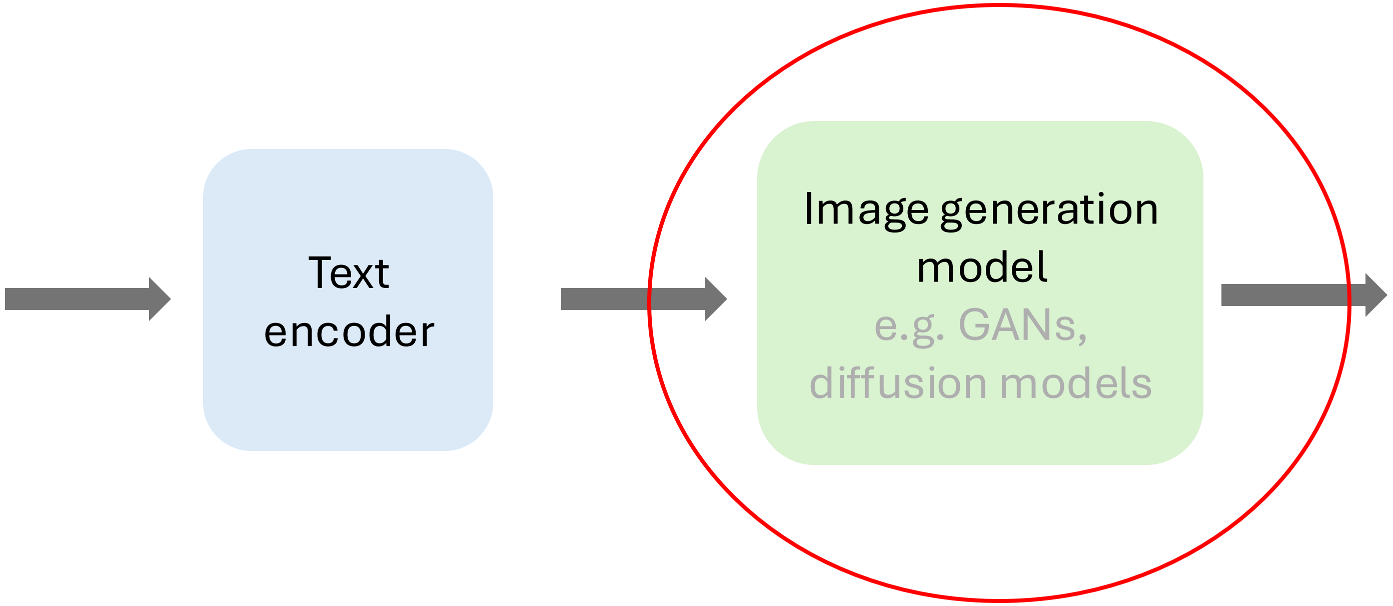
Text encoder

Image generation model
e.g. GANs, diffusion models

# T2I objectives

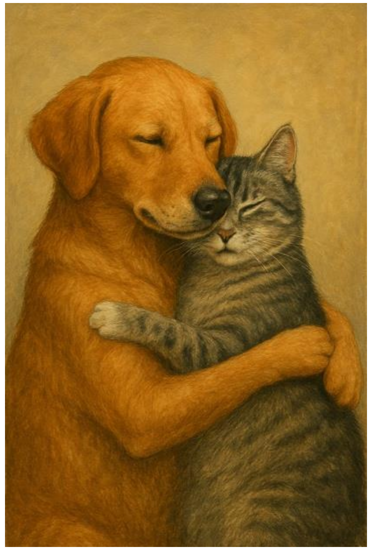
- Also train on text and image pairs, generally scraped from the internet
- The interesting part here is the **image generation** part
  - Many options: GANs, diffusion models…

**Text:**
"a dog is hugging a cat"



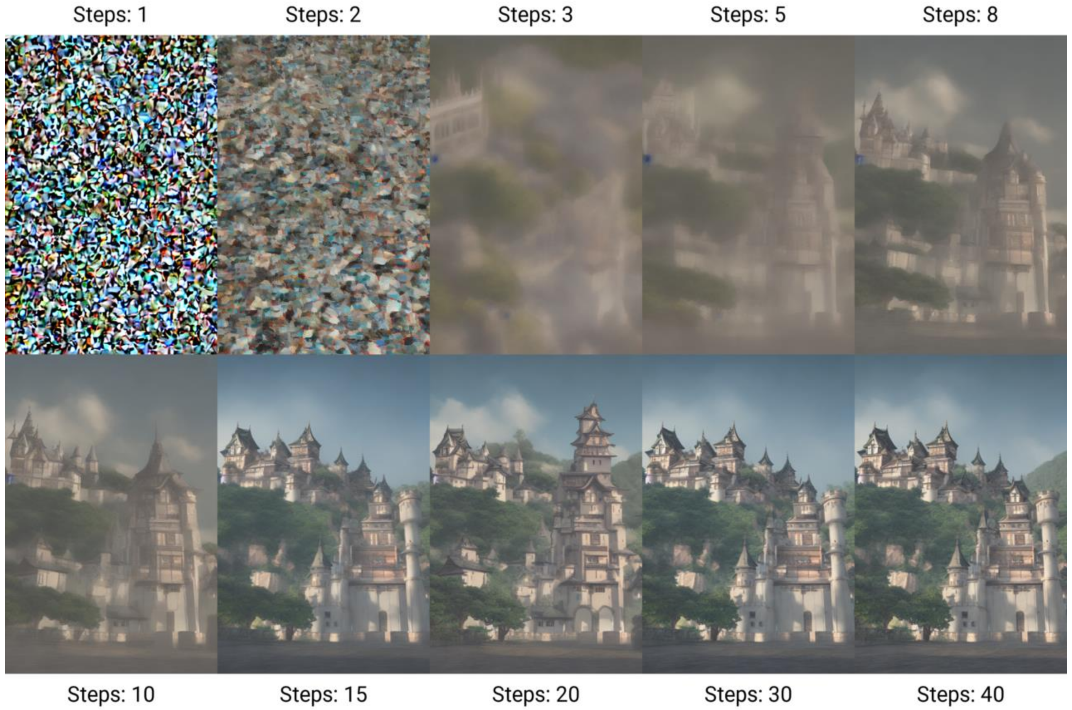
Text encoder

Image generation model
e.g. GANs, diffusion models



# Briefly how diffusion model works

- Start from random noise.
- Gradually denoise using a neural network conditioned on text.
- Trained by learning to predict and remove noise.



# Language and vision

- If you are interested in diving deeper into the vision aspect, it’s highly recommended that you take a class on graduate-level Computer Vision.
  - CNN, facial recognition, camera calibration, SIFT, object detection, and many more interesting topics! A lot of linear algebra. Fun!

- You could request us to talk more about T2I generation for the final lecture next Tuesday!
  - See the Ed post!