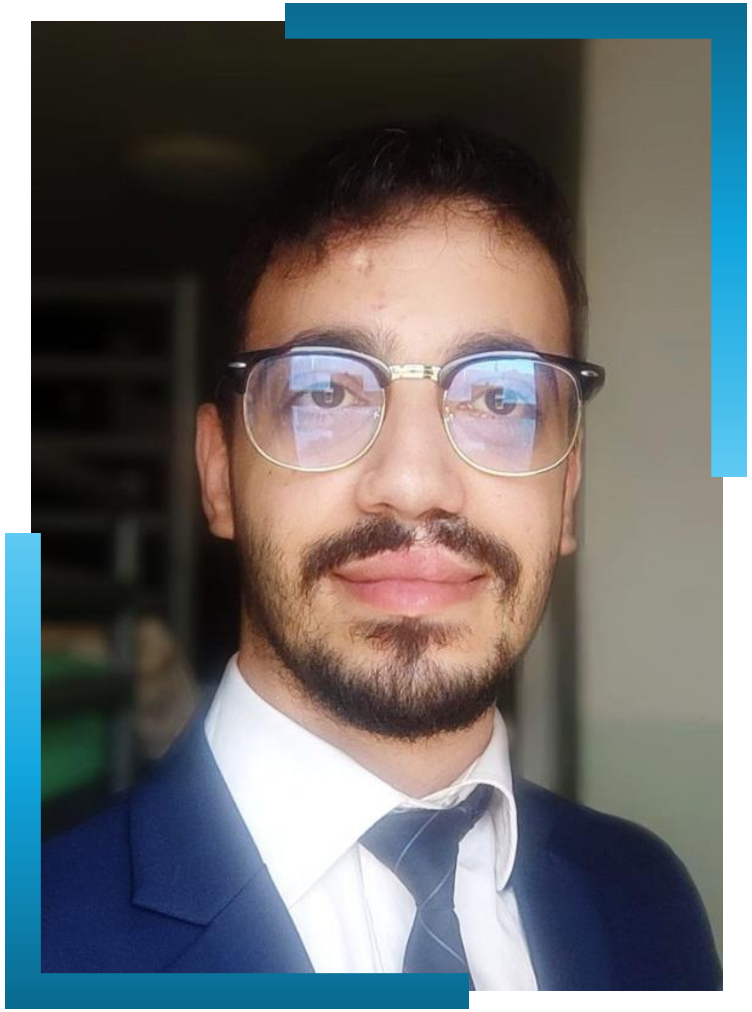
Transformers in Vision

AN INTRODUCTION

By:

Dr. Safouane El Ghazouali

BIO



Safouane El Ghazouali, Ph. D.

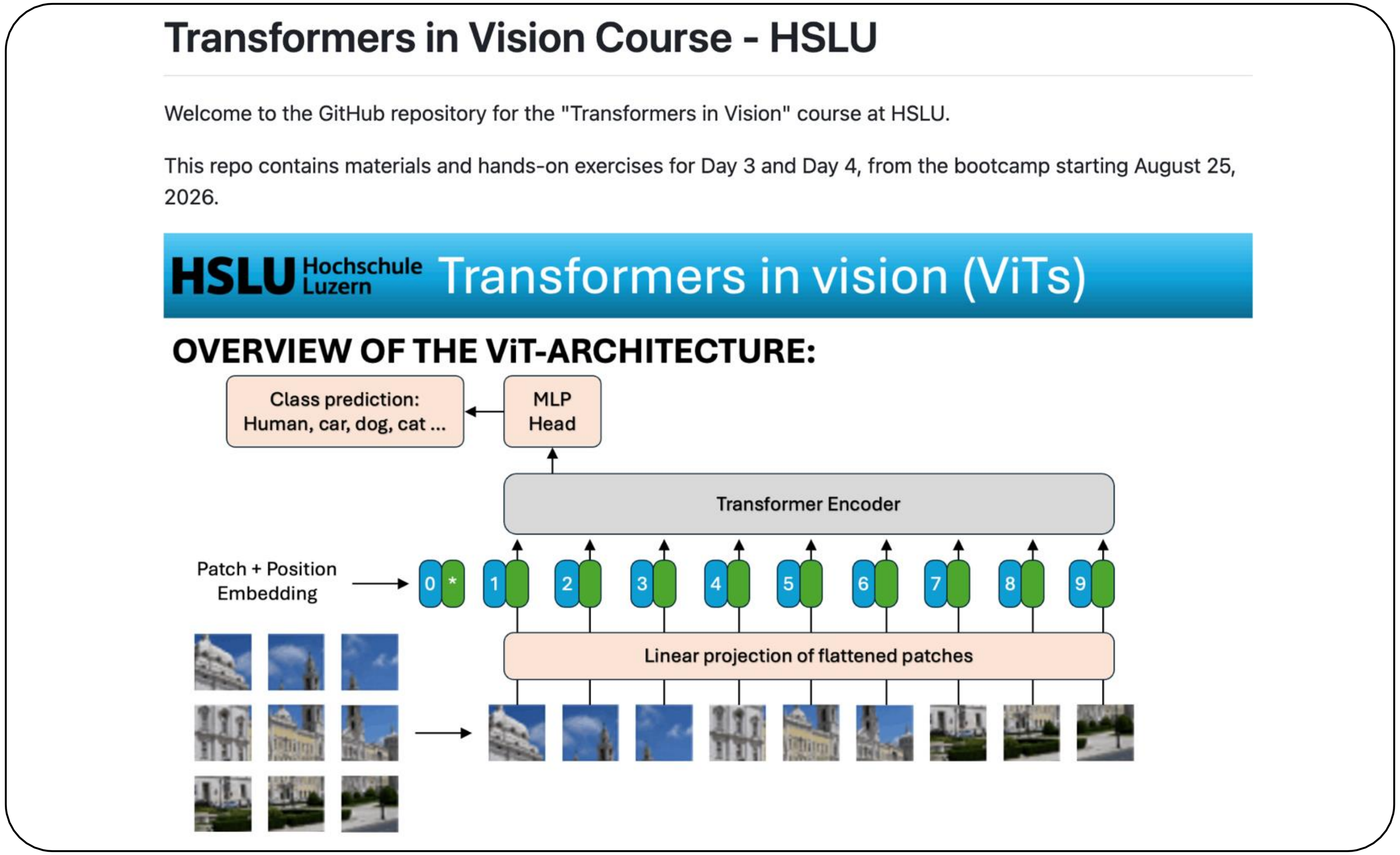
- Senior researcher in AI at TOELT
- Senior data scientist in Computer Vision
- Experienced in VLM / LLM inference and fine-tuning
- Buit a web app UI for fast annotation (VisioFirm)
- Experienced in training / fine-tuning of ML model for domain-specific application

Background

- Ph.D. in AI applied to 3D metrology, LNE, Paris
- Master in computer vision and embedded systems, Polytechnique Haut-de-France, Valenciennes

GitHub repo for the course :

https://github.com/safouaneelg/HSLU-Transformers-in-Vision



Transformers in Vision Course - HSLU

Welcome to the GitHub repository for the "Transformers in Vision" course at HSLU.

This repo contains materials and hands-on exercises for Day 3 and Day 4, from the bootcamp starting August 25, 2026.

HSLU Hochschule Luzern Transformers in vision (ViTs)

OVERVIEW OF THE ViT-ARCHITECTURE:

HSLU Hochschule Luzern

AGENDA. DAY 1

Vision Transformer

- General Introduction
- Overview of neural architectures evolution
- Transformers in vision (ViTs)
- Divers ViTs architectures
- Hands-on (using “Torch” and ”timm” frameworks):
  - Pre-trained classification model
  - Visualization of features map
  - Finetuning ViT on classification task
- ViTs in object detection
- Hands-on Ultralytics (YOLO models)
  - Inferencing pre-trained model
  - Finetuning YOLOv10 on custom dataset
- Hugginface Transformers library
- Limitations of ViTs

AGENDA. DAY 2

- Background & Motivations
- Applications: image captioning, retrieval, grounding, generative tasks
- CLIP and Zero-Shot Learning
- Hands-on:
    - Inference CLIP for zero-shot classification
    - GroundingDINO for zero-shot detection
    - Prompt Engineering
- BLIP and Multimodal Generation
- Diffusion Model – introduction to Markov Chain
- Hands-on:
    - Stable-diffusion
    - Prompt engineering

General introduction

HSLU Hochschule Luzern

General introduction

Computer vision includes:

1- Image/signal processing

2- 3D reconstruction

3- Feature extraction

4- Image restoration

5- Image classification

6- Object detection

7- Semantic/instance segmentation

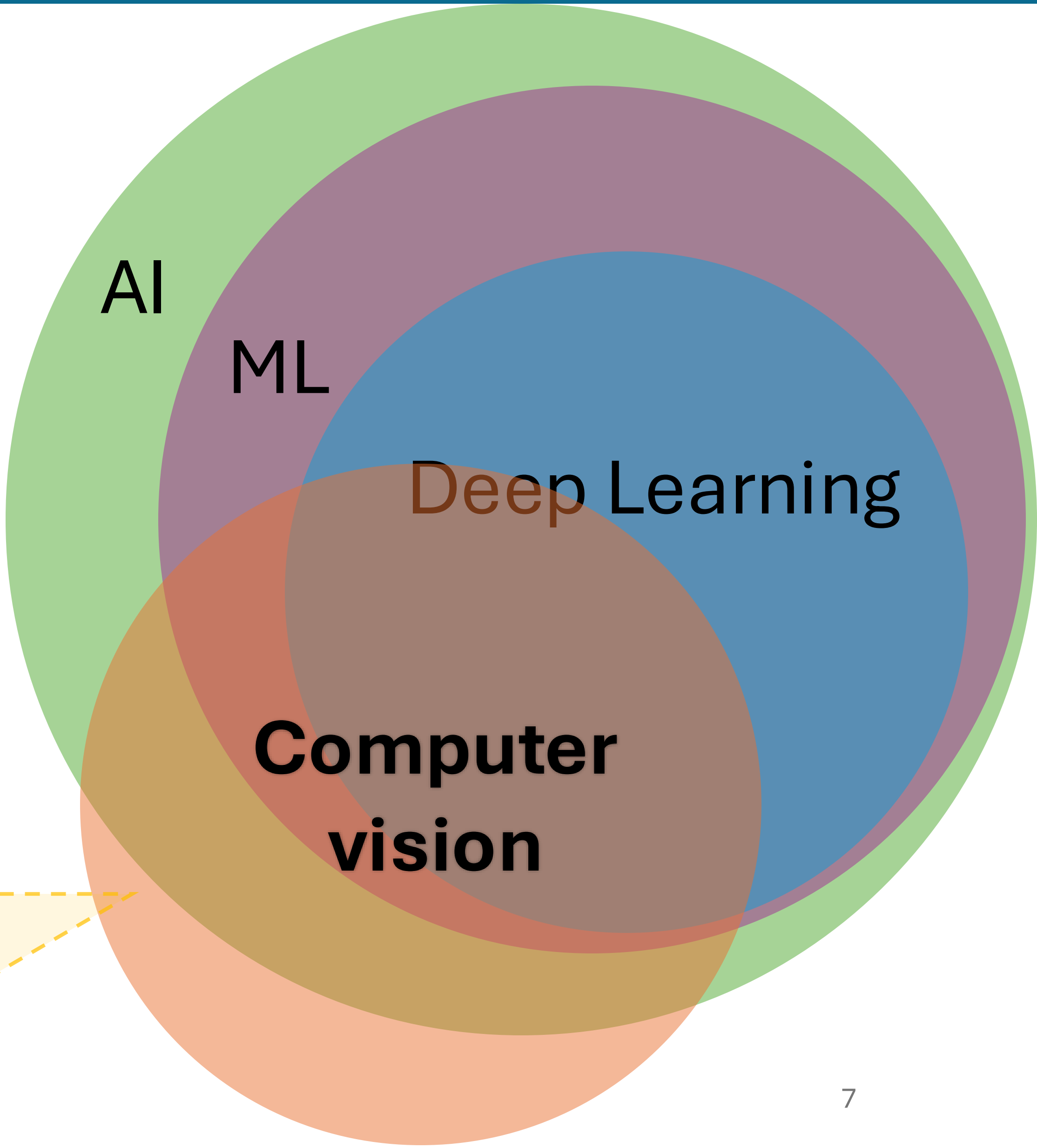
8- Scene understanding

9- Image generation

10- Optical flow analysis

11- Pose estimation

12- Video tracking



Overview of neural architectures evolution

Computer vision evolution timeline

1950

Foundations in optics and early mathematical-based processing. Focus on understanding light, vision, pinhole and basic digital imaging.

1970

Shift to more structured NNs with hierarchical feature learning. Introduction of convolutional ideas for shift-invariant pattern recognition.

2017

Emergence of advanced modular architectures and powerful backbones able to extract deep features.

The most known backbones

- VGGNet 16–19 layers with uniform 3x3 convolutions for depth exploration
- AlexNet: 8-layer CNN winning ImageNet challenge
- DenseNet: Layer-wise connections for feature reuse

2020

Rise of pure vision transformers, outperforming CNNs on large datasets. Multimodal integration begins.

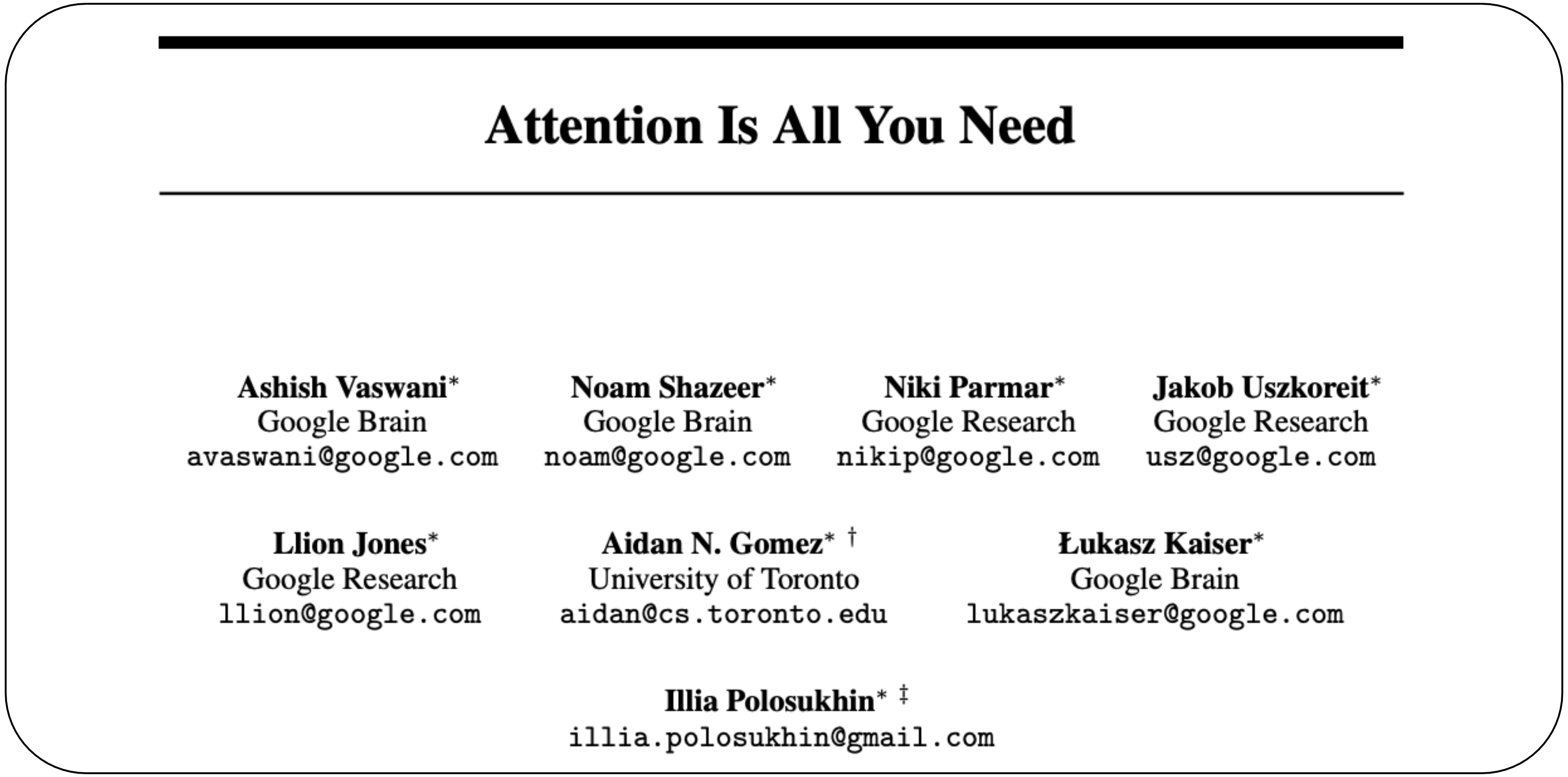
Vision Transformer: Patched images fed into transformers for classification.

DeiT (2021): Data-efficient ViT variant.

Swin Transformer (2021): Hierarchical shifted windows for efficiency.

Transformers in vision (ViTs)

Original Tranformer paper



Attention Is All You Need

Ashish Vaswani[*]
Google Brain
avaswani@google.com

Noam Shazeer[*]
Google Brain
noam@google.com

Niki Parmar[*]
Google Research
nikip@google.com

Jakob Uszkoreit[*]
Google Research
usz@google.com

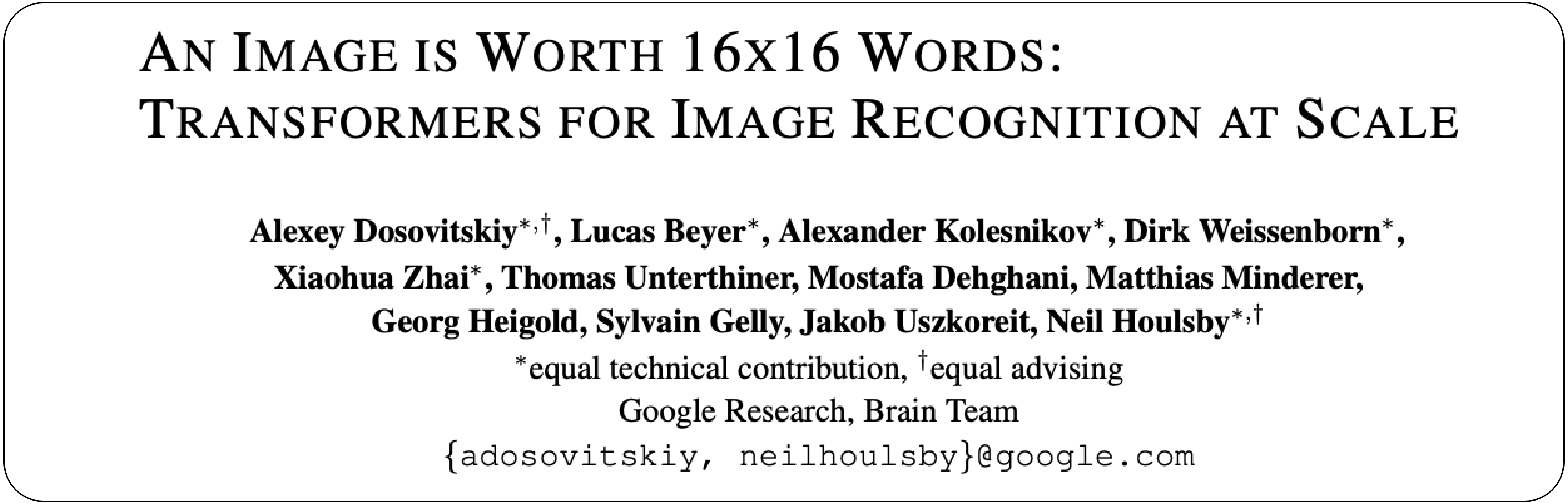
Llion Jones[*]
Google Research
llion@google.com

Aidan N. Gomez[*] [†]
University of Toronto
aidan@cs.toronto.edu

Łukasz Kaiser[*]
Google Brain
lukaszkaiser@google.com

Illia Polosukhin[*] [‡]
illia.polosukhin@gmail.com

First ViT paper



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

Transformers in vision (ViTs)

Objectives:

1. **Demonstrate the Applicability of Pure Transformers to Image Recognition**: Show that a standard Transformer architecture can be applied directly to sequences of image patches for classification tasks, without relying on convolutional neural networks (CNNs), thereby challenging CNN dominance in computer vision.
2. **Evaluate Performance on Large-Scale Pre-Training and Transfer Learning**: Assess the Vision Transformer (ViT) model's performance when pre-trained on massive datasets (e.g., ImageNet-21k or JFT-300M) and fine-tuned on various benchmarks (e.g., ImageNet, CIFAR-100, VTAB), comparing it to state-of-the-art CNNs.
3. **Investigate the Impact of Dataset Scale on Transformer Performance**: Explore how training dataset size influences ViT's effectiveness, emphasizing that large-scale data can compensate for Transformers' lack of built-in inductive biases like those in CNNs (e.g., translation equivariance and locality).
4. **Compare Computational Efficiency and Resource Requirements**: Highlight that ViT achieves strong results with significantly lower computational costs during training compared to leading CNNs, using metrics such as TPUv3-core-days.

Transformers in vision (ViTs)

Aspect	NLP Tokenization	Vision Tokenization
Definition	Process of breaking down text into smaller units (tokens) such as words, subwords, or characters.	Process of dividing an image into fixed-size patches or regions to process as tokens.
Normalization	- Converts text to a standard format (e.g., lowercasing, removing accents). - Removes punctuation or special characters. - Handles contractions (e.g., "don't" → "do not").	- Normalizes pixel values (e.g., scaling to [0,1] or standardizing with mean and std). - Adjusts color channels (e.g., RGB normalization). - Resizes images to a fixed resolution.
Segmentation/Splitting	- Splits text into tokens based on whitespace, punctuation, or subword units (e.g., WordPiece, BPE). - Example: "I am running" → ["I", "am", "running"]. - Subword tokenization for rare words (e.g., "unhappiness" → ["un", "##happiness"]).	- Divides image into fixed-size patches (e.g., 16x16 pixels). - Each patch is treated as a token. - Example: A 224x224 image with 16x16 patches → 196 patches (tokens).
Cleanup	- Removes irrelevant tokens (e.g., extra whitespace, stop words like "the"). - Filters out low-frequency tokens or replaces them with [UNK]. - Handles encoding issues (e.g., UTF-8 normalization).	- Removes or adjusts noisy patches (e.g., handling corrupted pixels). - Applies data augmentation (e.g., random cropping, flipping) to enhance robustness. - Filters out irrelevant regions (e.g., background suppression in some cases).
Token Representation	- Tokens are mapped to vocabulary indices or embeddings (e.g., Word2Vec, BERT embeddings). - Each token is a discrete unit in a vocabulary.	- Patches are flattened and projected to a fixed-dimensional embedding space (e.g., linear projection in ViT). - Each patch is a vector in a continuous space.
Handling Variability	- Handles linguistic variations (e.g., synonyms, misspellings, multilingual text). - Uses subword tokenization to manage out-of-vocabulary words.	- Handles variations in lighting, orientation

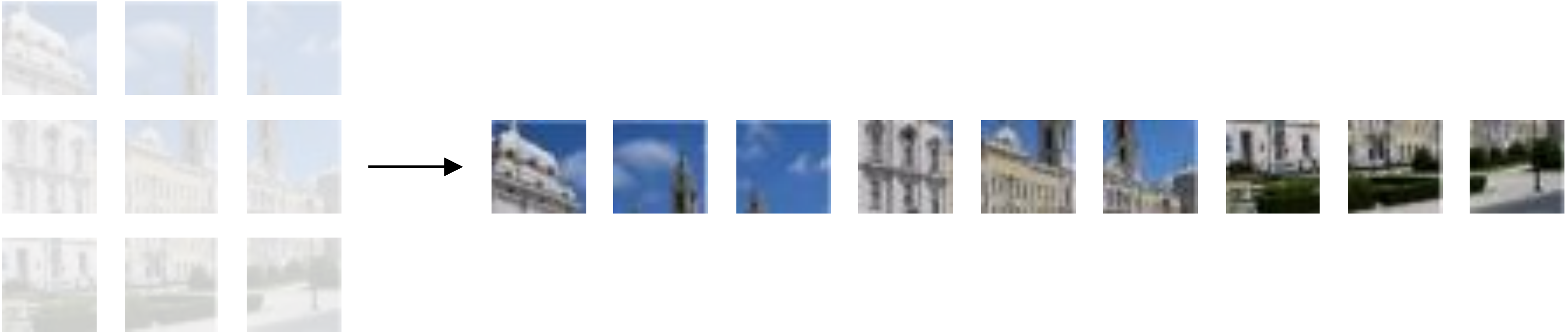
OVERVIEW OF THE ViT-ARCHITECTURE:

Transformers in vision (ViTs)

OVERVIEW OF THE ViT-ARCHITECTURE:

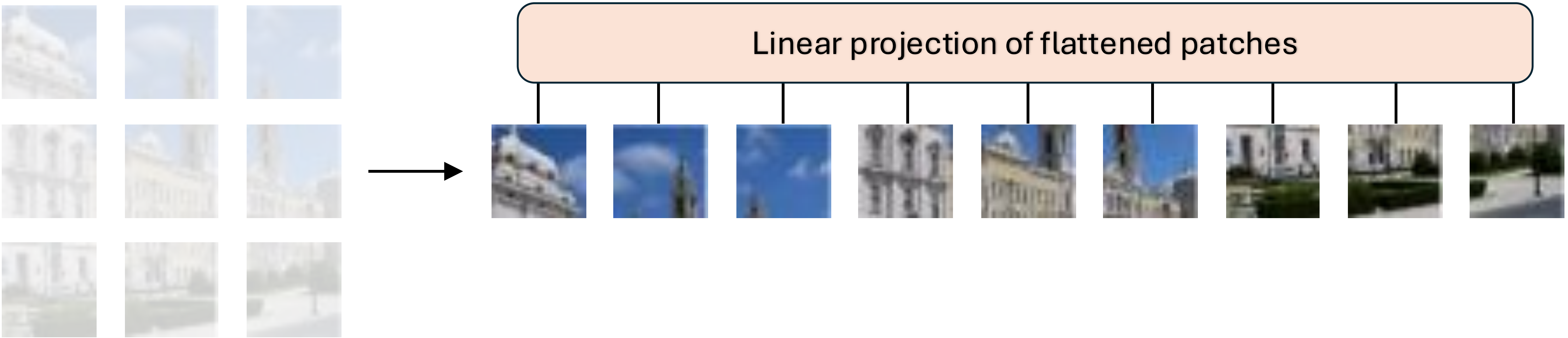
Transformers in vision (ViTs)

OVERVIEW OF THE ViT-ARCHITECTURE:



Transformers in vision (ViTs)

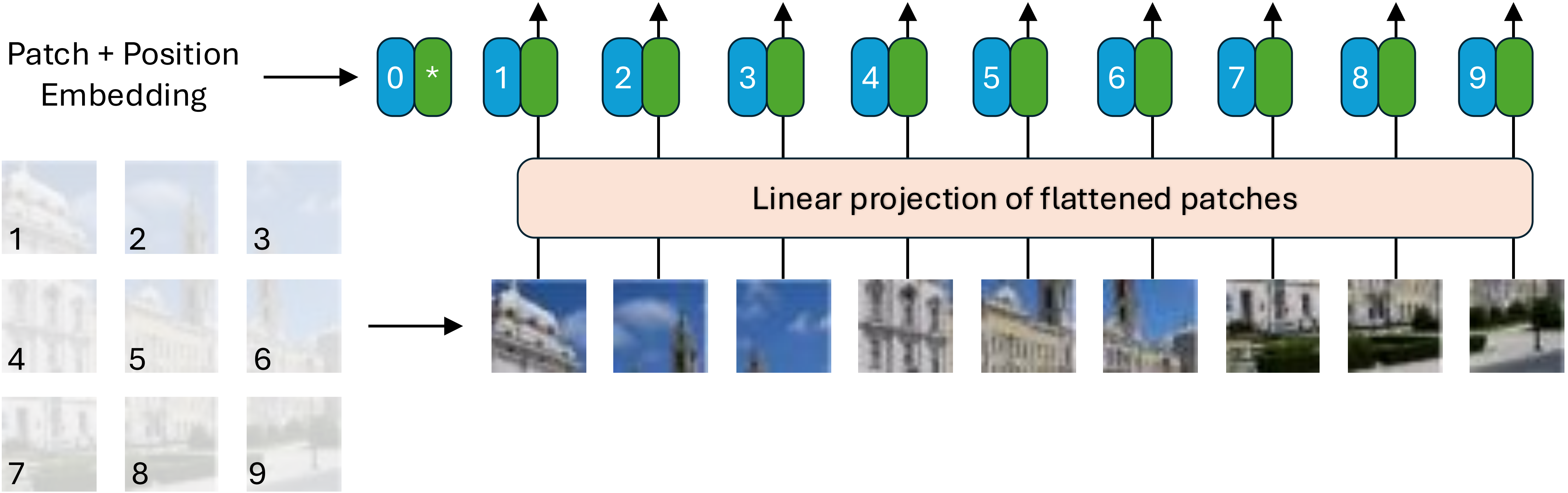
OVERVIEW OF THE ViT-ARCHITECTURE:



Linear projection of flattened patches

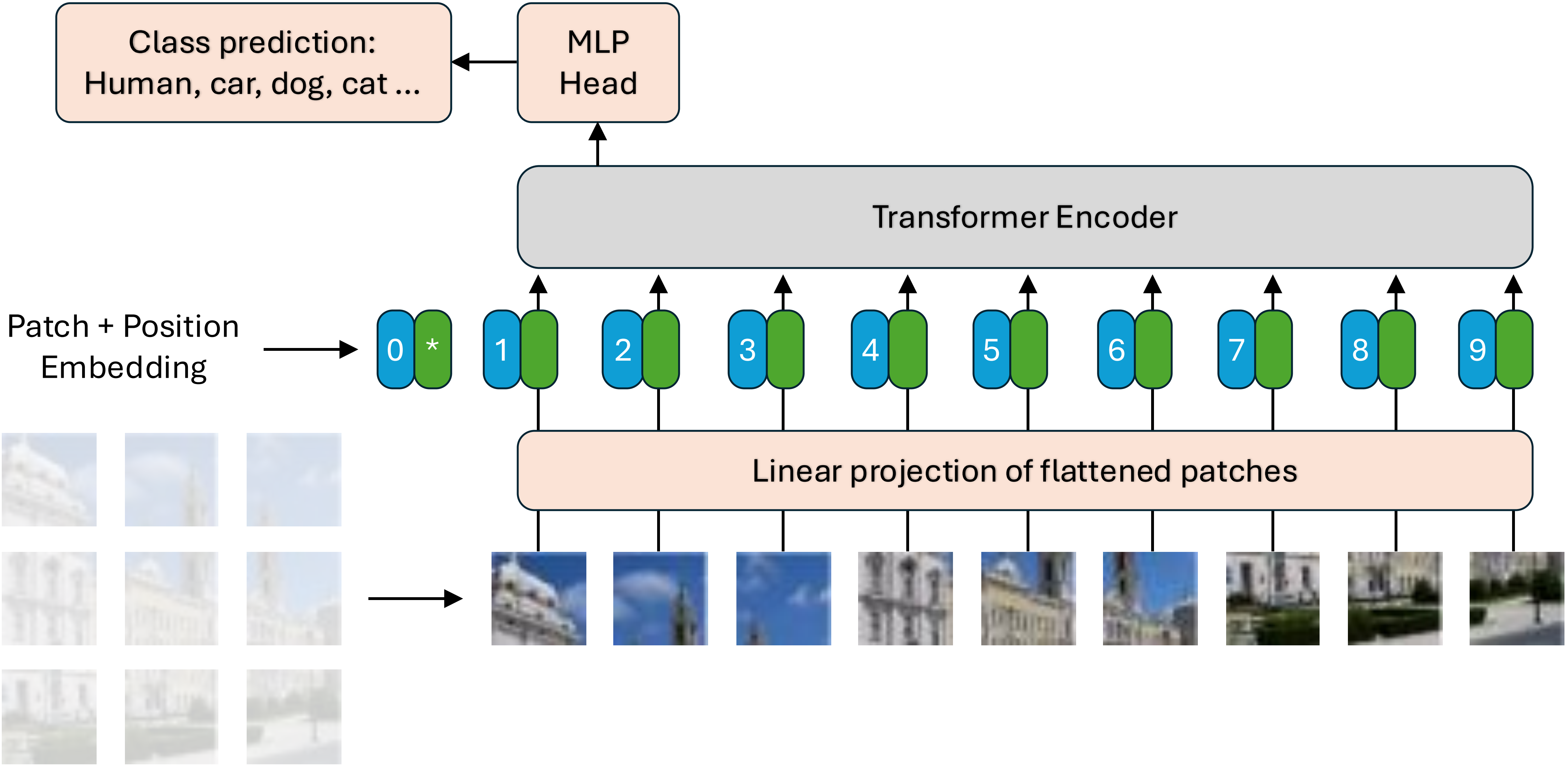
Transformers in vision (ViTs)

OVERVIEW OF THE ViT-ARCHITECTURE:



Transformers in vision (ViTs)

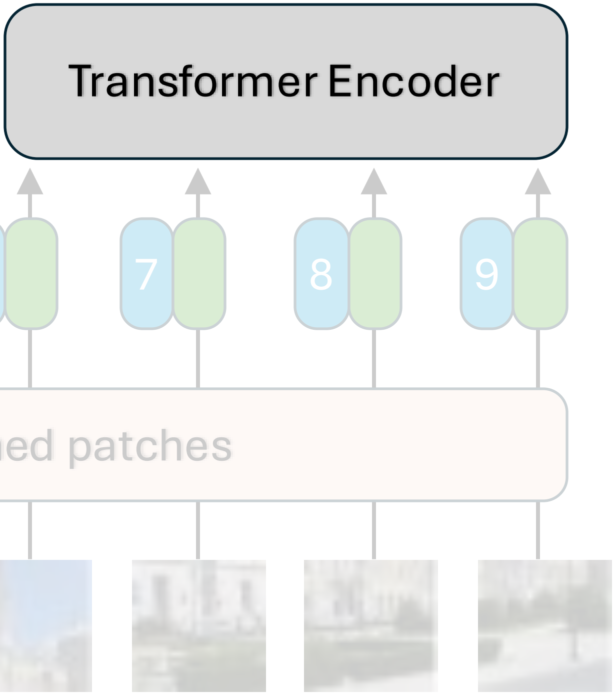
OVERVIEW OF THE ViT-ARCHITECTURE:



HSLU Hochschule Luzern

Transformers in vision (ViTs)

OVERVIEW OF THE ViT-ARCHITECTURE:



What is the "Transformer Encoder" how different / similar to NLP is it?

Both encoders are fundamentally similar in architecture and operation, but there are key differences due to the nature of the input data

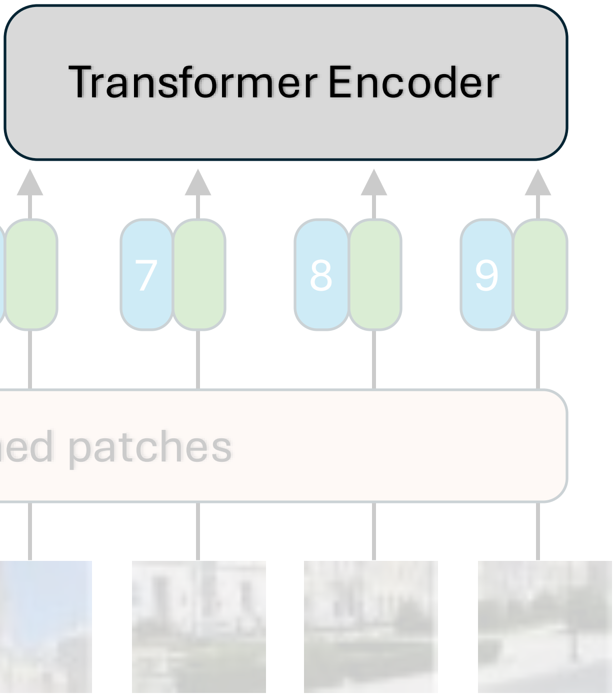
Core Similarity

- **Multi-Head Self-Attention**: Computes relationships between all input tokens/patches, capturing global dependencies.
- **Feed-Forward Networks (FFNs)**: Applied independently to each token/patch for feature transformation.
- **Layer Normalization and Residual Connections**: Stabilize training and improve gradient flow.
- **Positional Encoding/Embeddings**: Preserve sequence or spatial order.

The encoder processes a sequence of input tokens (or patches in ViTs) and produces contextualized representations by attending to all elements in the sequence simultaneously.

Transformers in vision (ViTs)

OVERVIEW OF THE ViT-ARCHITECTURE:



What is the “Transformer Encoder” how different / similar to NLP is it?

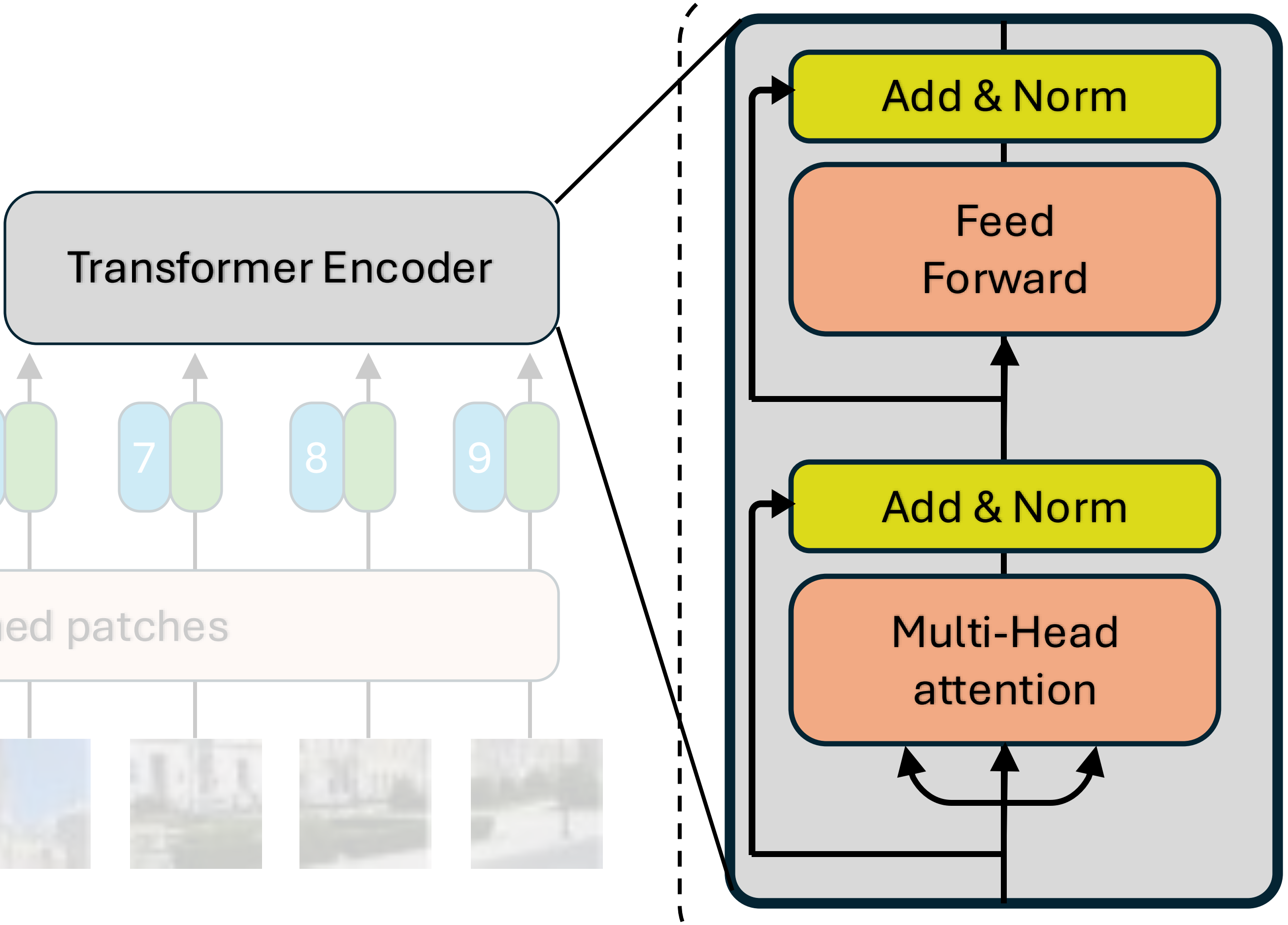
Differences

- **Input**
  - **NLP**: Takes a sequence of tokenized text (such as words, subwords) and tokens are mapped to embeddings via a vocabulary lookup (e.g., WordPiece)
  - **Vision**: Takes a sequence of image patches (e.g., 16x16 pixel patches). And the patches are flattened and linearly projected to a fixed-dimensional embedding space.
- **Positional Encoding**
  - **NLP**: In natural language, the input is a 1D sequence of words. The position can be represented by a single index, like 1, 2, 3, and so on. The positional encoding is therefore a 1D vector corresponding to each word's index.
  - **Vision**: Images are 2D. The position of a patch is defined by its row and column coordinates. While some ViTs use a flattened 1D sequence of patches (e.g., from top-left to bottom-right), more advanced methods might use a 2D encoding that explicitly models the row and column positions

HSLU Hochschule Luzern

Transformers in vision (ViTs)

OVERVIEW OF THE ViT-ARCHITECTURE:



Residual Connection & **Layer Normalization**
(Skip connections)

$$head_i = Attention(Q * W_i^Q, K * W_i^K, V * W_i^V)$$

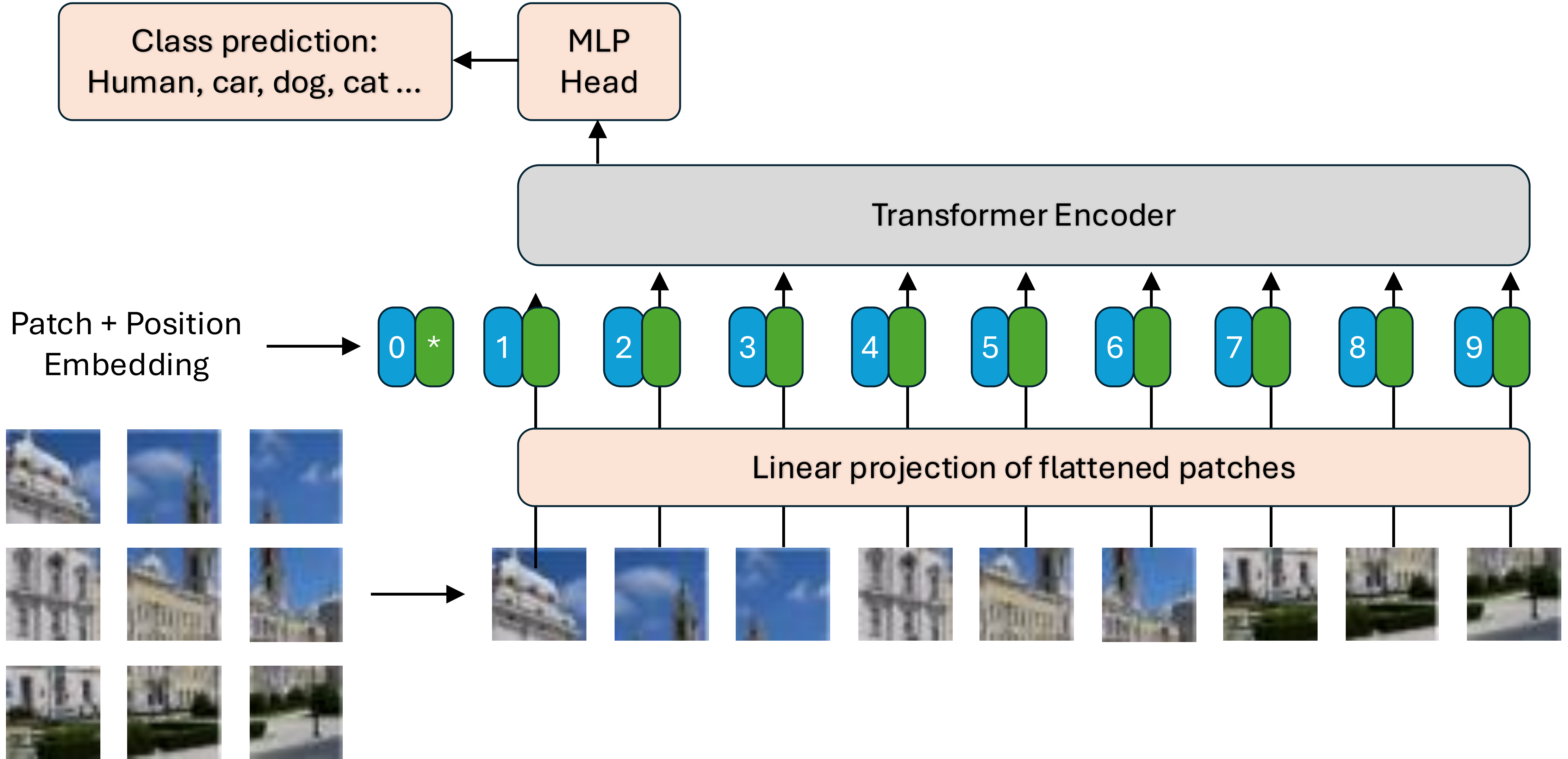
$$\mathrm{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^\top}{\sqrt{d_k}})V$$

Q (Query), K (Key), and V (Value)

Divers ViTs architectures

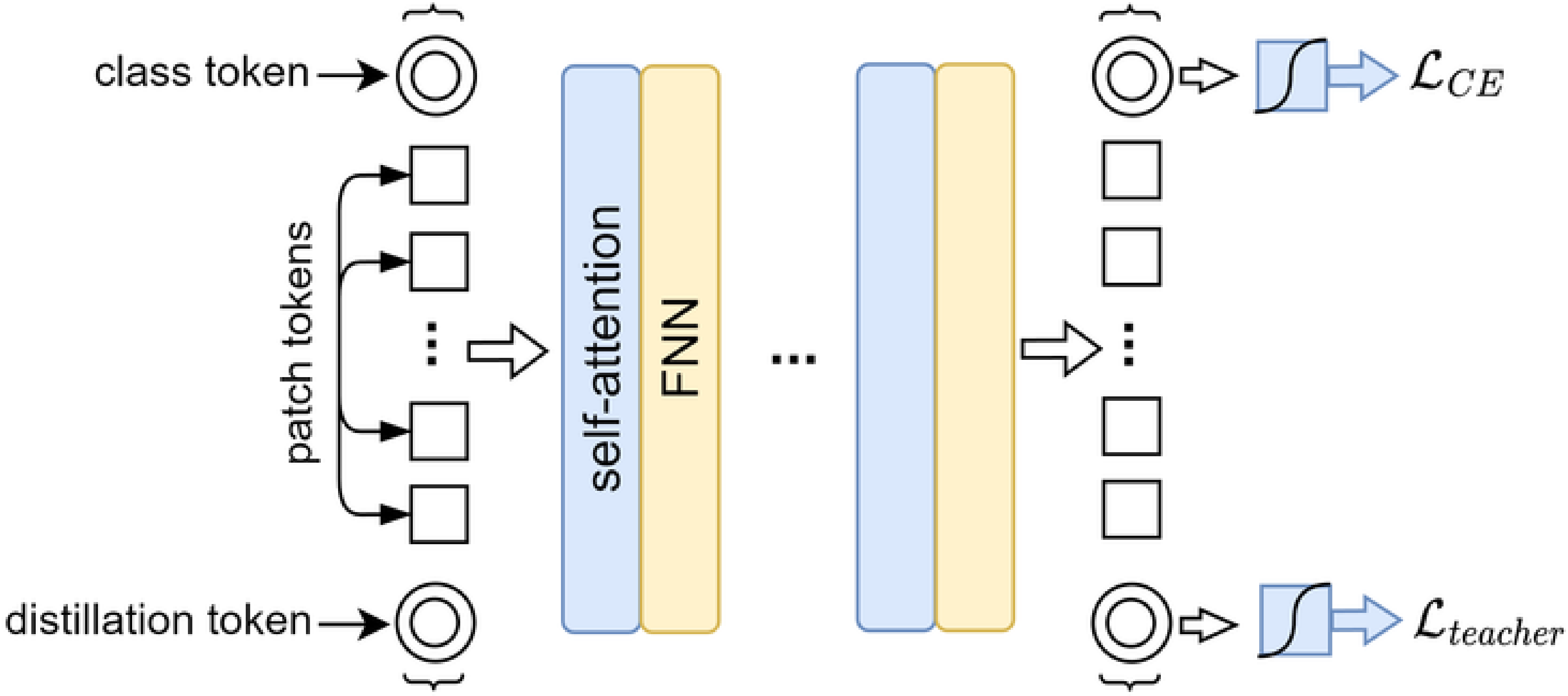
Divers ViTs architectures

ViT : Google - Vision Transformer (paper)



Divers ViTs architectures

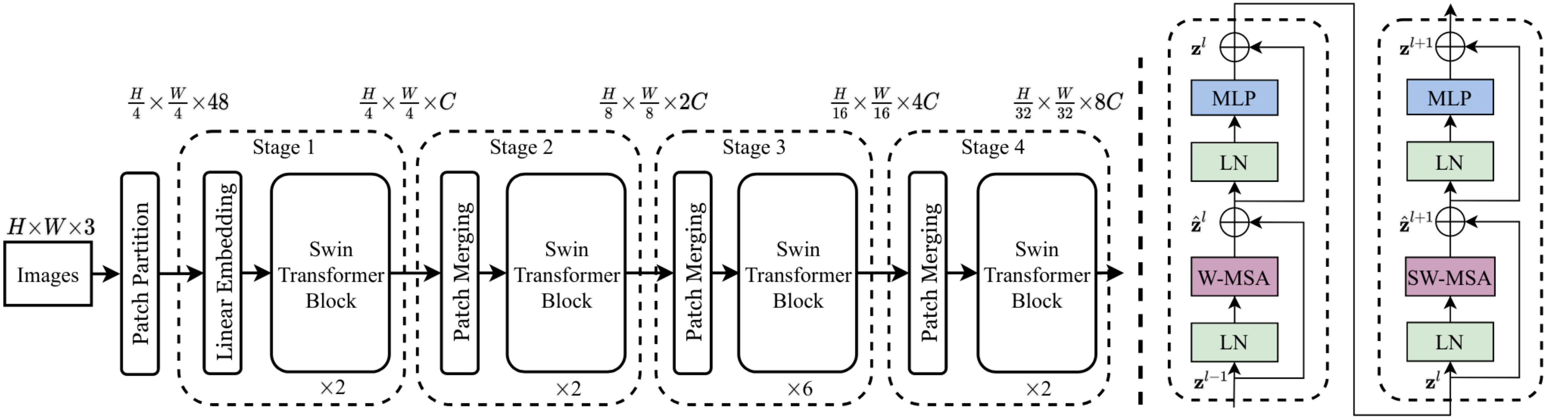
DeiT : Meta - Data efficient Image Transformer (paper)



HSLU Hochschule Luzern

Divers ViTs architectures

Microsoft – Swin-Transformer (paper)



(a) Architecture

(b) Two Successive Swin Transformer Blocks

PyTorch

Hands-on:

(1) Pre-trained classification model

timm

PyTorch

Hands-on:

(2) Visualization of features map

timm

HSLU Hochschule Luzern

Hands-on:

(3) Finetuning ViT on classification task

ViTs in object detection

ViTs in object detection

DETR: End-to-end Detection TRansformer (paper)

Backbone Encoder Decoder Prediction heads

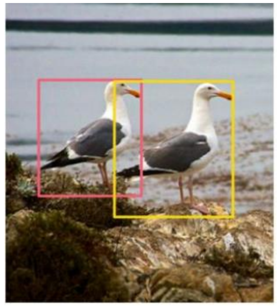


Diagram: YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions, Nidhal Jegham1,2, Chan Young Koh 2 Marwan Abdelatti2,3, and Abdeltawab Hendawi2 1 Tunis Business School, University of Tunis. (https://arxiv.org/html/2411.00201v3)

HSLU Hochschule Luzern

ViTs in object detection

DETR: End-to-end Detection TRansformer (paper)

Backbone   Encoder   Decoder   Prediction heads

Pretrained CNN: VGG – ResNet …etc

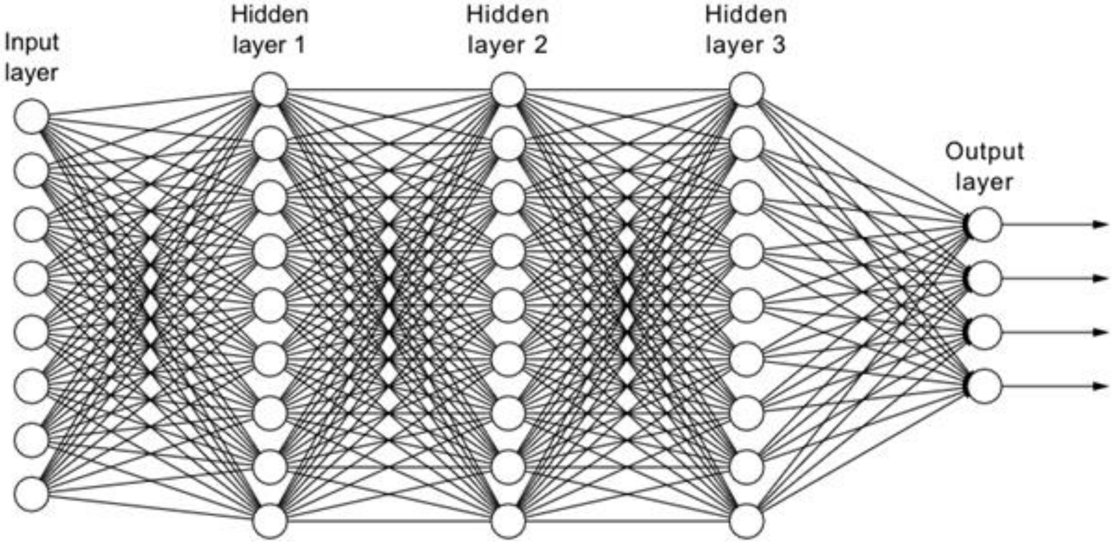


Diagram: YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions, Nidhal Jegham1,2, Chan Young Koh 2 Marwan Abdelatti2,3, and Abdeltawab Hendawi2 1 Tunis Business School, University of Tunis. (https://arxiv.org/html/2411.00201v3)

ViTs in object detection

DETR: End-to-end Detection TRansformer (paper)

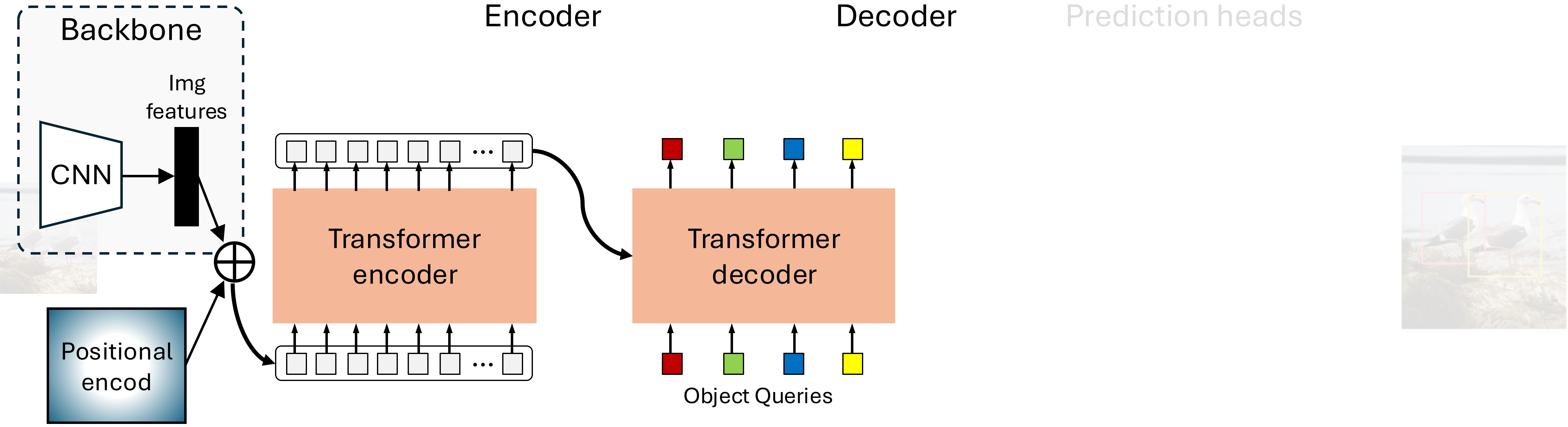


Diagram: YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions, Nidhal Jegham1,2, Chan Young Koh 2 Marwan Abdelatti2,3, and Abdeltawab Hendawi2 1 Tunis Business School, University of Tunis. (https://arxiv.org/html/2411.00201v3)

ViTs in object detection

DETR: End-to-end Detection TRansformer (paper)

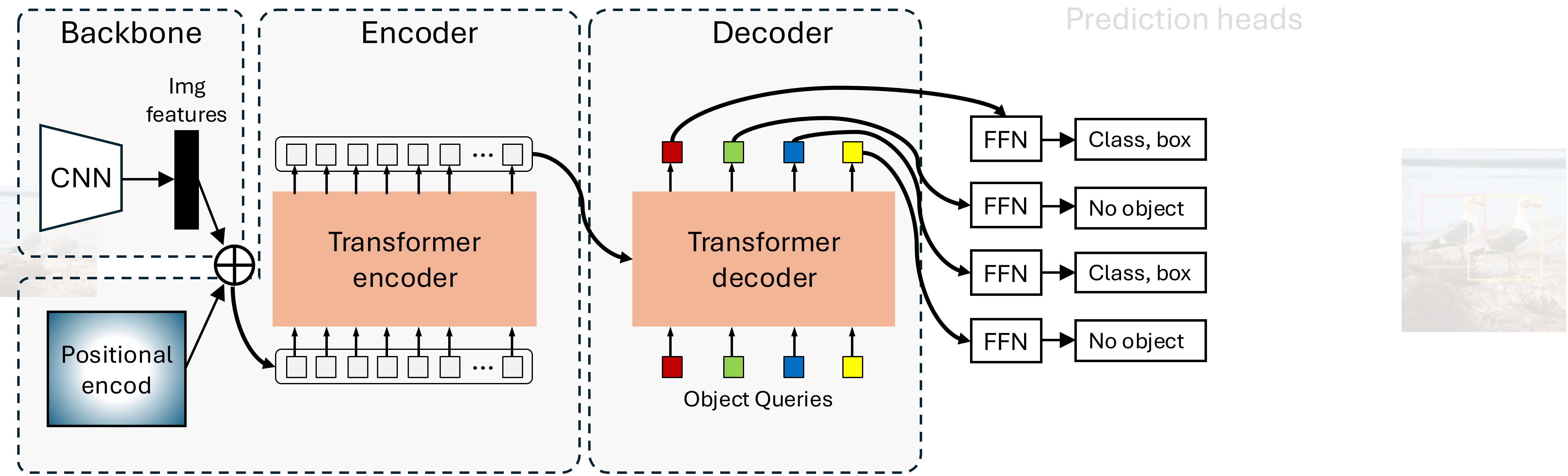


Diagram: YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions, Nidhal Jegham1,2, Chan Young Koh 2 Marwan Abdelatti2,3, and Abdeltawab Hendawi2 1 Tunis Business School, University of Tunis. (https://arxiv.org/html/2411.00201v3)

ViTs in object detection

DETR: End-to-end Detection TRansformer (paper)

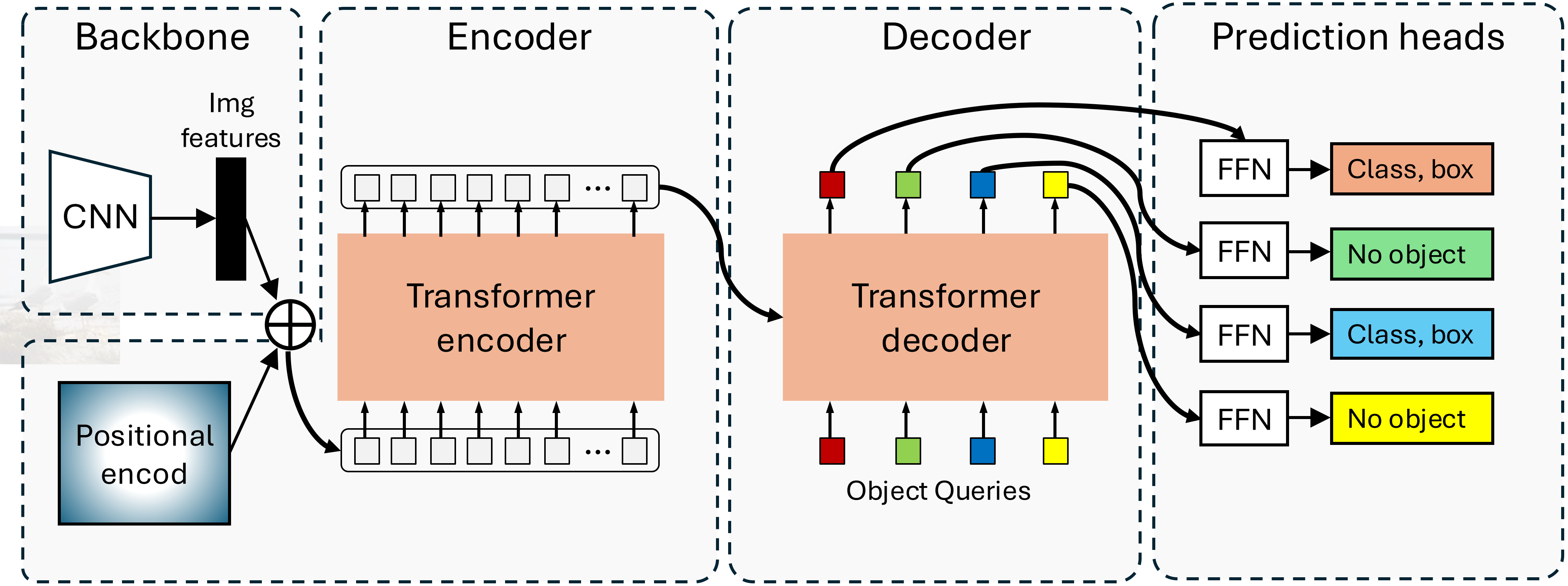


Diagram: YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions, Nidhal Jegham1,2, Chan Young Koh 2 Marwan Abdelatti2,3, and Abdeltawab Hendawi2 1 Tunis Business School, University of Tunis. (https://arxiv.org/html/2411.00201v3)

HSLU Hochschule Luzern

ViTs in object detection

DETR: End-to-end Detection TRansformer (paper)

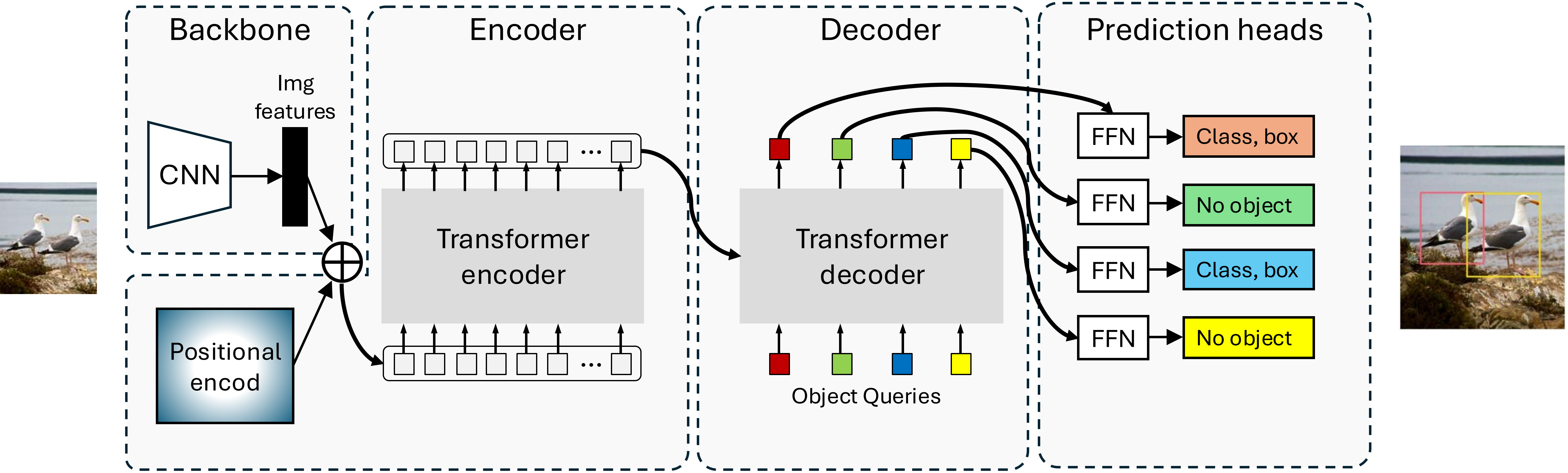


Diagram: YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions, Nidhal Jegham1,2, Chan Young Koh 2 Marwan Abdelatti2,3, and Abdeltawab Hendawi2 1 Tunis Business School, University of Tunis. (https://arxiv.org/html/2411.00201v3)

ViTs in object detection

YOLO series - v12: Attention-centric object detection

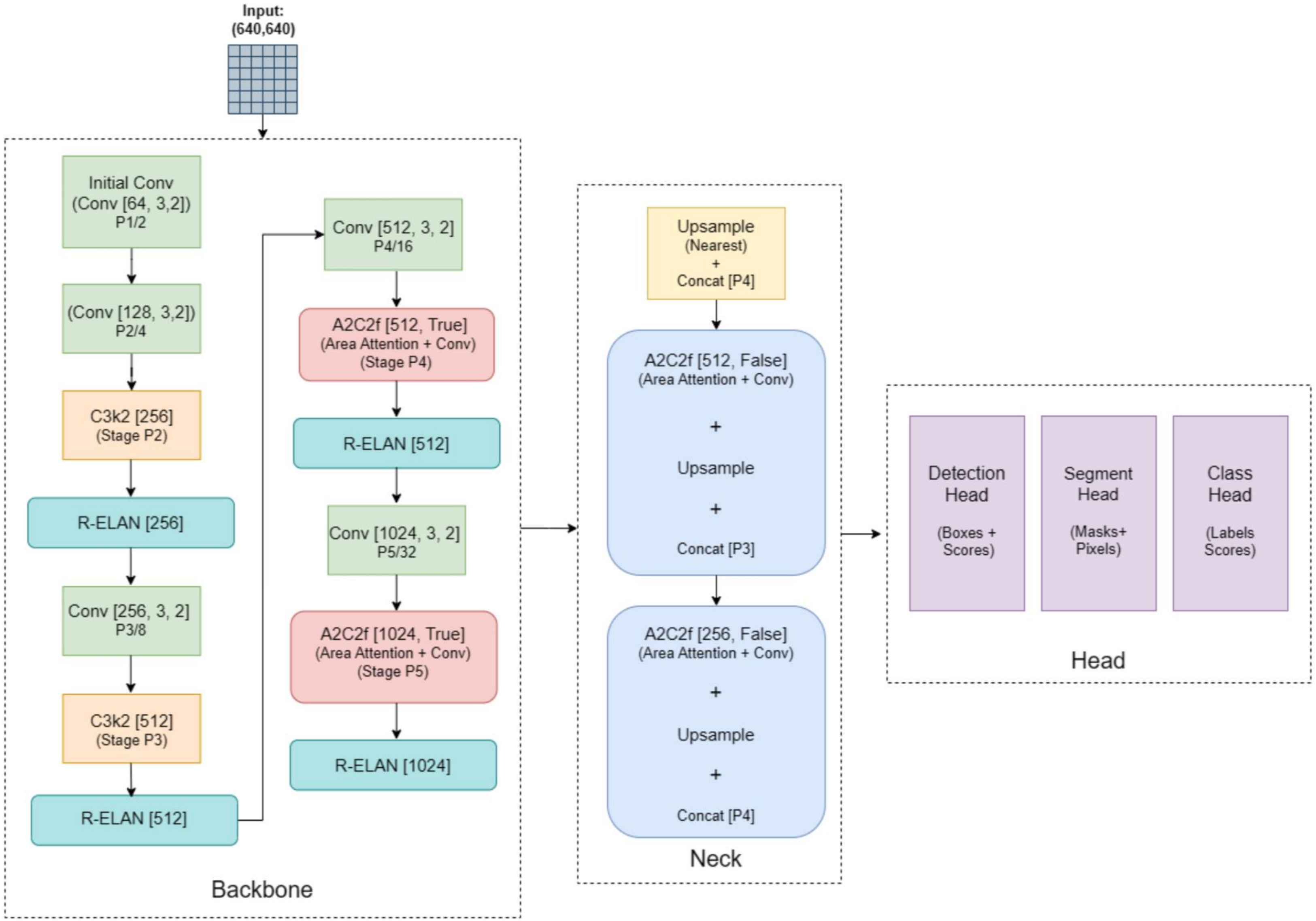

Input:
(640,640)
Initial Conv (Conv [64, 3,2]) P1/2
(Conv [128, 3,2]) P2/4
C3k2 [256] (Stage P2)
R-ELAN [256]
Conv [256, 3, 2] P3/8
C3k2 [512] (Stage P3)
R-ELAN [512]
Conv [512, 3, 2] P4/16
A2C2f [512, True] (Area Attention + Conv) (Stage P4)
R-ELAN [512]
Conv [1024, 3, 2] P5/32
A2C2f [1024, True] (Area Attention + Conv) (Stage P5)
R-ELAN [1024]
Backbone
Upsample (Nearest) + Concat [P4]
A2C2f [512, False] (Area Attention + Conv) + Upsample + Concat [P3]
A2C2f [256, False] (Area Attention + Conv) + Upsample + Concat [P4]
Neck
Detection Head (Boxes + Scores)
Segment Head (Masks+ Pixels)
Class Head (Labels Scores)
Head

Diagram: YOLO Evolution: A Comprehensive Benchmark and Architectural Review of YOLOv12, YOLO11, and Their Previous Versions, Nidhal Jegham1,2, Chan Young Koh 2 Marwan Abdelatti2,3, and Abdeltawab Hendawi2 1 Tunis Business School, University of Tunis. (https://arxiv.org/html/2411.00201v3)

Hands-on:

(1) Exploring models – inferencing

ultralytics

Hands-on:

(2) Finetuning object detection model on custom dataset

Hugging Face Transformers library

AGENDA. DAY 2

- Background & Motivations
- Applications: image captioning, retrieval, grounding, generative tasks
- CLIP and Zero-Shot Learning
- Hands-on:
  - Inference CLIP for zero-shot classification
  - GroundingDINO for zero-shot detection
  - Prompt Engineering
- BLIP and Multimodal Generation
- Diffusion Model – introduction to Markov Chain
- Hands-on:
  - Stable-diffusion
  - Prompt engineering

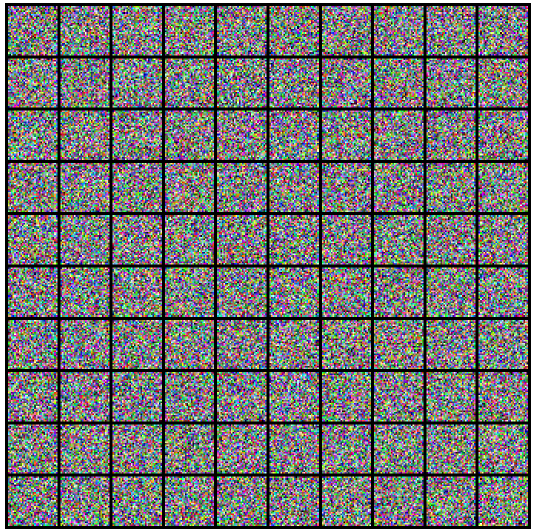
Background & Motivations

Background & Motivations

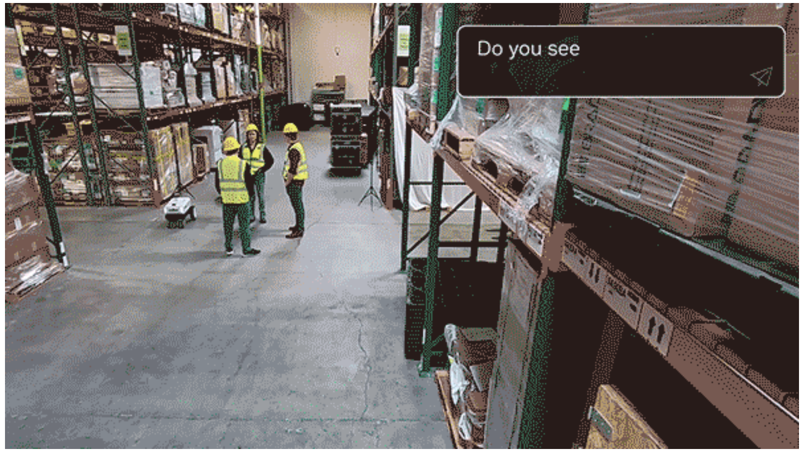
Introduction to Generative AI Fundamentals: Previous generative AI models rely on GAN models. Attention-Generative AI as models that create new data (e.g., images, text) from learned patterns in training data, contrasting it with discriminative models.

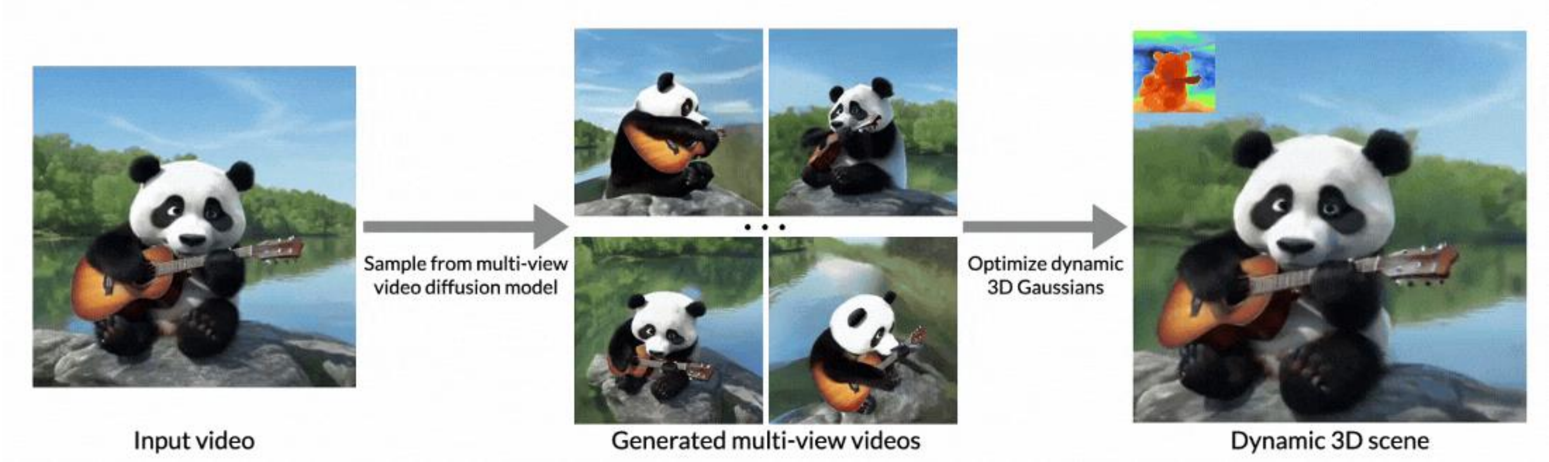
Motivations: solving data scarcity (e.g., synthetic data for training), creativity (art, design), personalization (e.g., custom content), and efficiency (automating tasks like content creation). Highlight real-world drivers like computational power growth, large datasets (e.g., LAION-5B), and ethical considerations (e.g., bias in generated outputs).

Synthetic image creation



Captioning/VQA

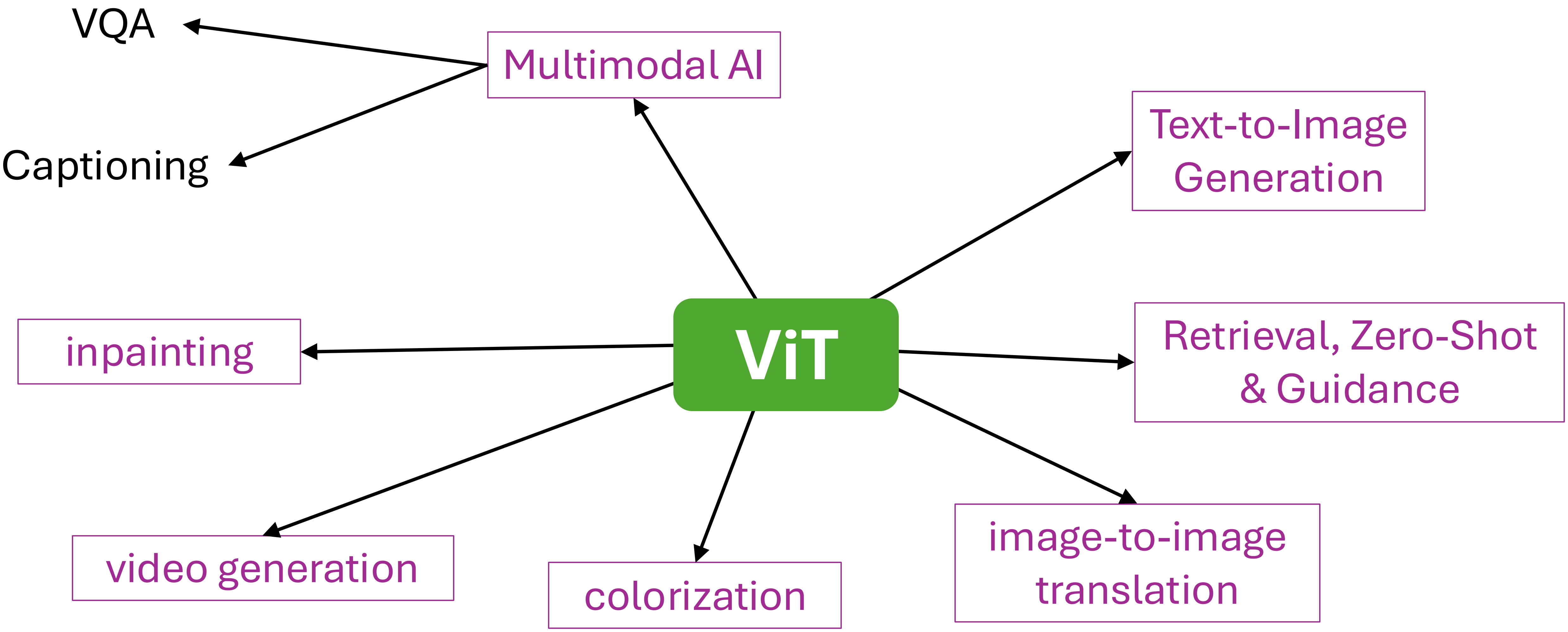



Sample from multi-view video diffusion model
Optimize dynamic 3D Gaussians
Input video
Generated multi-view videos
Dynamic 3D scene

Applications of Vision Transformers in Gen AI

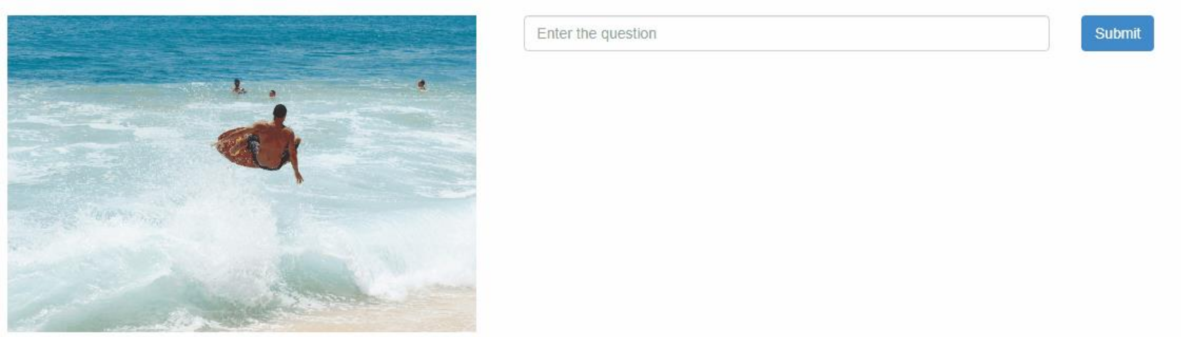
HSLU Hochschule Luzern

Applications in Generative AI



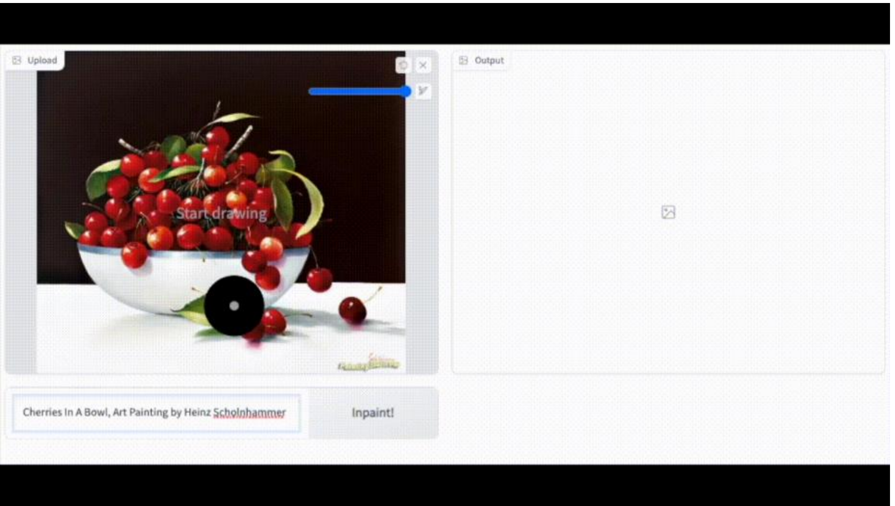
Applications in Generative AI

Multimodal AI: VQA



Text-to-Image Generation

inpainting



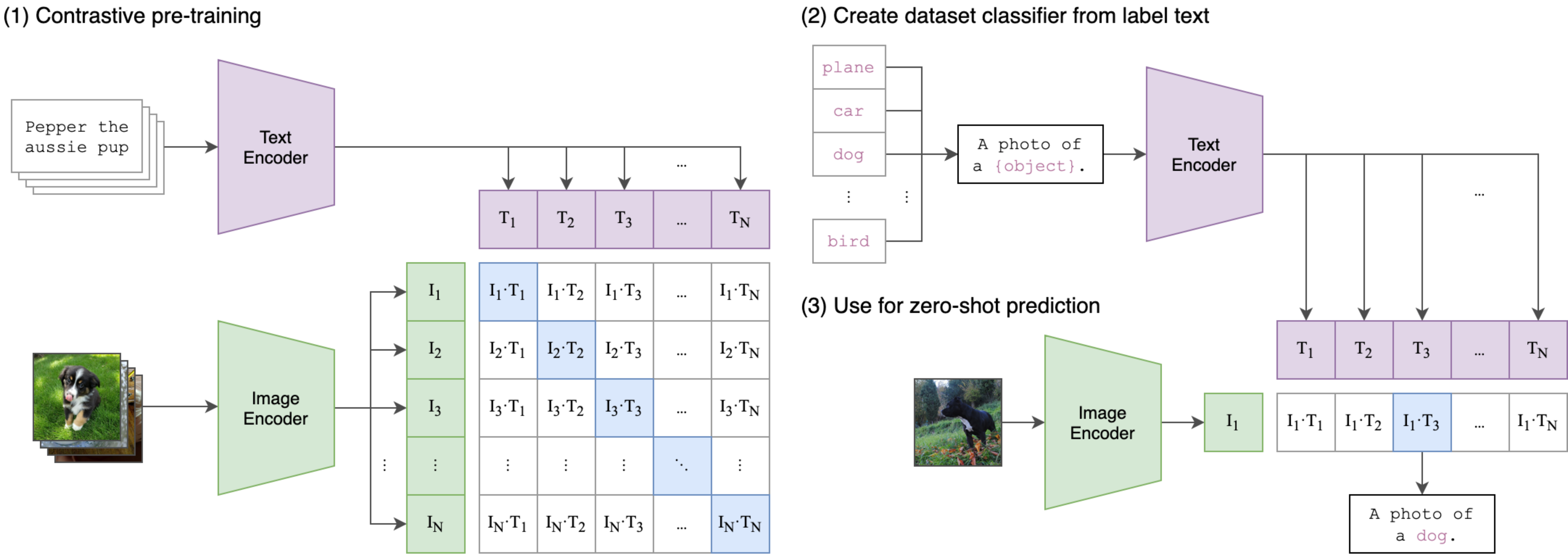
Cherries In A Bowl, Art Painting by Heinz Scholnhammer

image-to-image translation

CLIP and zero-shot Learning

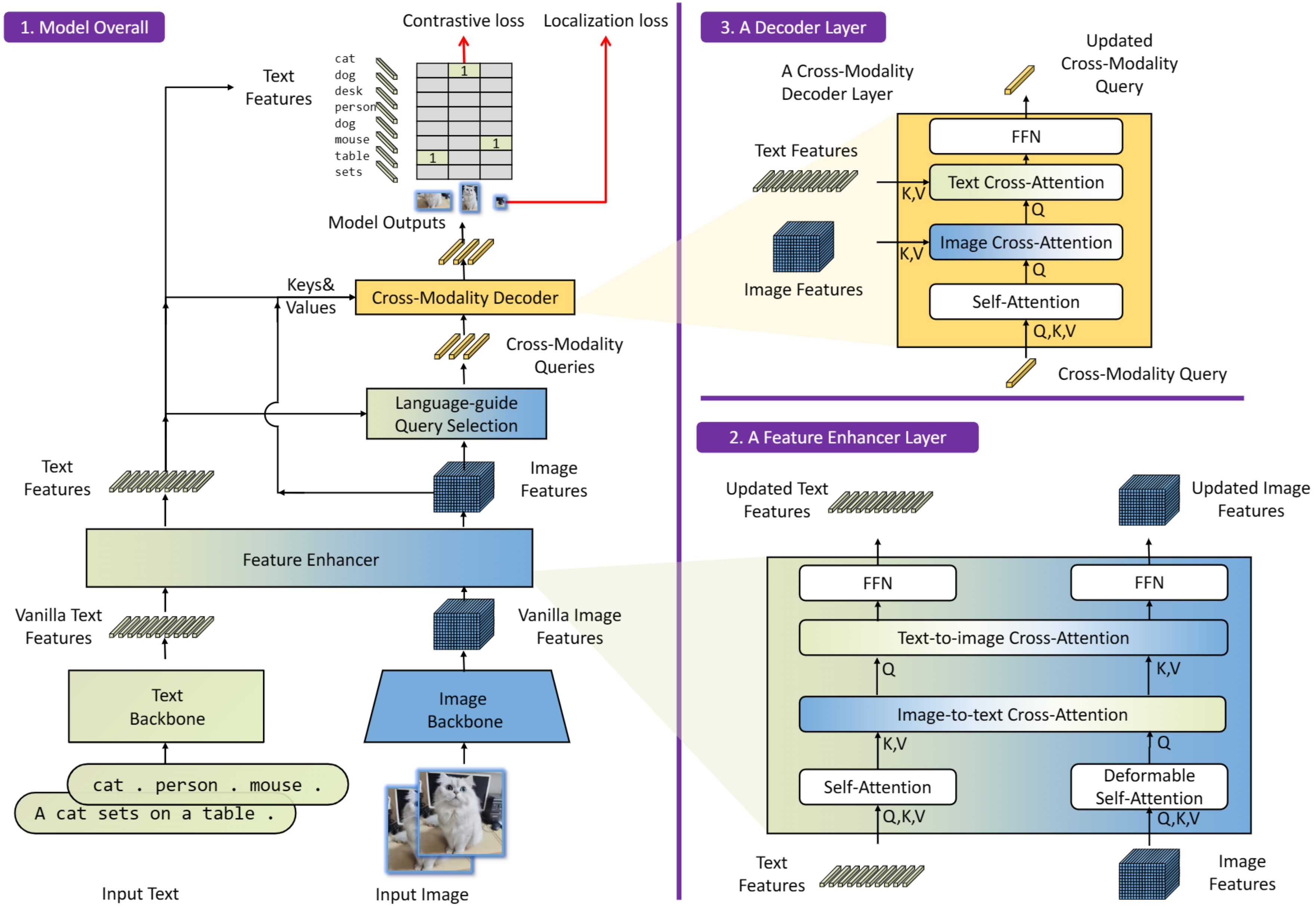
CLIP and zero-shot Learning

CLIP: Learning Transferable Visual Models From Natural Language Supervision (Paper)


(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
T1 T2 T3 ... TN
I1 I2 I3 ⋮ IN
I1·T1 I1·T2 I1·T3 ... I1·TN
I2·T1 I2·T2 I2·T3 ... I2·TN
I3·T1 I3·T2 I3·T3 ... I3·TN
IN·T1 IN·T2 IN·T3 ... IN·TN
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
T1 T2 T3 ... TN
(3) Use for zero-shot prediction
Image Encoder
I1
I1·T1 I1·T2 I1·T3 ... I1·TN
A photo of a dog.

CLIP and zero-shot Learning

GroundingDINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection (Paper)



Hugging Face

Hands-on:

(1) CLIP for zero-shot classification

OpenAI

Hugging Face

Hands-on:

(2) GroundingDINO for zero-shot detection

OpenAI

HSLU Hochschule Luzern

Hugging Face

Hands-on:

(3) Prompt Engineering

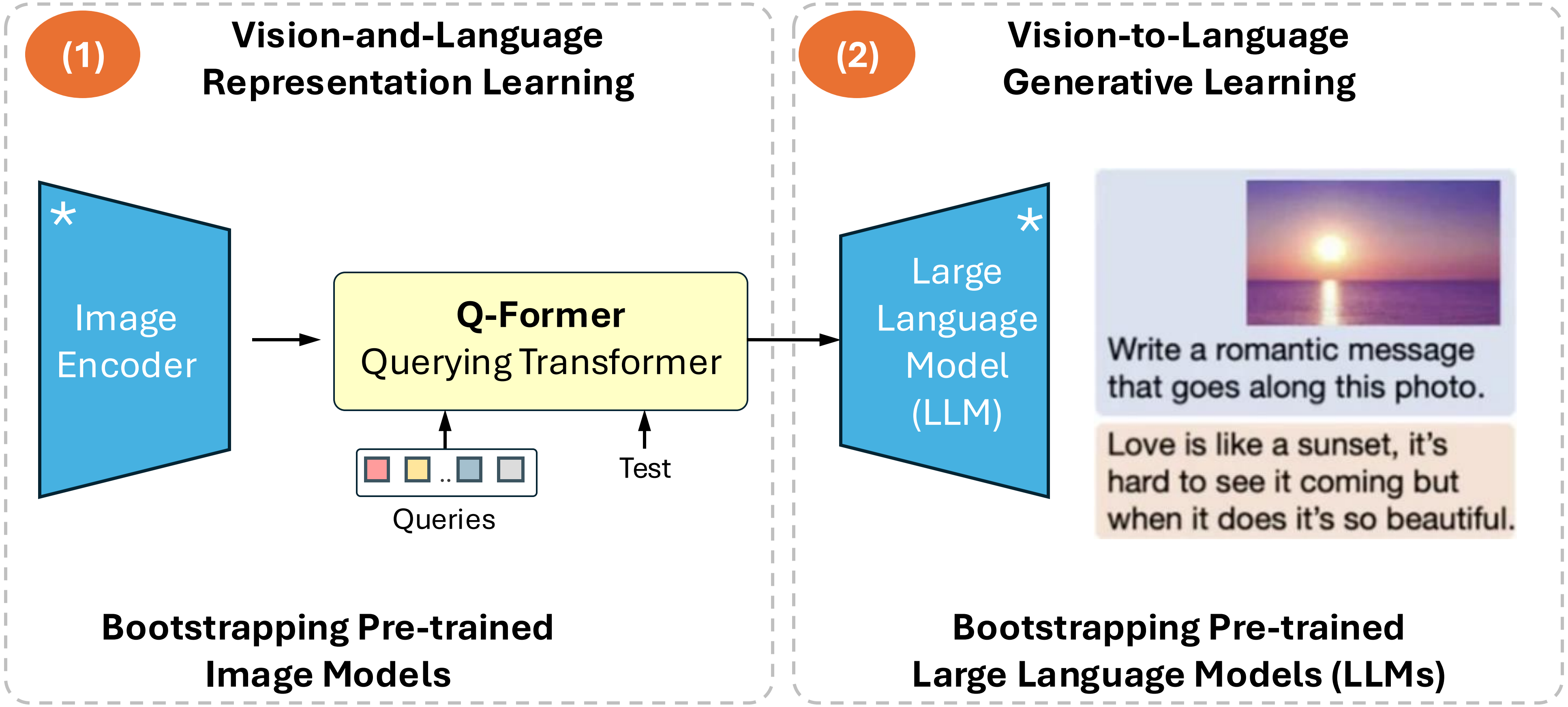
OpenAI

BLIP and multimodal generation

HSLU Hochschule Luzern BLIP and Multimodal generation

BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models (Paper)

Pre-train a light weight Q-Former in two stages:



(1)
Vision-and-Language Representation Learning
*
Image Encoder
Q-Former
Querying Transformer
..
Queries
Test
Bootstrapping Pre-trained Image Models
(2)
Vision-to-Language Generative Learning
*
Large Language Model (LLM)
Bootstrapping Pre-trained Large Language Models (LLMs)

Write a romantic message that goes along this photo.

Love is like a sunset, it's hard to see it coming but when it does it's so beautiful.

HSLU Hochschule Luzern

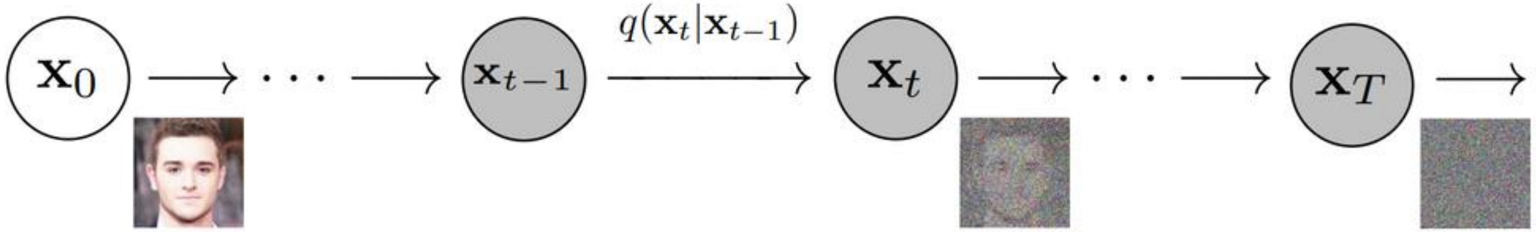
BLIP and Multimodal generation

	Concept	Paper
BLIP	A vision-language pretraining framework that uses a bootstrapped approach to generate and filter captions from noisy web data, enabling unified vision-language understanding and generation tasks like image captioning. It combines a vision encoder and text decoder for efficient captioning.	BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation
IDEFICS	A multimodal model designed for vision and language tasks, including image captioning, built by fine-tuning open-access models like LLaVA and CLIP-ViT. It processes images and text jointly, leveraging a large-scale pretraining dataset for robust performance.	IDEFICS: An Open-Source Visual Language Model
GIT	A generative image-to-text transformer that excels in image captioning by directly generating text from image inputs. It uses a single vision transformer and a text decoder, optimized for lightweight and efficient captioning with performance comparable to BLIP-2.	GIT: A Generative Image-to-text Transformer for Vision and Language
InstructBLIP	An enhanced version of BLIP-2, fine-tuned with instruction-tuning datasets to improve zero-shot and few-shot performance in image captioning and other vision-language tasks. It leverages BLIP-2's architecture with additional task-specific optimizations.	InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning
COCA	A contrastive captioning model that combines contrastive learning with generative captioning. It uses image-text alignment to produce high-quality captions, offering performance close to BLIP-2 with lower computational requirements.	CoCa: Contrastive Captioners are Image-Text Foundation Models

Diffusion Models

Diffusion Models

Diffusion Models work by destroying training data through the successive addition of Gaussian noise, and then learning to recover the data by reversing this noising process.



Forward diffusion process

$$\boxed{q(x_t|x_{t-1})} = \mathcal{N}(\boxed{x_t}; \boxed{\sqrt{1-\beta_t}x_{t-1}}, \boxed{\beta_t\mathbf{I}})$$

distribution of the noised images

Output

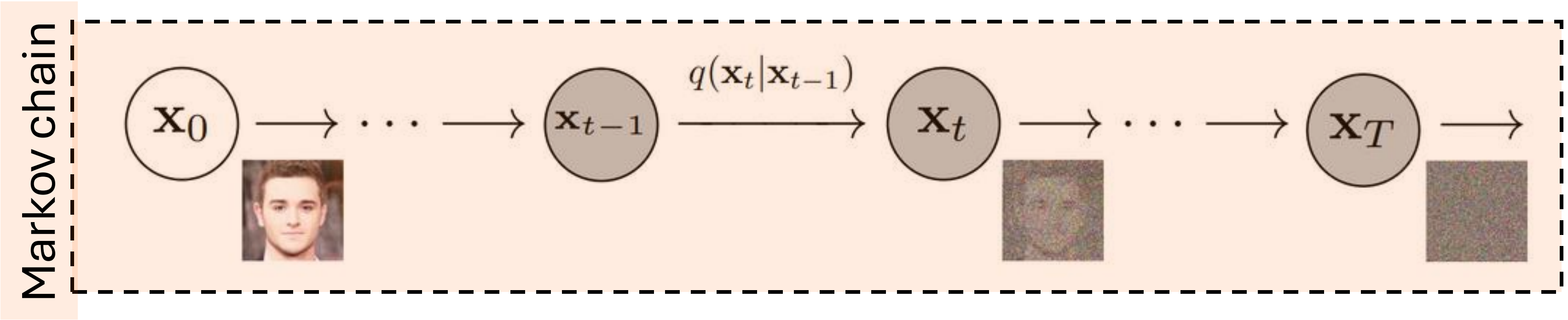
Mean μ_t

Variance Σ_t

- ***t***: This represents the **time step**. It goes from a clear image at t=0 to a completely noisy image at t=T.
- ***x_0***: This is your **original, clean data sample**. It's the starting point of the whole process.
- ***β_t***: This is the **variance schedule**. It's a list of small numbers that control how much noise is added at each step. (1) It's a fixed sequence (not learned). (2) it typically starts with a very small number ($\beta0\approx0$) so the first few steps add very little noise. (3) It gradually increases to a larger number ($\beta_t\approx1$), so the later steps add a lot of noise. (4) The values of βt are always between 0 and 1.
- **I**: This is the **identity matrix**. It ensures that the noise is added independently to each pixel of the image, without any correlation between pixels.

Diffusion Models

Diffusion Models work by destroying training data through the successive addition of Gaussian noise, and then learning to recover the data by reversing this noising process.



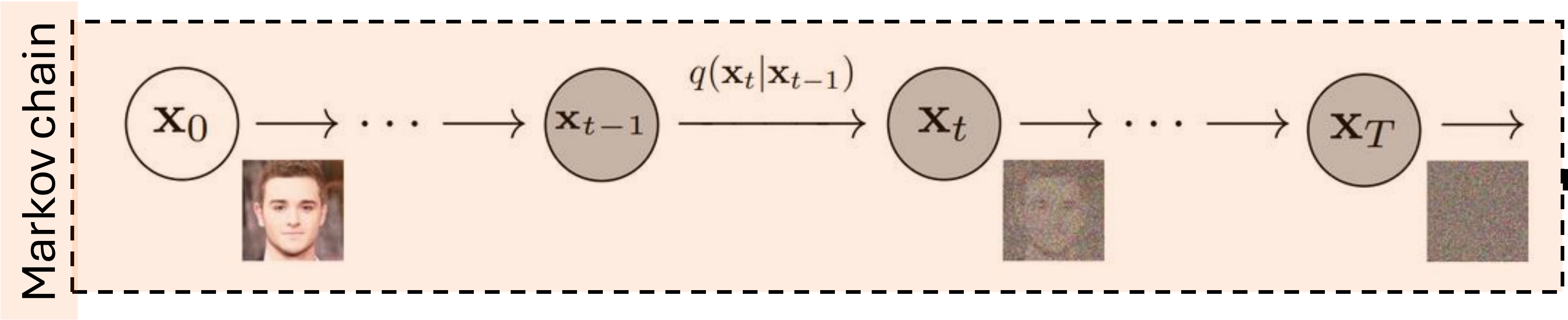
Ultimately, the image is asymptotically transformed to pure Gaussian noise.

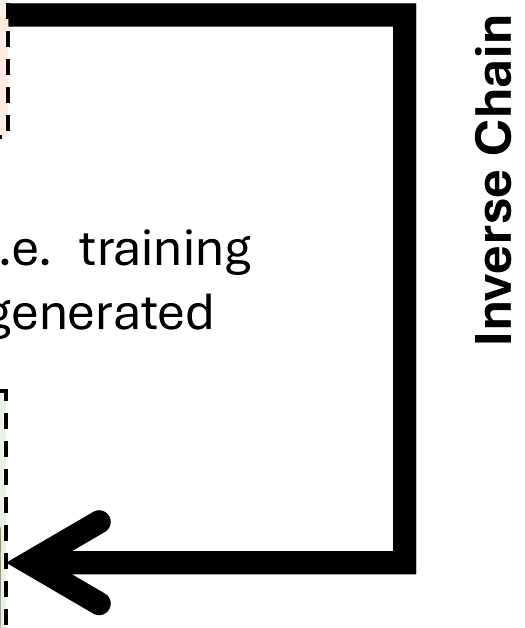
The goal of training these models is to learn the reverse process (i.e. training $q_0(x_{t-1}|x_t)$). By traversing backwards along this chain, new data can be generated

Diffusion Models:A Comprehensive Survey of Methods and Applications. LING YANG *et. al. Feb. 2024* *arXiv:2209.00796*

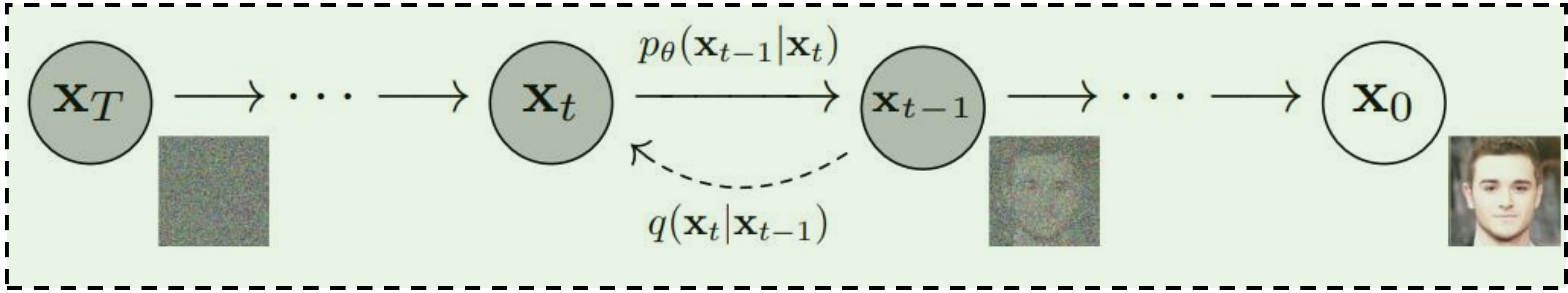
Diffusion Models

Diffusion Models work by destroying training data through the successive addition of Gaussian noise, and then learning to recover the data by reversing this noising process.





The goal of training these models is to learn the reverse process (i.e. training $q_0(x_{t-1}|x_t)$). By traversing backwards along this chain, new data can be generated


$\mathbf{x}_T$
$\mathbf{x}_t$
$p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t)$
$\mathbf{x}_{t-1}$
$q(\mathbf{x}_t|\mathbf{x}_{t-1})$
$\mathbf{x}_0$

Diffusion Models:A Comprehensive Survey of Methods and Applications. LING YANG *et. al. Feb. 2024* arXiv:2209.00796

Hugging Face

Hands-on:

(1) BLIP image captioning

OpenAI

Hugging Face

Hands-on:

(2) Stable diffusion

OpenAI

Computer Vision – Fundamentals

Introduction To Tranformers In Vision

Program directors:

Dr. Prof.
Umberto Michelucci

Dr.
Aygul Zagidullina

Dr.
Safouane El Ghazouali

HSLU Lucerne University
of Applied Sciences
and Arts

BOOTCAMP AI-VISION

Starting date: **25 August 2026**

DAY 1

- …

DAY 2

- …

DAY 5

Projects realisation

DAY 3

- Introduction to transformers in vision
- Exploring ViT architecture
- Other vision tranformers alternatives
- Transformers in object detection
- Transformers library for ViT application

DAY 4

- Use of transformers if vision generative AI
- Fields of applications
- Zero-shot learning for classification and detection tasks
- Multimodal AI models
- Image Diffusion Models