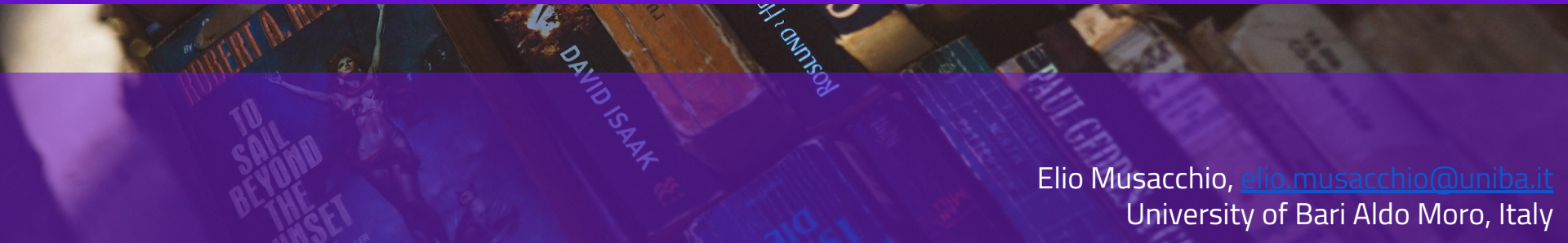
UNIVERSITÀ
DEGLI STUDI DI BARI
ALDO MORO

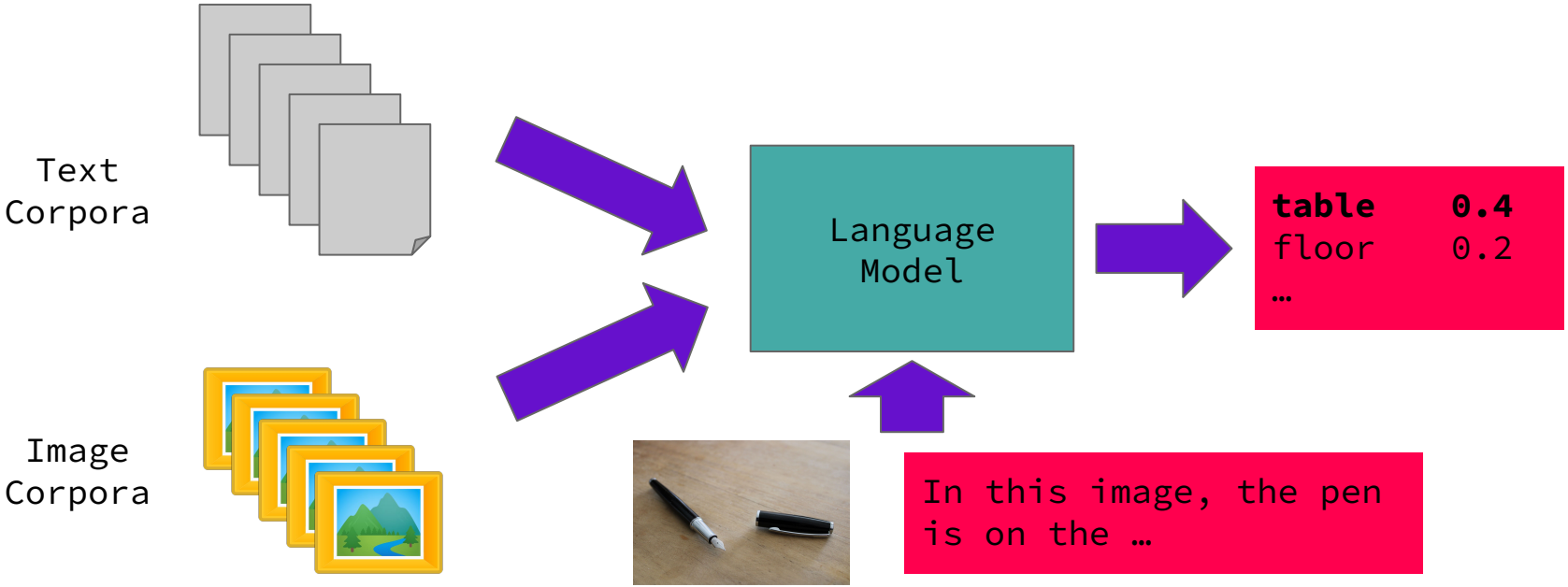
Multimodal NLP and Large Vision-Language Models (LVLMs)



Elio Musacchio, elio.musacchio@uniba.it
University of Bari Aldo Moro, Italy

Vision-Language Model

Extends a Language Model, **assigning probabilities to sequences** of words and images, **based on the analysis of multimodal corpora**.



Large Vision-Language Model

Large vision-language model:

- trained on large **amounts** of image-text data
- usually extends a Large Language Model trained on text

Large Vision-Language Model

Large vision-language model:

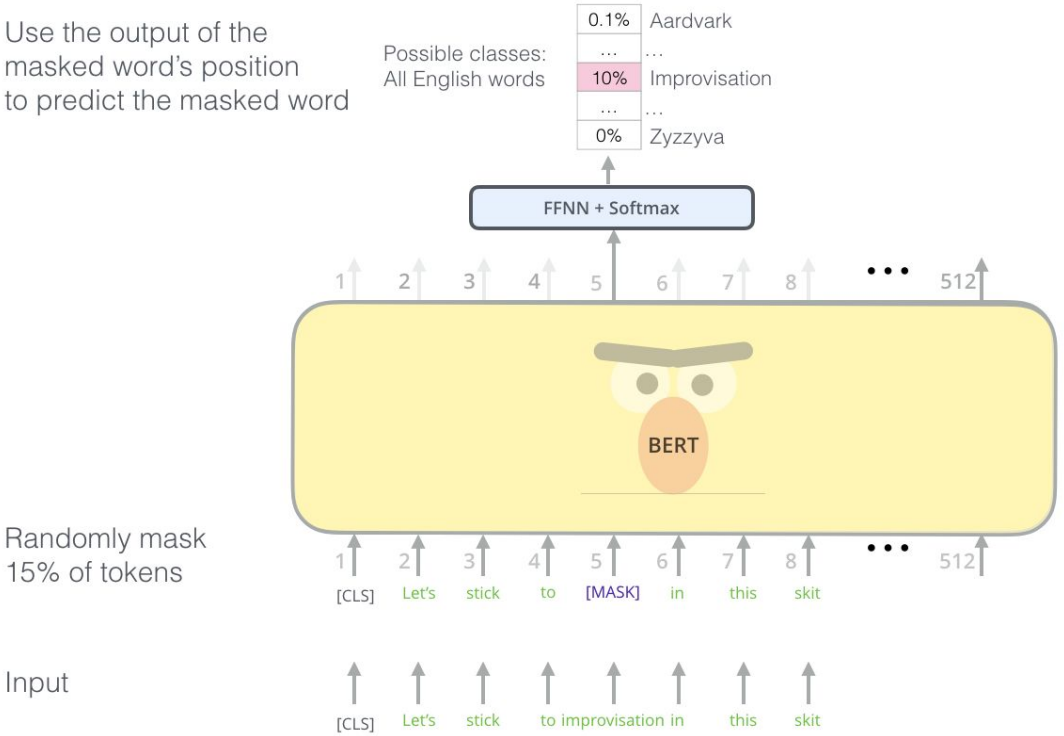
- trained on large **amounts** of image-text data
- usually extends a Large Language Model trained on text

But how did we get there?

Multimodal NLP

Bert - 2018

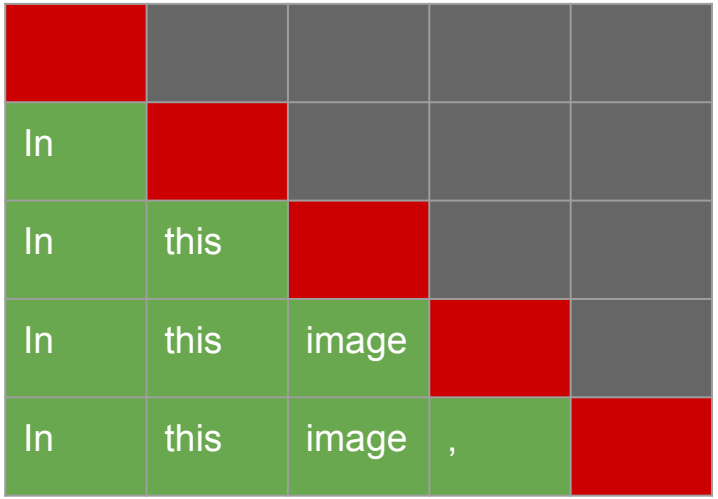
Masked Language Modeling pre-training



Multimodal NLP
Bert - 2018

Causal Language Modeling

T -> Future Tokens (unseen)
T -> Token to Predict (unseen)
T -> Past Tokens (seen)

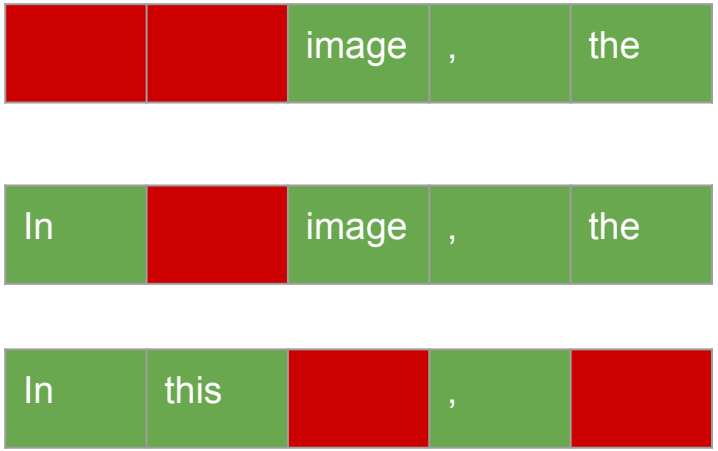


Multimodal NLP
Bert - 2018

Masked Language Modeling

T -> Token to Predict (unseen)
T -> Past and Future Tokens (seen)

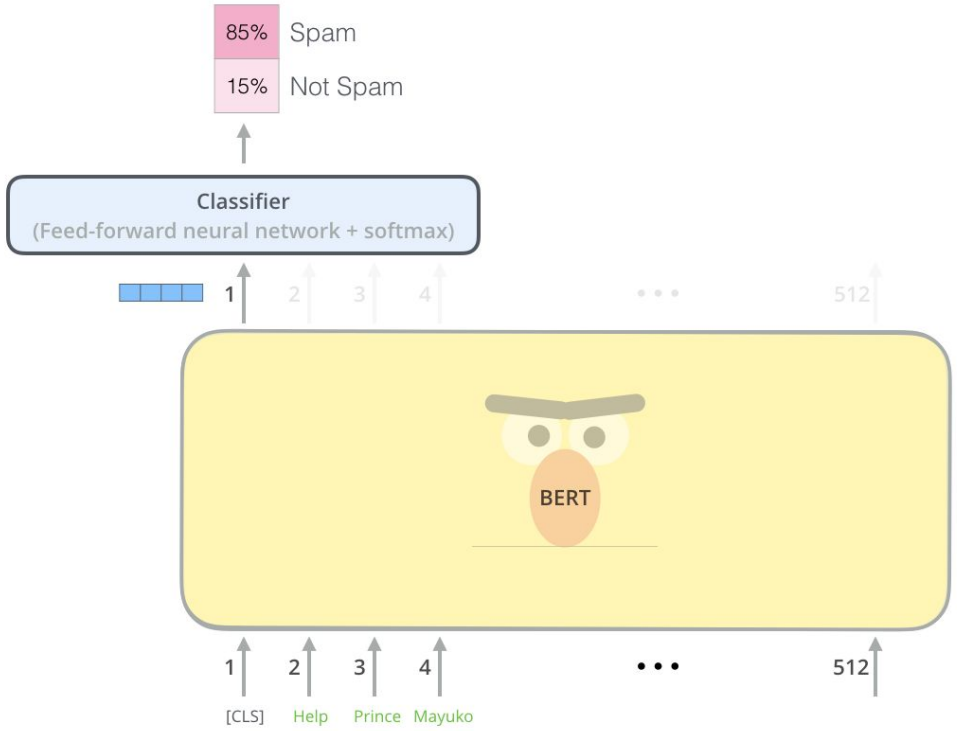
Training loss and setup is the same as LLMs, the only thing that changes is how the input-output is structured w.r.t. the objective



Multimodal NLP

Bert - 2018

Classification fine-tuning



Multimodal NLP
Pooling

- **Fun Fact**
  - While Bert uses the [CLS] token embedding for classification, there are several ways to extract embeddings from Bert
  - One could also perform mean pooling (average all token embeddings representations)
  - Several pooling strategies are used in literature (CLS, mean, last, …)
  - Best pooling strategy is usually determined by performing experiments with different pooling mechanisms

Multimodal NLP

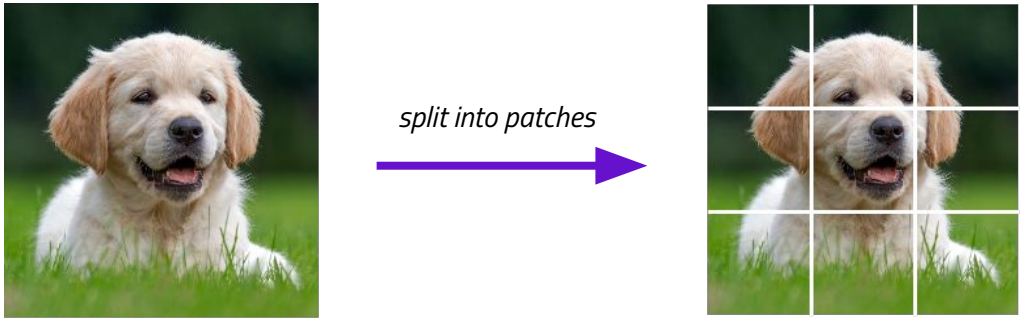
Vision Transformer - 2020

- **How can we use the Transformer architecture on visual inputs?**

Multimodal NLP

Vision Transformer - 2020

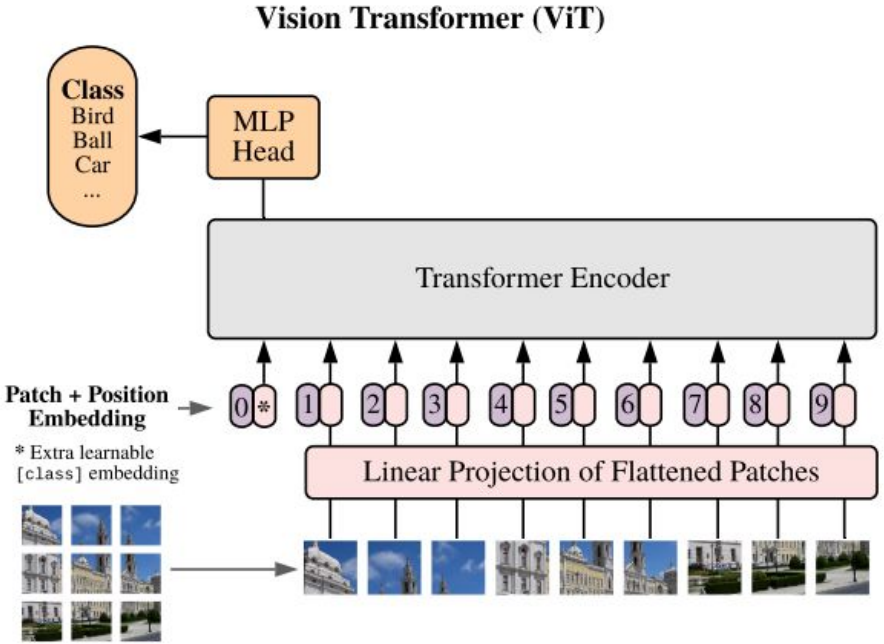
- **How can we use the Transformer architecture on visual inputs?**



- **Treat images as sequences of tokens**

Multimodal NLP

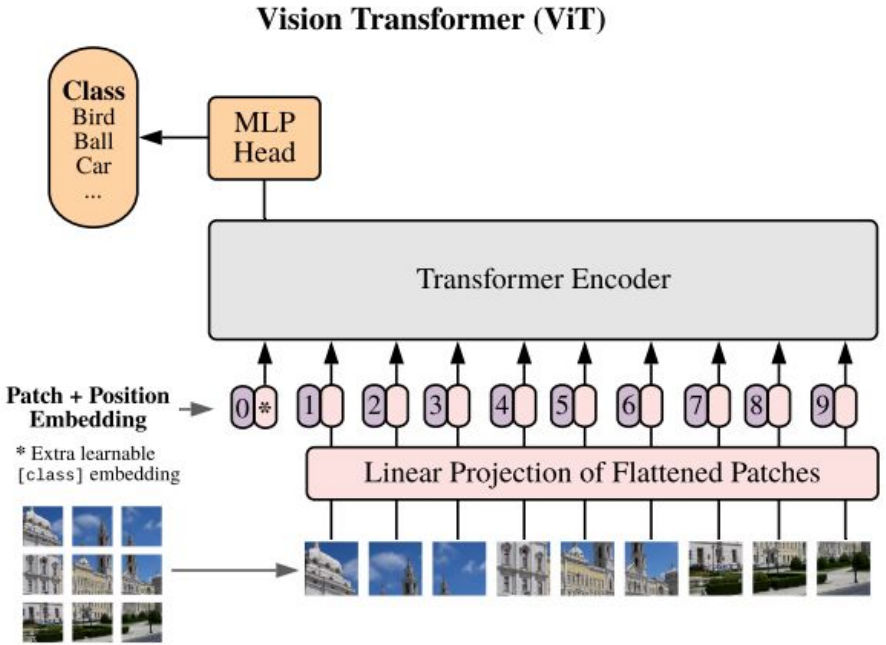
Vision Transformer - 2020



Multimodal NLP

Vision Transformer - 2020

- **Split image into fixed-size patches**
- **Treat these patches as if they were textual tokens**
- **Append special CLS embedding**
- **Input to Transformer encoder**



Multimodal NLP

Vision Transformer - 2020

- **Main differences w.r.t. Bert**
  - 1: There is no fixed token-to-embedding matrix, a projection layer is used to obtain the embeddings for the patches
  - 2: Fixed positional encoding learned at train time, since we know that there will always be a fixed number of patches
  - 3: No masked modeling pre-training step, fine-tuning directly

Multimodal NLP

Vision Transformer - 2020 (Bonus Cool Thing)

- Since ViT is based on a Transformer encoder, differently from CNNs like ResNet, it leverages the Self Attention mechanism
- This means that it is possible to see which patches of the images where more relevant for the prediction of the image class

Multimodal NLP

Vision Transformer - 2020 (Bonus Cool Thing)

- **Attention Rollout**
  - 1: Start with the attention matrix from the first layer
  - 2: For each subsequent layer, multiply the current layer's attention matrix with the previous layer's attention rollout matrix
  - 3: Normalize the rows of the final attention rollout matrix to ensure the total attention flow sums to 1

- This recursive computation allows the method to capture how information flows through the self-attention layers of the Transformer, rather than focusing on the last layer only

https://www.kaggle.com/code/arnavs19/attention-rollout-for-vision-transformers

Multimodal NLP

Vision Transformer – 2020 (Bonus Cool Thing)

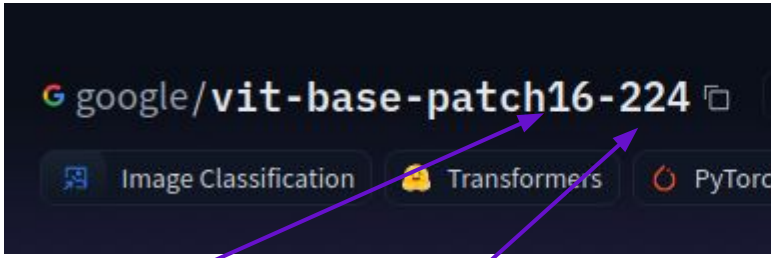
- **PROBLEM!**
  - How to account for residual connections?
  - Specifically, Attention is added to the input following this formula:
  - $V_{l+1} = V_l + W_{att} V_l$
  - Hence, we compute the attention matrix using the following formula:
  - $A = 0.5 W_{att} + 0.5 I$

https://www.kaggle.com/code/arnavs19/attention-rollout-for-vision-transformers

Multimodal NLP

Vision Transformer - 2020

- **Patch size and image size depends on the parameters the model was trained with**



Patch Size **Image Size**

(224/16) * (224/16) + 1 = 197

Multimodal NLP
Vision Transformer - 2020

- **Ok but what about images of different size?**

Multimodal NLP
Vision Transformer - 2020

- **Ok but what about images of different size?**
  - Option 1: Apply Processor transformations to resize the image into the expected size of the model
  - Option 2: if image size is important (e.g. resolution-sensitive images where information may be lost after resize) use **interpolation of positional encodings**

Multimodal NLP

Vision Transformer - 2020

- **Interpolation of Positional Encodings**
  - Step 1: Reshape learned Positional Encodings into a (num_patches**0.5) grid (preserve positional information learned by the encodings)
  - Step 2: Interpolate w.r.t. the new width and height
  - Step 3: Flatten this reshaped matrix

Multimodal NLP

CLIP - 2021

- **What if we wanted to compare the embedding representation learned by ViT and by BERT?**

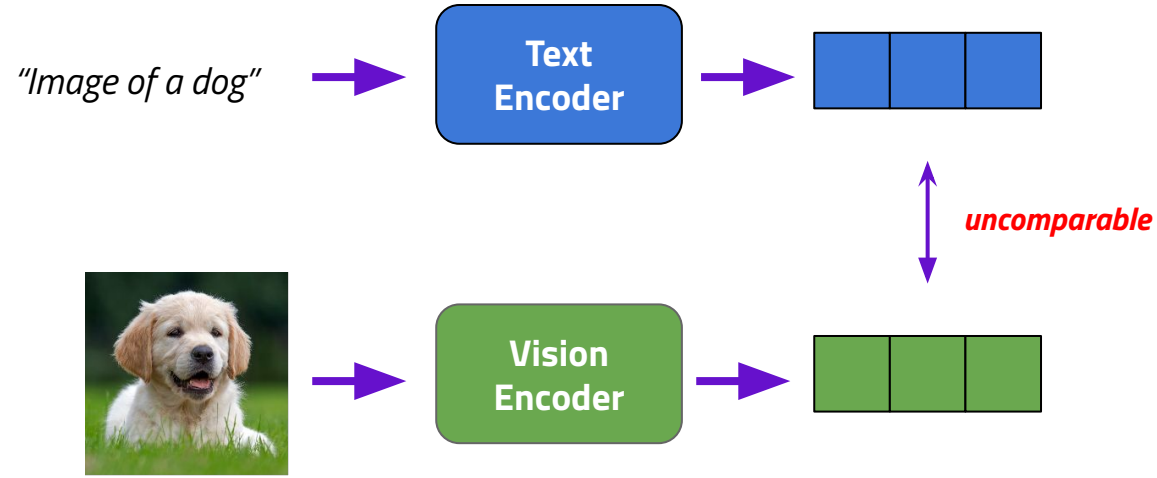
Multimodal NLP
CLIP - 2021

- **What if we wanted to compare the embedding representation learned by ViT and by BERT?**
  - This is impossible since the models learn a space that is conceptually different
  - Additionally, the hidden dimension of the two models may be different (e.g. 512 vs 768). This would make comparison impossible from a computational point of view

Multimodal NLP

CLIP - 2021

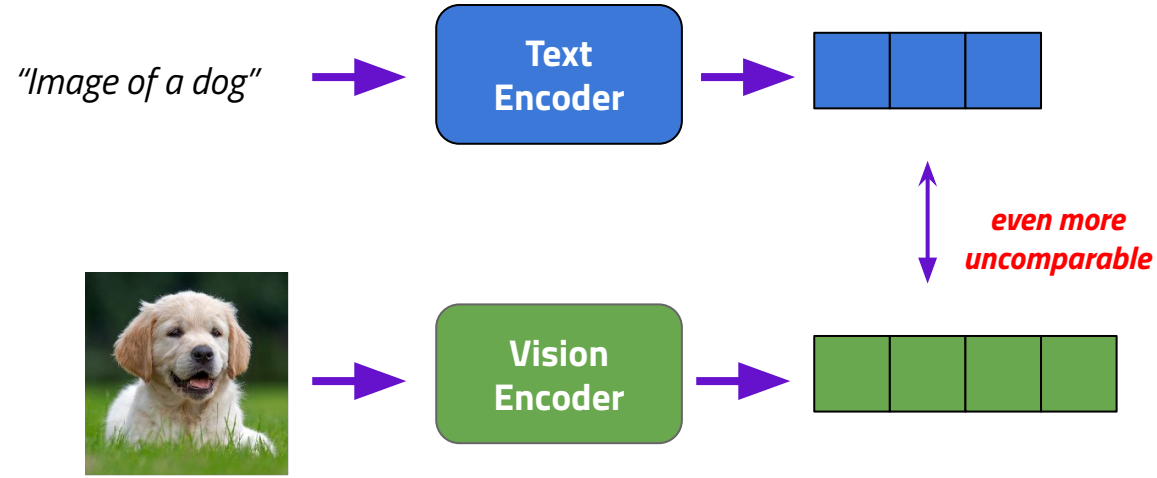
- **Contrastive-Learning in Pre-Training (CLIP)**



Multimodal NLP

CLIP - 2021

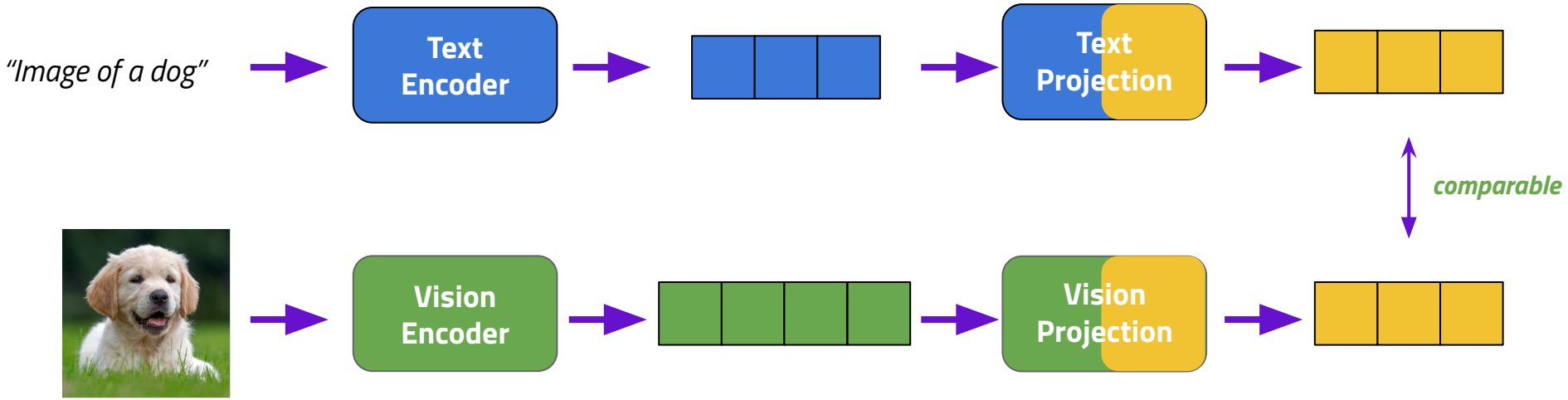
- **Contrastive-Learning in Pre-Training (CLIP)**



Multimodal NLP

CLIP - 2021

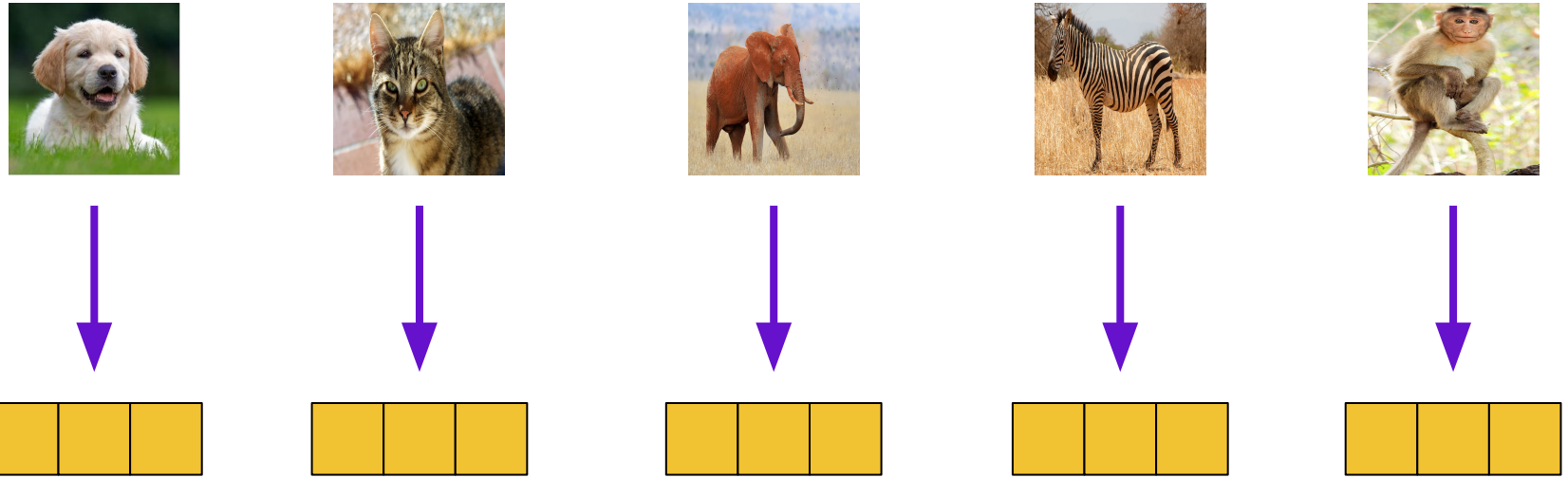
- **Contrastive-Learning in Pre-Training (CLIP)**



Multimodal NLP

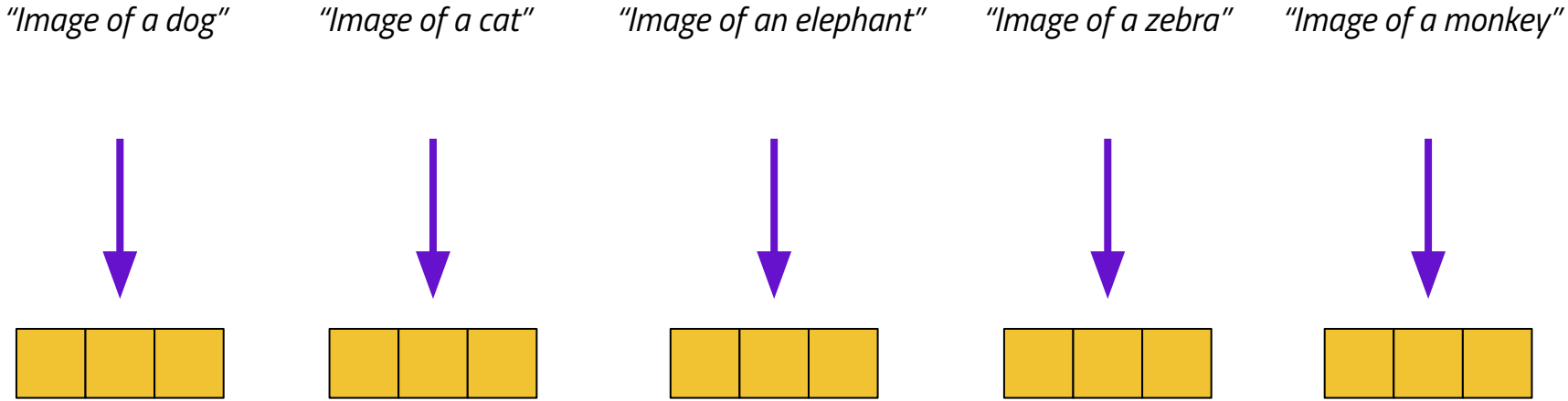
CLIP - 2021

- **Contrastive Loss**



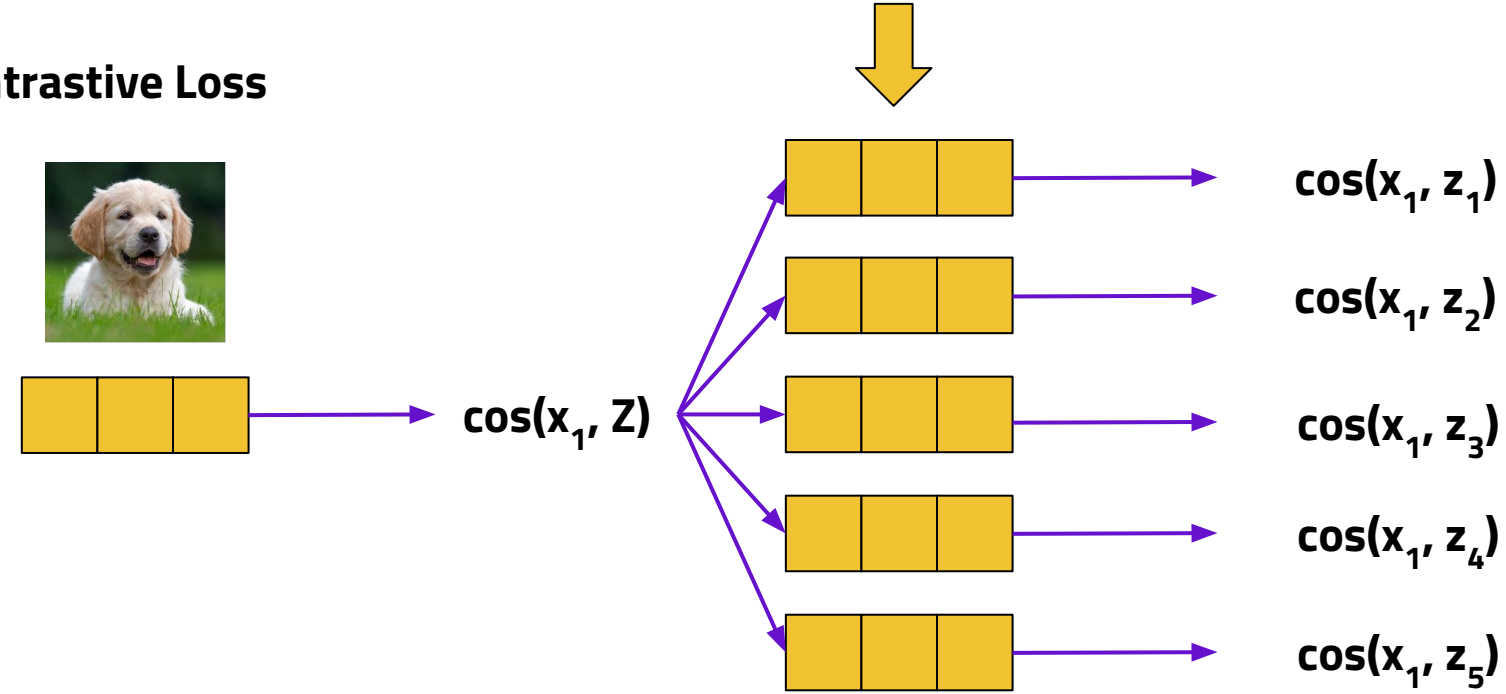
Multimodal NLP
CLIP - 2021

- **Contrastive Loss**



Multimodal NLP
CLIP - 2021

- **Contrastive Loss**



Multimodal NLP
CLIP - 2021

- **Contrastive Loss**
- **Ground Truth**





"Image of a dog"	"Image of a cat"	"Image of an elephant"	"Image of a zebra"	"Image of a monkey"
1	0	0	0	0
0	1	0	0	0
0	0	1	0	0
0	0	0	1	0
0	0	0	0	1

Multimodal NLP
CLIP - 2021

- **Contrastive Loss**
- **Predictions**
- **Loss is computed column and row wise. The sim scores will become probabilities thanks to the softmax function**





"Image of a dog"	"Image of a cat"	"Image of an elephant"	"Image of a zebra"	"Image of a monkey"
0.9	0.4	0.8	0.2	0.4
0.5	…	…	…	…
0.6	…	…	…	…
0.3	…	…	…	…
0.2	…	…	…	…

Multimodal NLP
CLIP - 2021

- **Contrastive Loss**
- **Predictions**
- **Loss is computed column and row wise. The sim scores will become probabilities thanks to the softmax function**





"Image of a dog"

0.2893	1
0.1939	0
0.2143	0
0.1588	0
0.1437	0

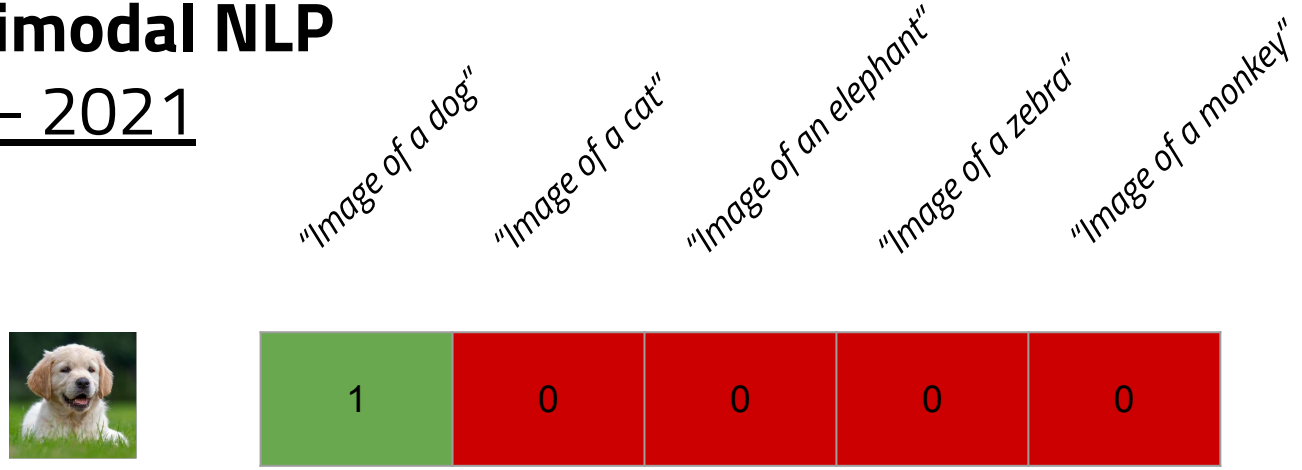
Multimodal NLP
CLIP - 2021

- **Contrastive Loss**
- **Predictions**
- **Loss is computed column and row wise. The sim scores will become probabilities thanks to the softmax function**

"Image of a dog"	"Image of a cat"	"Image of an elephant"	"Image of a zebra"	"Image of a monkey"
0.2767	0.1678	0.2503	0.1374	0.1678
1	0	0	0	0

Multimodal NLP

CLIP - 2021

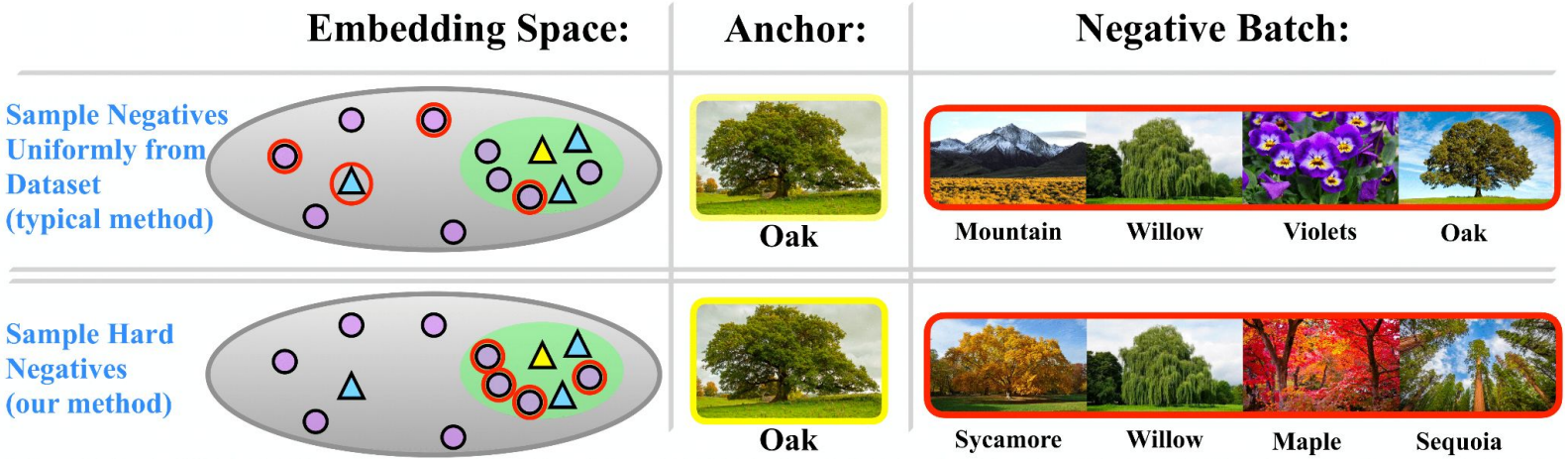


- **Fun Fact**

Batch size is an important hyperparameter when training a model using a contrastive loss. This is because the other labels in the training instance will act as negatives, the image of the dog will be distant in the embedding space w.r.t. the other four labels. **In general**, the greater the batch size, the better the results.

Multimodal NLP

CL with Hard Negative Samples - 2021



- **Hard Negative**

Negative samples that have different label from the anchor, but that are embedded nearby

Multimodal NLP

Gradient Cache - 2021

- **How to train a model with a bigger batch size than what the hardware is capable of? Gradient Accumulation**
  - Define a micro batch size, smaller than the overall batch size
  - Perform forward and backpropagation multiple times (once for each microbatch), the gradients are accumulated for each backward propagation
  - Update the model parameters once

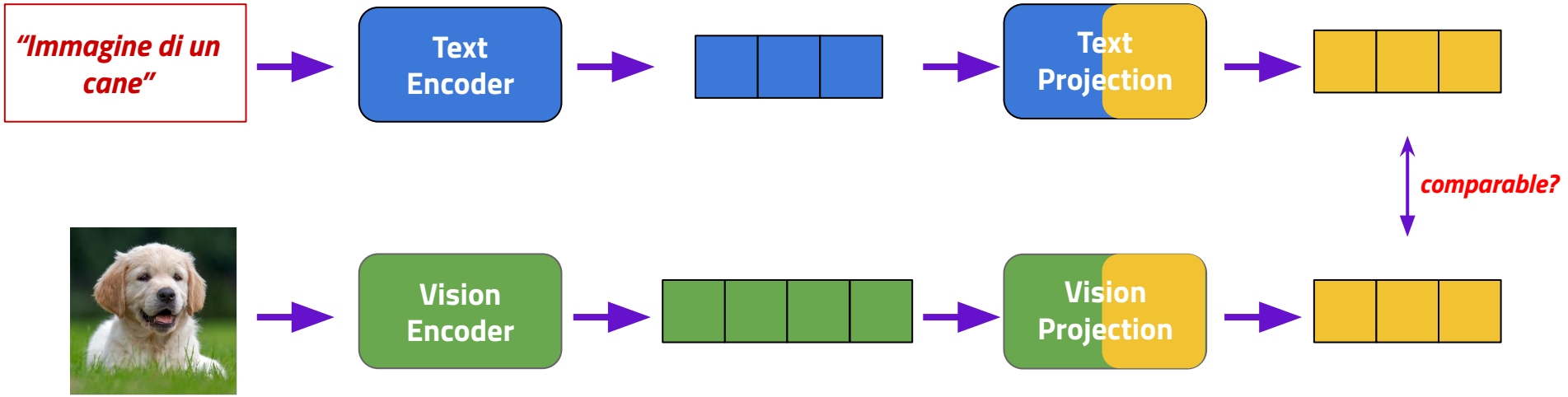
- **PROBLEM! This technique is sub-optimal for contrastive learning approaches. The micro batch size limits the amount of negatives in the construction of the similarity matrix since each loss is computed separately for each micro batch**

Multimodal NLP
Gradient Cache - 2021

- **Solution: Gradient Cache**
  - Step 1: compute the embeddings for the whole batch in inference mode (do not compute gradient), using a pre-defined micro batch size
  - Step 2: compute the contrastive loss using the extracted embeddings. calculate the gradient for the embeddings only (that is, the partial derivatives of the loss w.r.t. the embeddings)
  - Step 3: re-iterate through the micro batches of Step 1, perform the forward pass with gradient computation and perform backward propagation starting from the gradient computed in Step 2
- **Step 2 computes the loss by using the embeddings from the whole batch!**

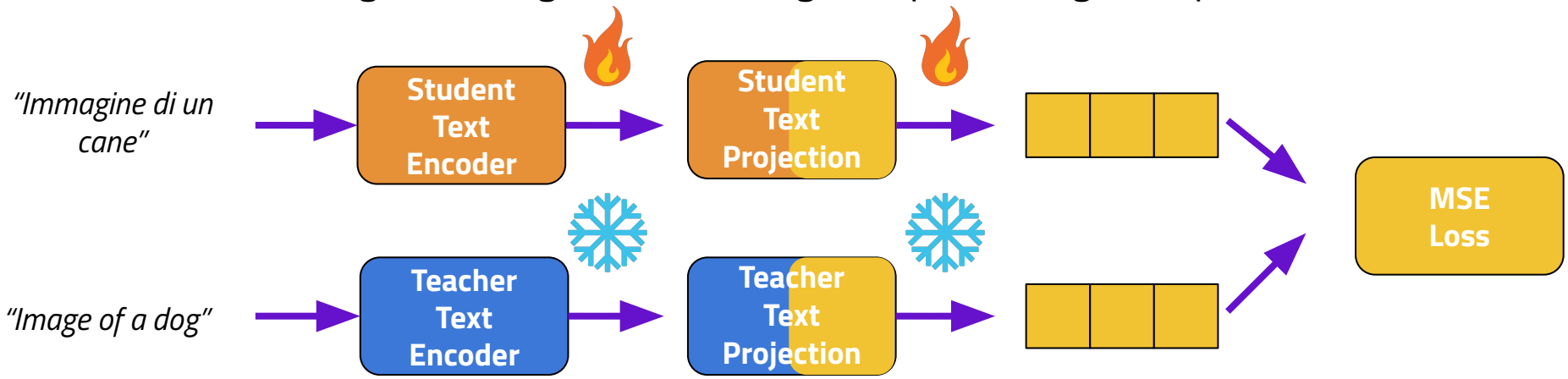
Multimodal NLP
mCLIP - 2023

- **Contrastive-Learning in Pre-Training (CLIP)**
  - What happens for texts not in English? Poor performance



Multimodal NLP
mCLIP - 2023

- **Contrastive-Learning in Pre-Training (CLIP)**
  - Align non-English embeddings to optimal English representation



Multimodal NLP
mCLIP - 2023

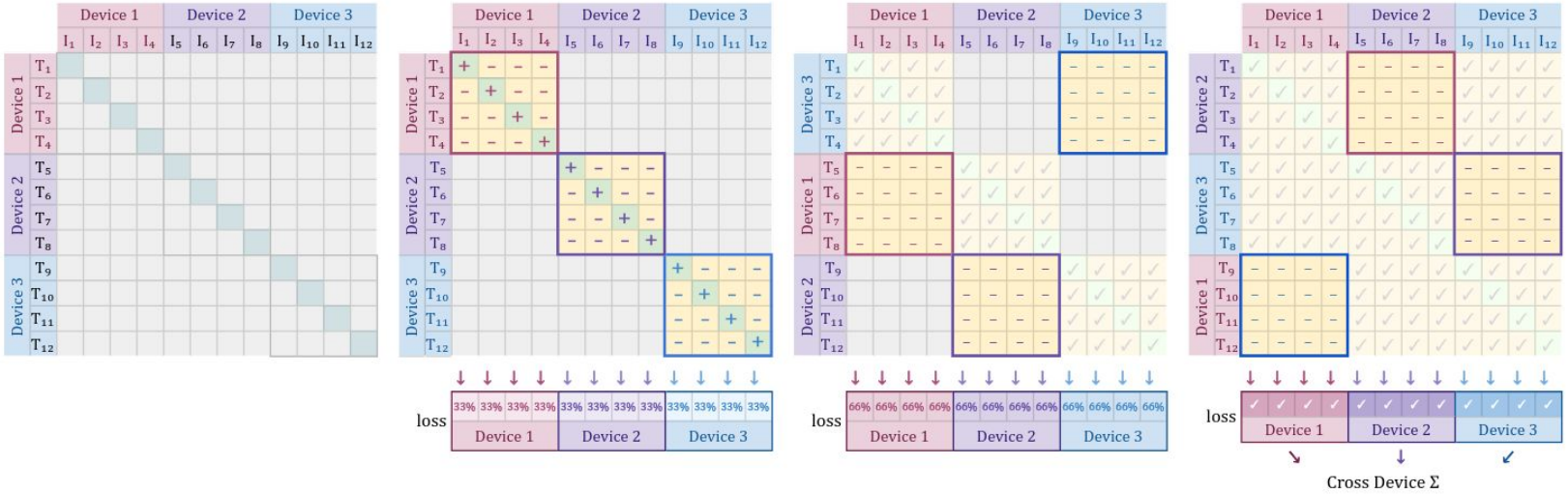
- **Can't I just train with multilingual inputs from the beginning?**
  - Yes, BUT
    - There is no guarantee that the embedding representation for inputs in different languages will be the same, since there is no constraint at train time for this
      - The model may learn two different optimal representations for the same inputs in English and Italian
    - Difficult to reproduce the same training setting as the original CLIP due to the hardware requirements

Multimodal NLP
SigLIP - 2023

- **Sigmoid loss for Language-Image Pre-training**
  - CLIP uses a Softmax function on the cosine similarity matrix
  - This requires access to all scores obtained for each image and text in the batch
  - Solution: **Log Sigmoid** loss, effectively turning the problem into a binary classification one for each class
  - Even better for parallel devices training, there is no need to gather all embeddings on a single device to build the similarity matrix, negatives are swapped across all devices

Multimodal NLP

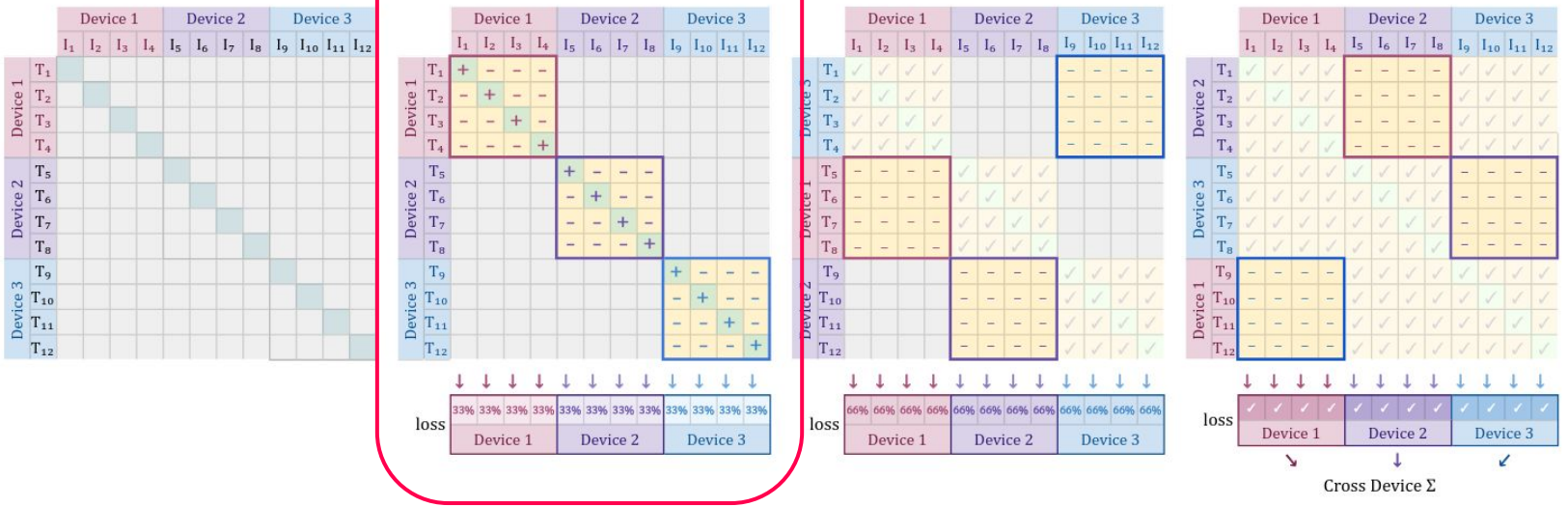
SigLIP - 2023



Multimodal NLP

SigLIP - 2023

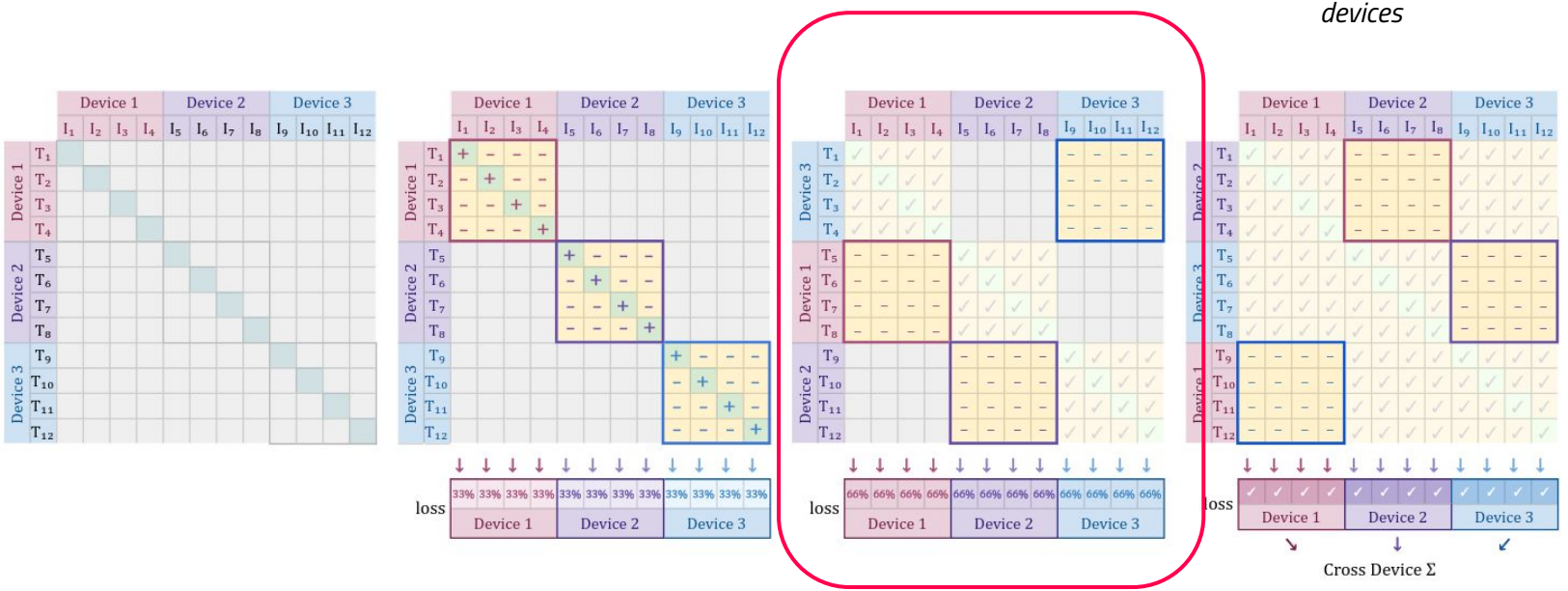
Compute loss for all items available on that device



Multimodal NLP

SigLIP - 2023

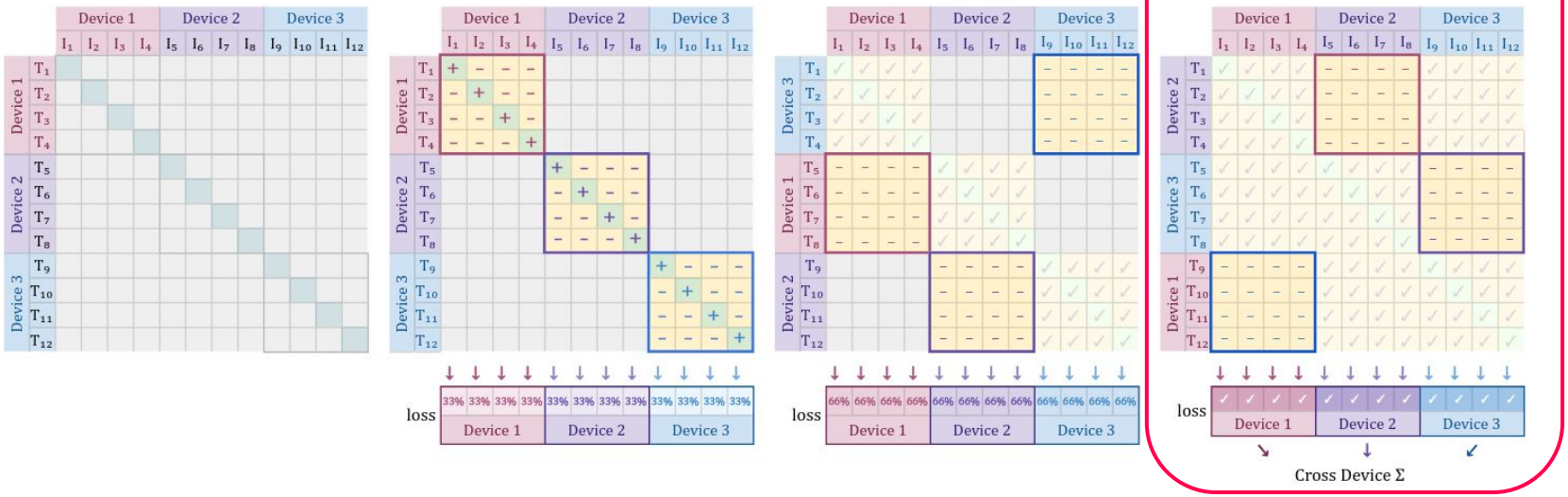
Swap negatives across devices



Multimodal NLP

SigLIP - 2023

All devices have seen all negatives, aggregate and compute final loss



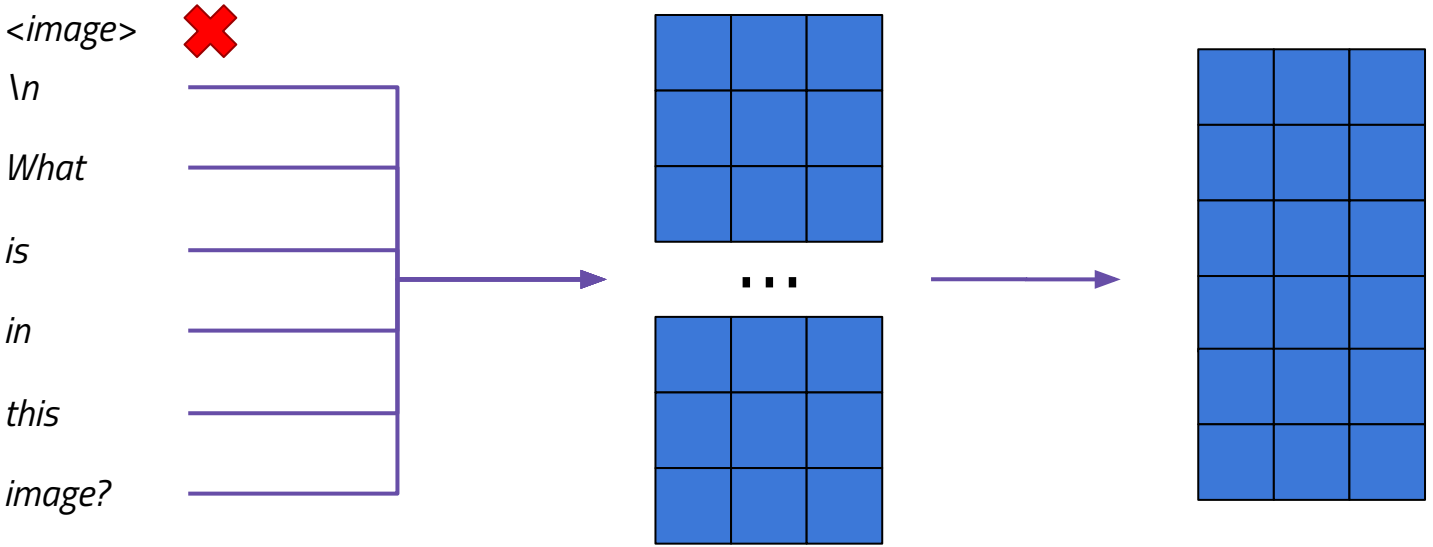
Multimodal NLP

Large Vision Language Models - 2024

- **There are four different approaches**
  - LLaVA (most used/famous)
  - BLIP-2
  - Flamingo
  - Chameleon

Multimodal NLP

LLaVA - 2024



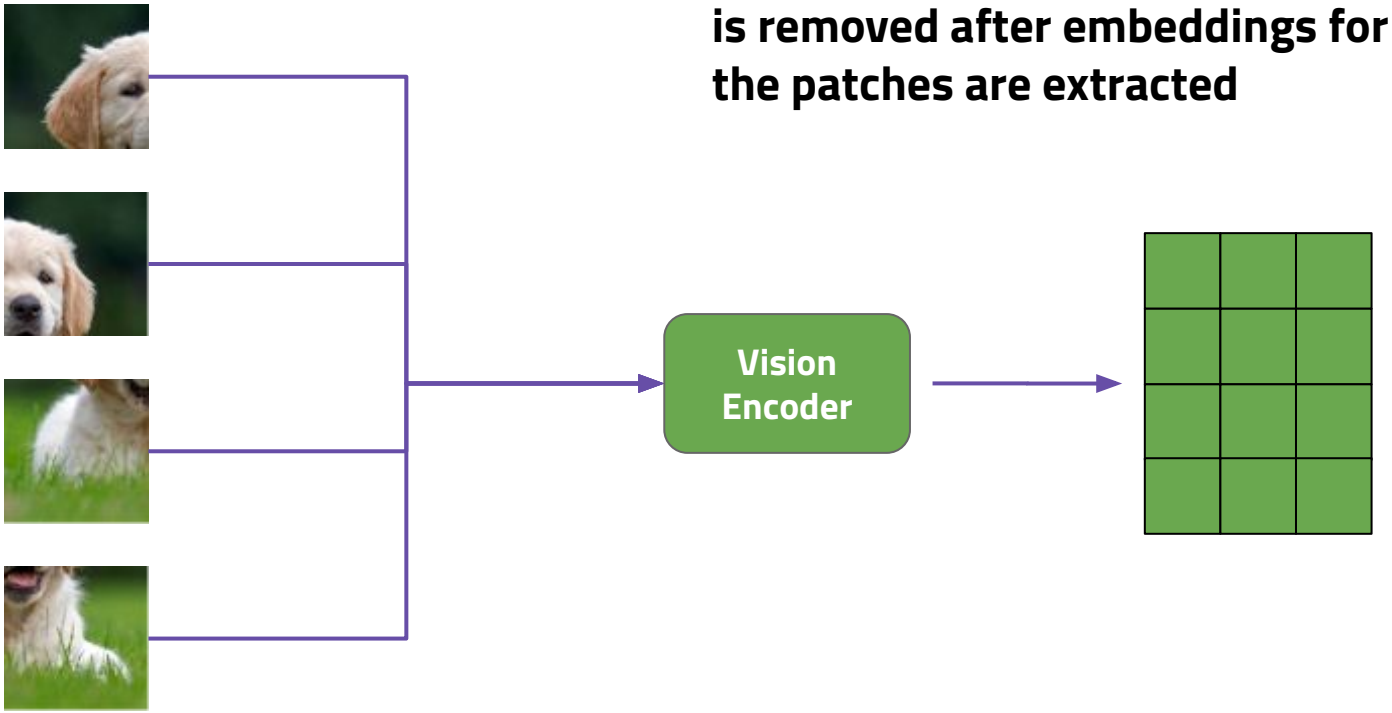
Natural Language Instruction

Token Embedding Matrix

Multimodal NLP

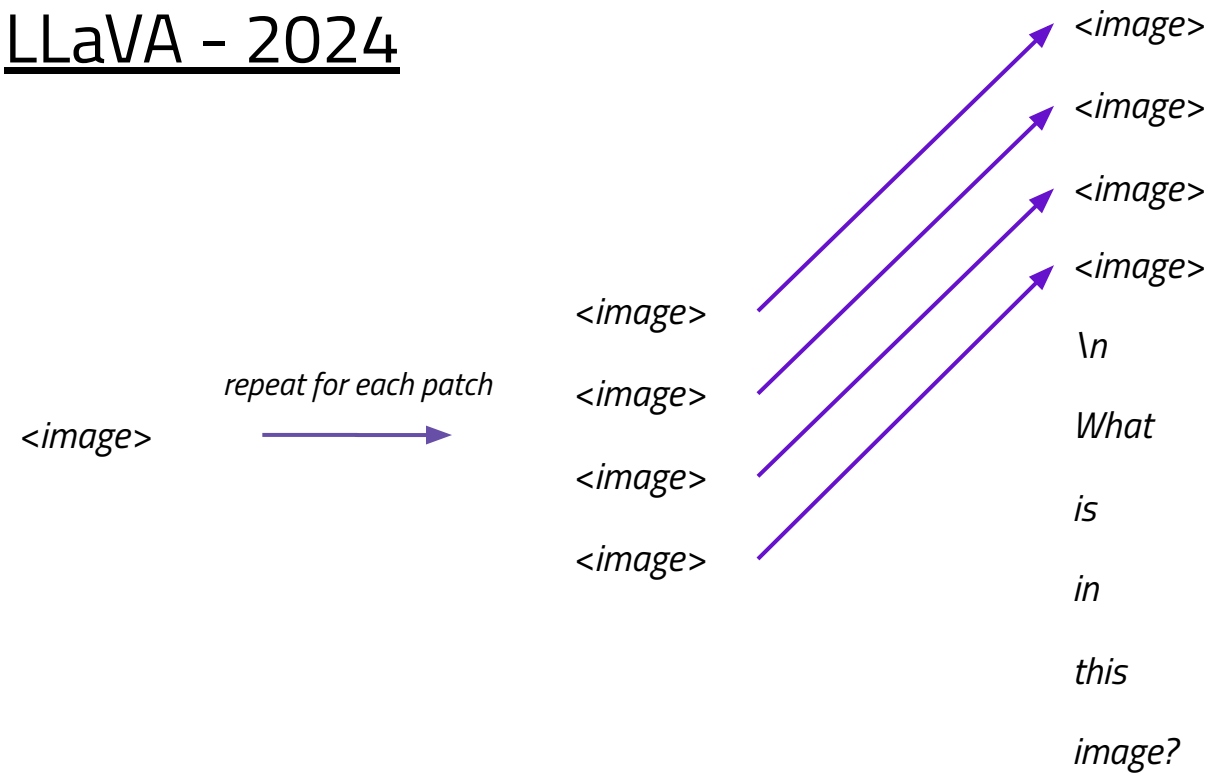
LLaVA - 2024

- **NOTE: the [CLS] token embedding is removed after embeddings for the patches are extracted**



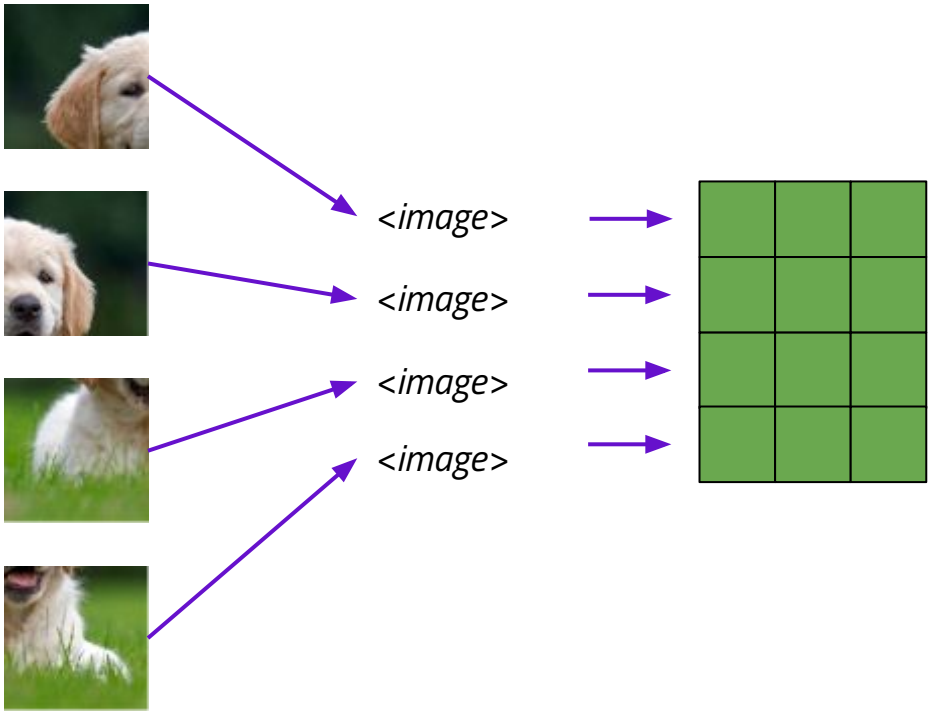
Multimodal NLP

LLaVA - 2024



Multimodal NLP

LLaVA - 2024



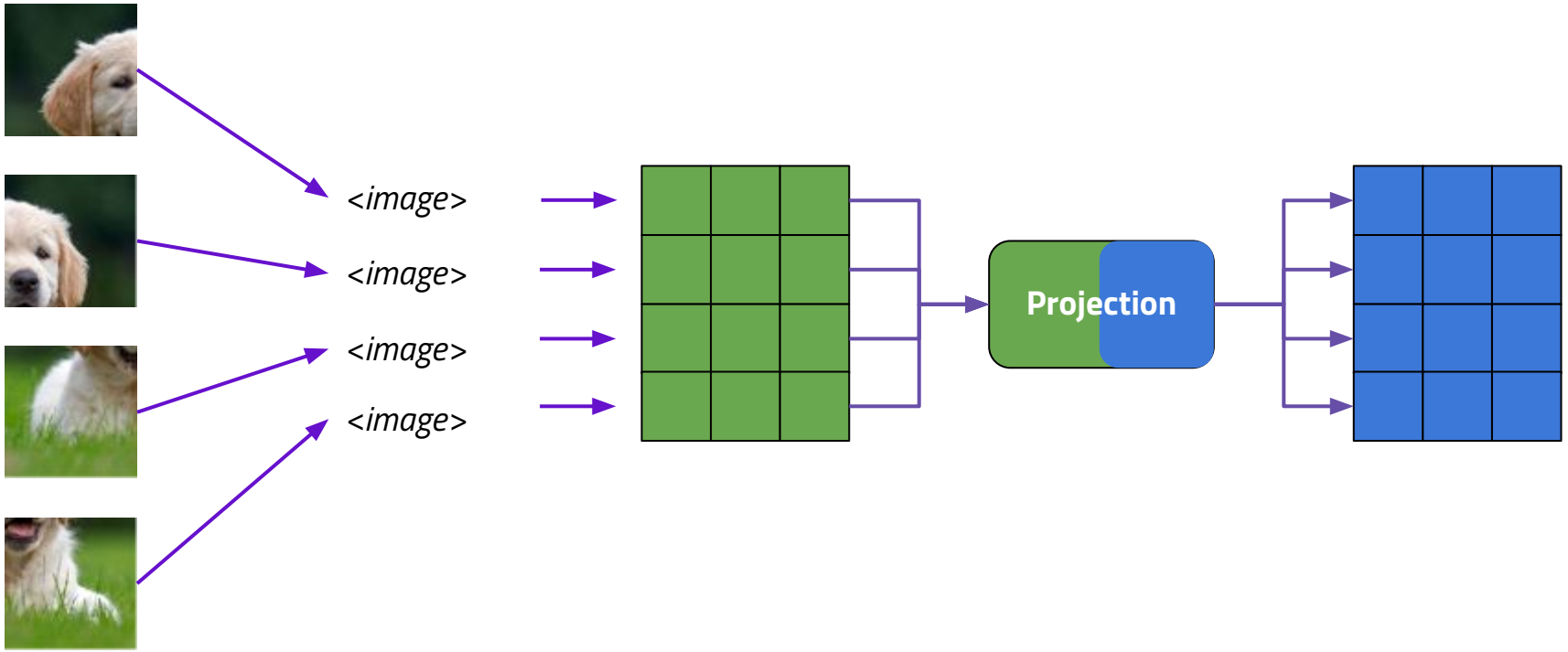
Problem !

Same issue as when we were trying to compare ViT and Bert embeddings

The LLM doesn't understand these embeddings

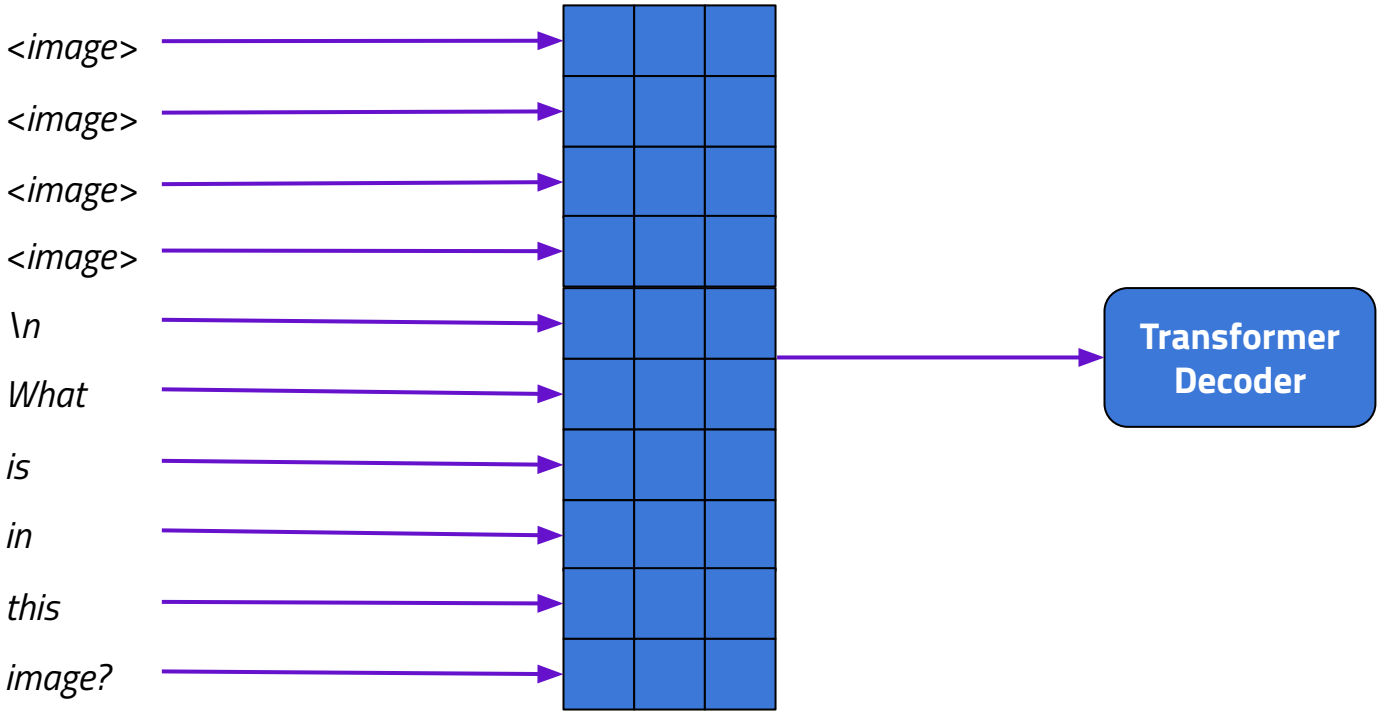
Multimodal NLP

LLaVA - 2024



Multimodal NLP

LLaVA - 2024



Multimodal NLP
LLaVA Training - 2024

- **LLaVA Training is split into two different steps**
  - Projector Pre-Training: freeze the weights of the vision encoder and of the LLM, train the projector only. A dataset of generic image-text pairs is often used in this step. The training objective is next-token prediction
  - Visual Instruction-Tuning: freeze the weights of the vision encoder only, train the projector and LLM. A dataset of visual instructions is used in this step. The training objective is next-token prediction

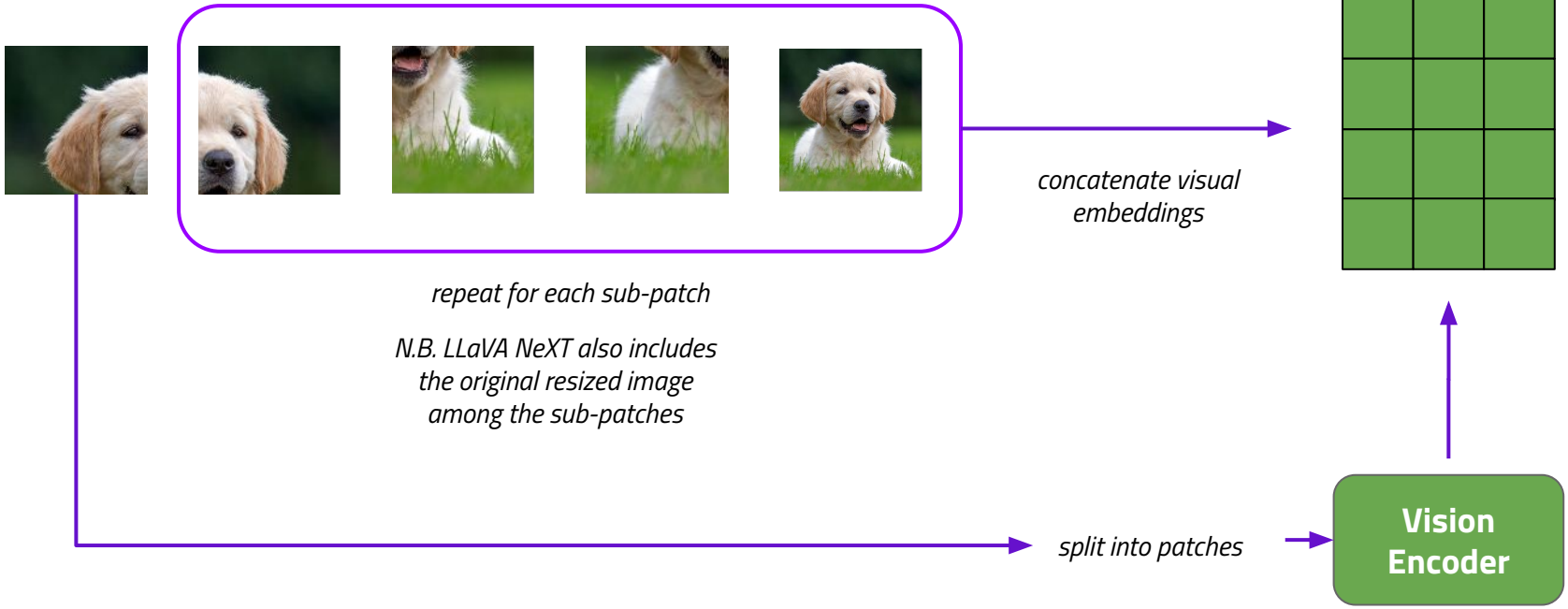
Multimodal NLP
LLaVA NeXT - 2024

- **To handle images with greater size than the visual encoder, split the image into greater sub-patches**

split into patches

Multimodal NLP

LLaVA NeXT - 2024



Multimodal NLP
Flamingo and BLIP-2

- **LLaVA and LLaVA NeXT have a very clear limitation:**
  - There is a fixed number of visual tokens that occupy the context length of the model
  - Considering a ViT-Clip-336 model as vision encoder, LLaVA has a total of 576 embeddings per image, while LLaVA NeXT has a total of 2880 per image

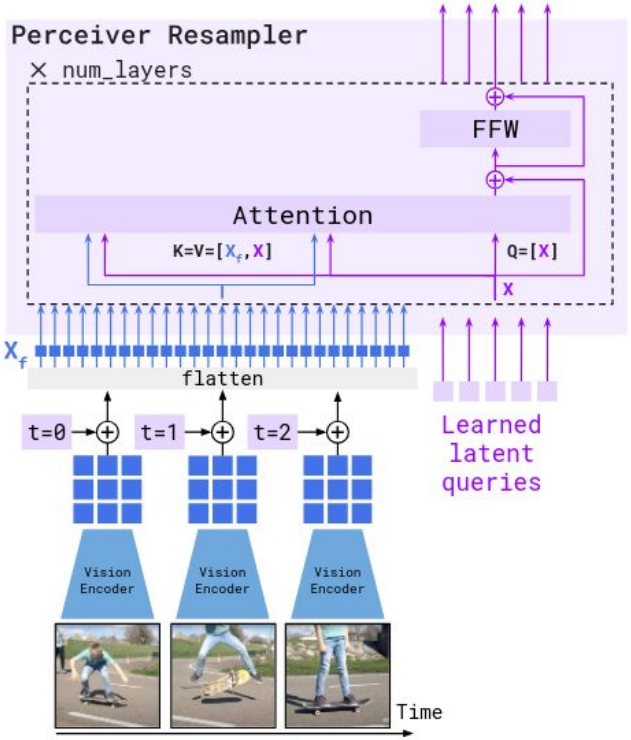
- **Are there other approaches that consider less visual embeddings at train and inference time?**
  - Perceiver Resampler (Flamingo)
  - Q-Former (BLIP-2)

Multimodal NLP
Flamingo - 2022

- **Perceiver Resampler:**
  - Extracts a meaningful visual summary of the provided input
  - **No pre-training**: it is trained directly with the LLM
    - In Flamingo's case, cross-attention trainable layers are added to each layer (the rest of the LLM is kept frozen)
- **LLM:**
  - **Cross-Attention**
  - **Tanh-gating Mechanism**

Multimodal NLP

Flamingo - 2022



Multimodal NLP
Flamingo - 2022

- **Perceiver Resampler in detail:**
  - Add positional encoding based on time step (for videos)
  - Concatenate visual embeddings (for videos)
  - **Use a cross-attention mechanism where the concatenation of the visual features and the latent queries is used as both Keys and Values**

Multimodal NLP

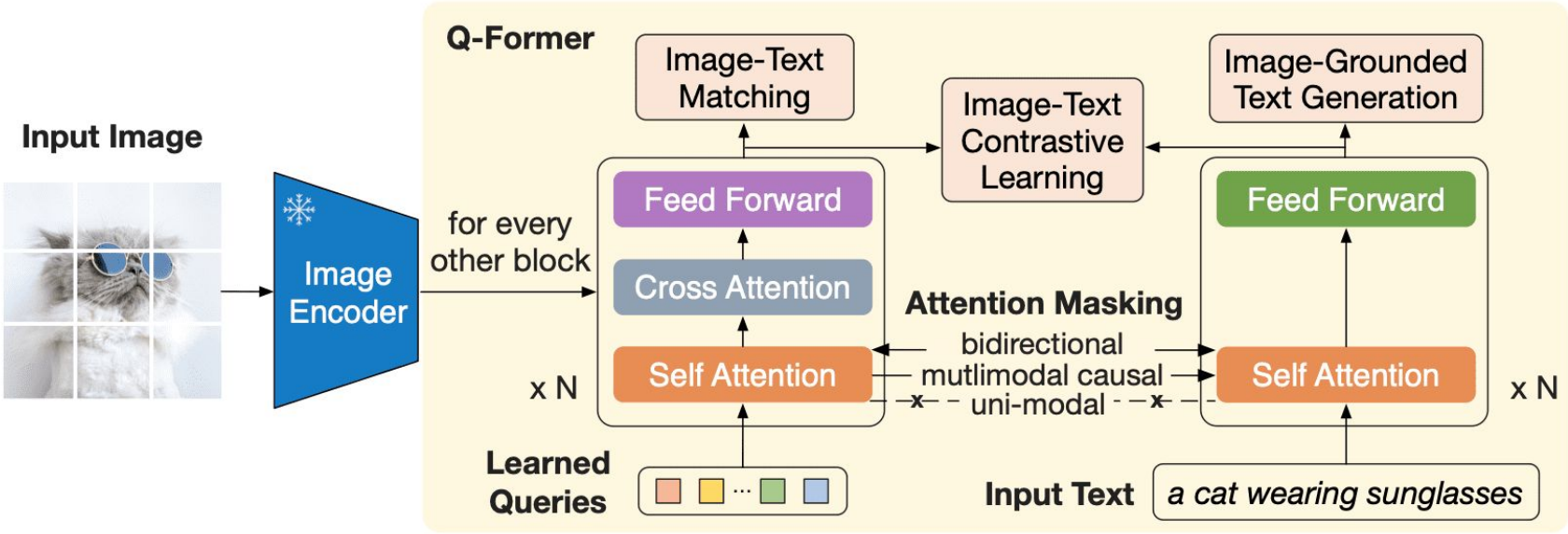
BLIP-2 - 2023

- **Q-Former:**
  - Init from Bert with additional Cross Attention layers
  - Learn a set of query embeddings of fixed shape (e.g. (32x1024))
  - **Objective**: find the 32 visual embeddings most relevant to a given input text

- **Differently from the Perceiver Resampler, the Q-Former is pre-trained on visual-textual inputs. By doing so, the visual queries are also influenced by the textual information**

Multimodal NLP

BLIP-2 - 2023



Multimodal NLP
BLIP-2 - 2023

- **NOTE:**
  - The Q-Former is a single model that processes both queries and textual inputs
  - Cross Attention is added when the input are the queries
  - The input of Self Attention is the concatenation of the queries and the text and the attention mask is modified to control what is attended in each training step
  - **Thanks to this design, the visual summary is influenced during pre-training by text**

Multimodal NLP
BLIP-2 - 2023

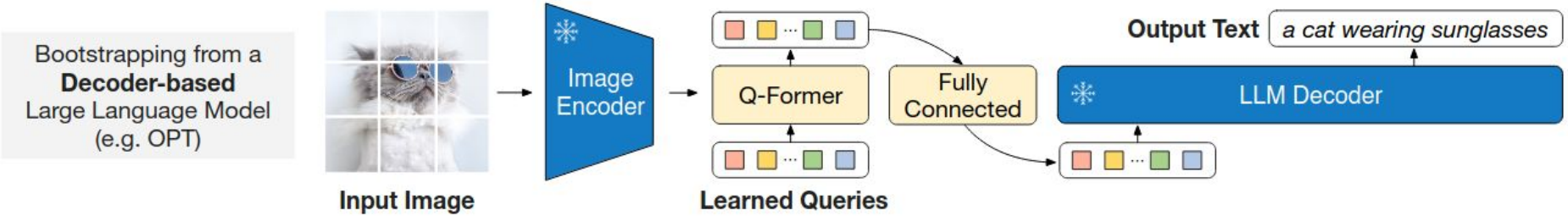
- **Q-Former Pre-Training:**
  - ITM: classifier-like training where the Q-Former has to predict if a text matches an image. Both images and text attend to the whole context (image+text)
  - ITG: generate text using images as conditioning
  - ITC: align images and texts (both attend to themselves only)

- **Using the same pre-training data, three different objectives are obtained by applying a different attention mask**

Multimodal NLP

BLIP-2 - 2023

- **After Q-Former pre-training:**
  - Project the visual features (queries) into the LLM space
  - **Similar to LLaVA, but only num_queries tokens rather than all patches**



Multimodal NLP
Chameleon - 2024

- **What if we wanted to represent images as a sequence of discrete tokens? How could we do that?**
  - Solution: Vector Quantization

Multimodal NLP

Chameleon Vector Quantization - 2024

- **Problem:** Map a given vector in a limited subset of possible vectors
- Compression technique

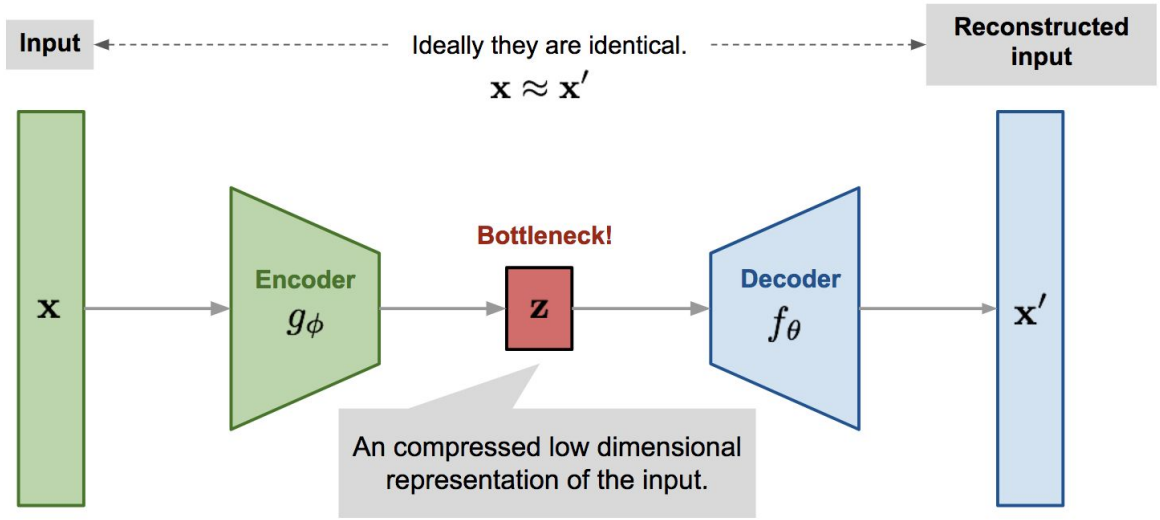
- **VQ:** Mapping function VQ
  - VQ: R**k -> C

- **R**k** space of dimension k
- **C = {y_1, y_2, ..., y_N}** codebook
- N is the size of the codebook and y_i is called code vector
- **We want to find the indices of the optimal code vector for a given vector x**

Multimodal NLP

Chameleon Vector Quantization - 2024

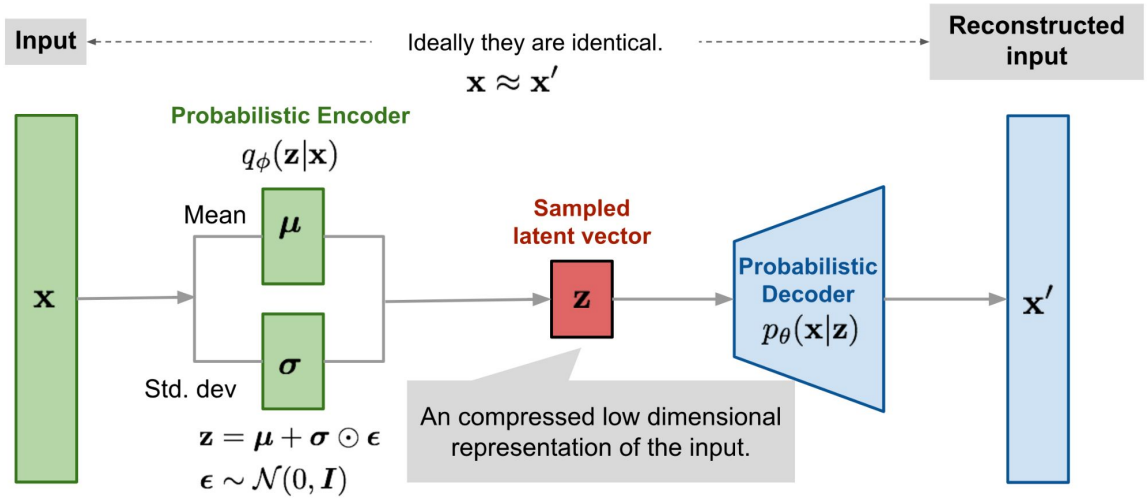
- **Remember: AutoEncoder**



Multimodal NLP

Chameleon Vector Quantization - 2024

- **Remember: VAE**



Multimodal NLP

Chameleon Vector Quantization - 2024

- **Solution: VQ-VAE**
  - After the encoding step of the VAE, map the latent vectors to the closest vector in the codebook
  - Note that the VQ-VAE trains the codebook, specifically, codebook vectors are pushed towards encoder outputs that have matched at each train step

Multimodal NLP
Chameleon - 2024

- **So what happens in Chameleon?**
  - Images are mapped into a fixed-length number of discrete visual tokens from the codebook
  - Specifically, it leverages both the encoder and codebook of the VQ-VAE
  - Texts are treated normally as if it was a standard LLM

- **We can now train the model using the classic LM objective**

Multimodal NLP

Chameleon (cool thing) - 2024

- **Given that images are modeled as a sequence of discrete tokens, the model is also able to generate said discrete tokens**
  - This means that we can train an image decoder to reconstruct images starting from the embedding representation of these discrete tokens
  - N.B. Unfortunately, the Chameleon checkpoint that has been released doesn't support this feature, the image tokens of the lm head are not trained

Notebook

Multimodal NLP Evaluation
Tasks

- **There are two types of tasks designed for LLMs and LVLMs:**
  - **Open-ended:** the model is expected to generate a complete answer in natural language for a given question
  - **Closed-ended:** the model is expected to generate the options' identifiers of the correct choices given a question and a list of possible options

Multimodal NLP Evaluation
Evaluation Strategies

- **There are three possible strategies to evaluate LLMs and LVLMs:**
  - **Log-likelihood :** evaluate choices based on probability of them being the next generated output of the LLM
  - **Generative:** let the LLM generate the output and use a metric to evaluate the goodness of the generation w.r.t. the labeled ground truth
  - **LLM-as-a-judge:** let a large LLM evaluate the output generated by a smaller LLM

Multimodal NLP Evaluation
Evaluation Strategies

- **All of these strategies have their limits:**
  - **Log-likelihood :** only considers the options that are given in the question; doesn't reflect how the model would be used in real world cases
  - **Generative:** extracting the answer from the label may be challenging
  - **LLM-as-a-judge:** output is dependent on the LLM judge, there is no guarantee of the correctness of the judging

Thanks!

elio.musacchio@uniba.it