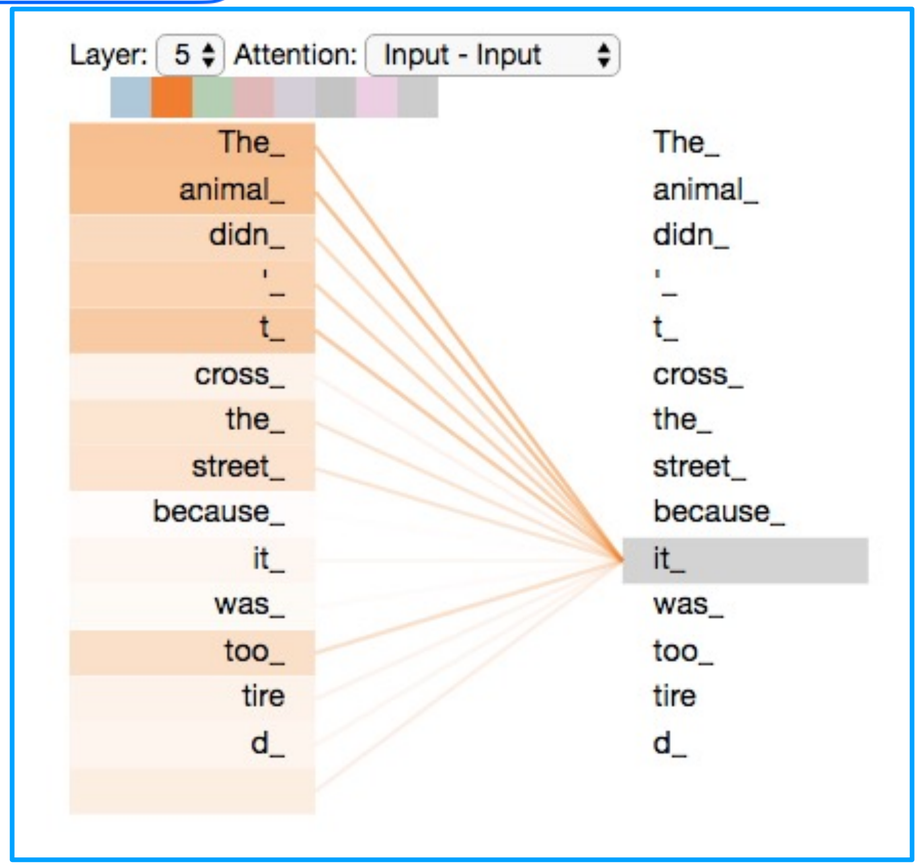
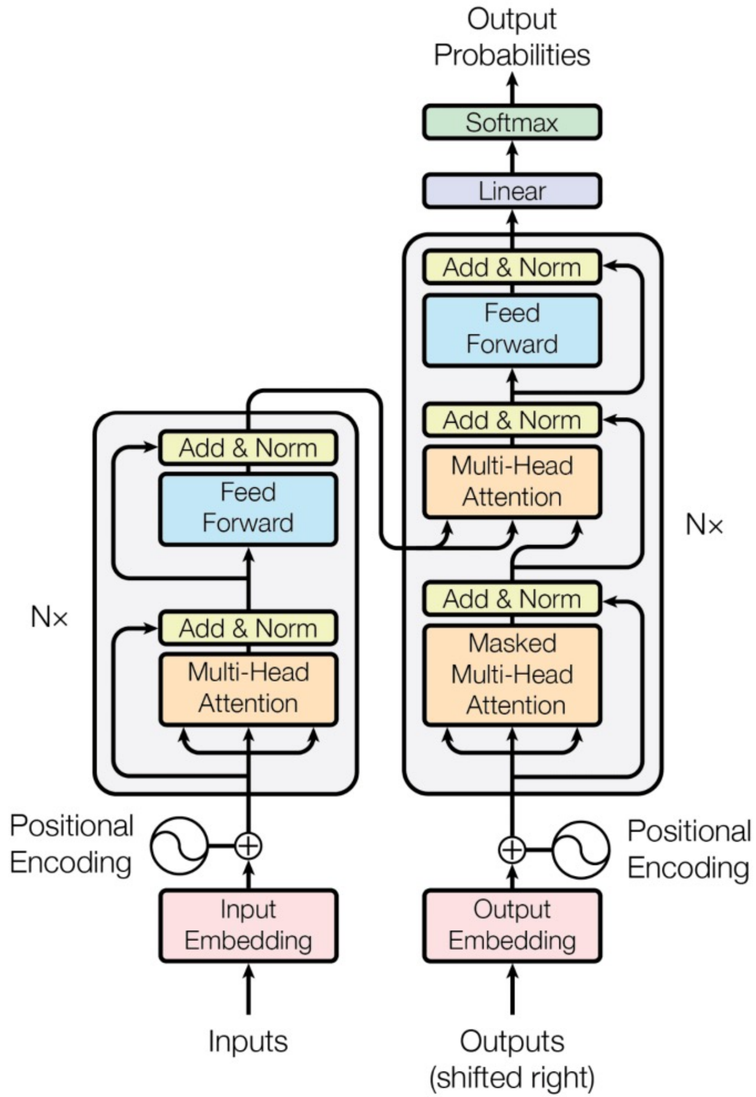
DeepRob

Lecture 16
Language Models
University of Michigan | Department of Robotics

https://jalammar.github.io/illustrated-transformer/

Transformers (review)





Transformers Mid-2017

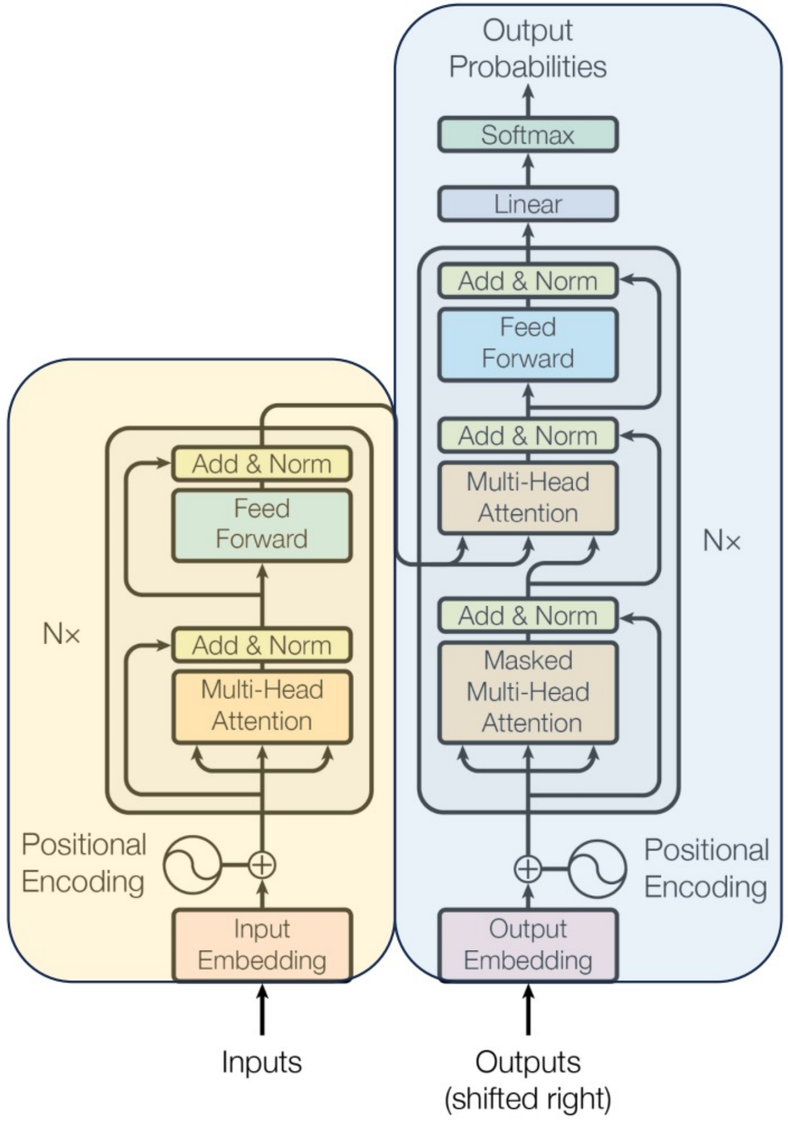
Input – input tokens
Output – hidden states

Model can see all timesteps

Does not usually output tokens, so no inherent auto-regressivity

Can also be adapted to generate tokens *by appending a module that maps hidden state dimensionality to vocab size*

Representation



Input – output tokens and hidden states*
Output – output tokens

Model can only see previous timesteps

Model is auto-regressive with previous timesteps' outputs

Can also be adapted to generate hidden states *by looking before token outputs*

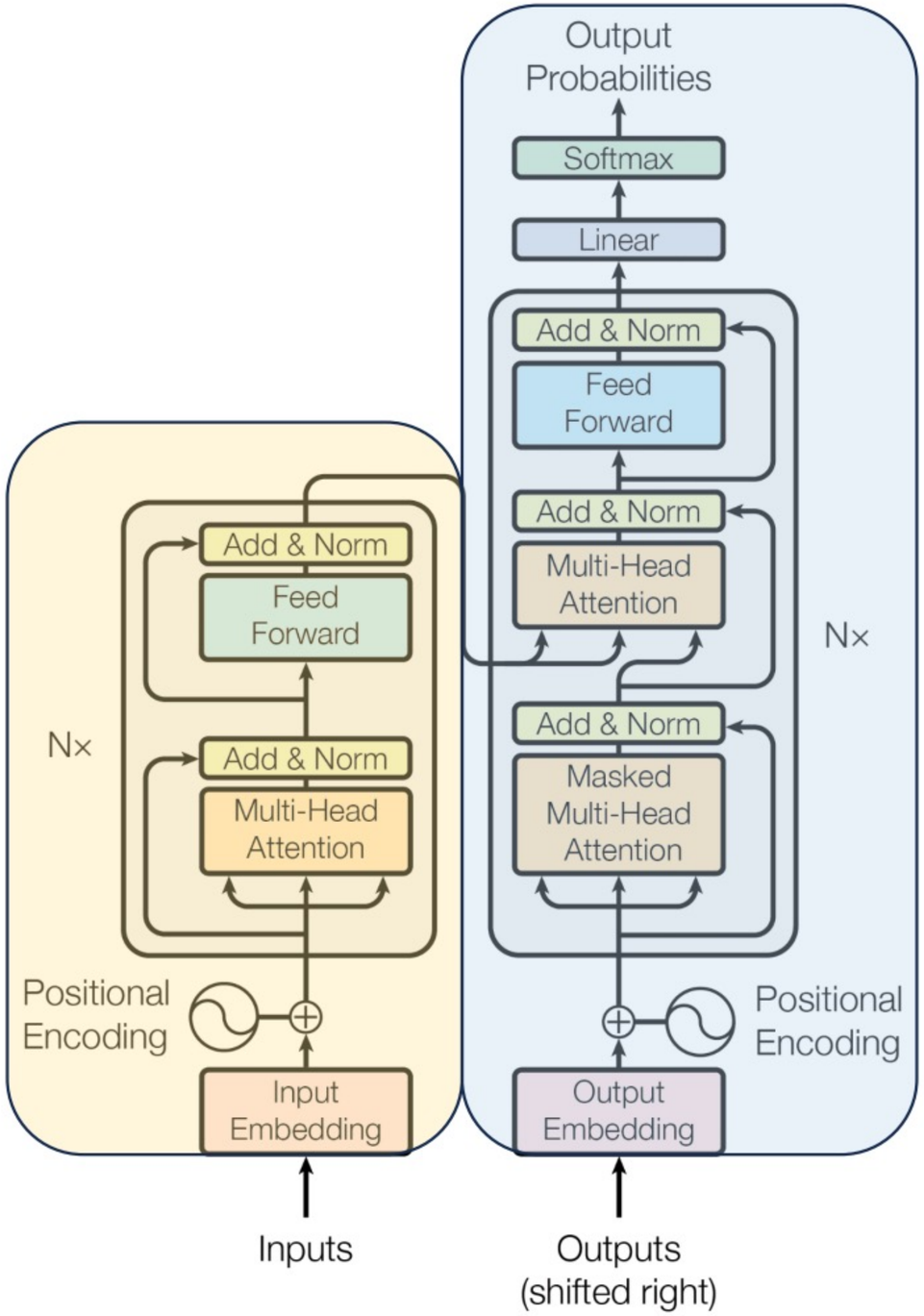
Generation

https://deeplearning.cs.cmu.edu/F23/document/slides/lec19.transformersLLMs.pdf

2018-
LLM Era

BERT
Oct 2018

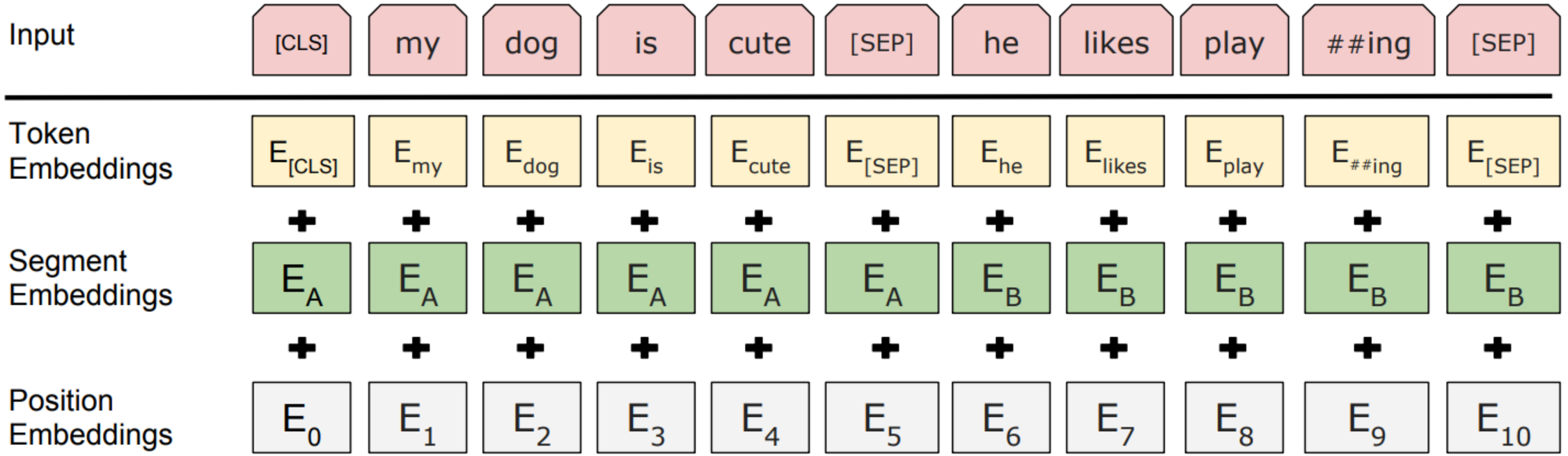
Representation



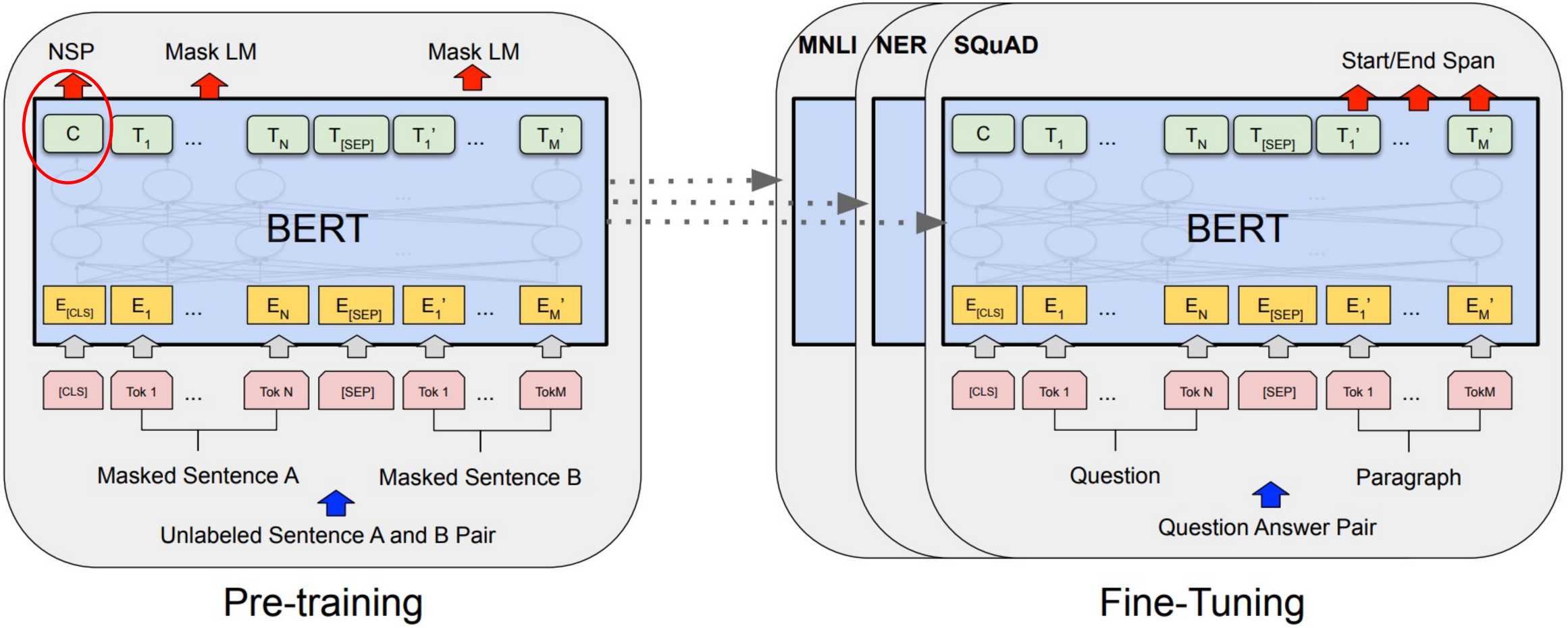
GPT
Jun 2018

Generation

BERT Input representations



BERT: Bidirectional Encoder Representations from Transformers



BLEU metric $\in [0,1]$

- Bilingual Evaluation Understudy
- https://huggingface.co/spaces/evaluate-metric/bleu

- R (reference): human expert
- C (candidate): produced by translation system (e.g., a Transformer)

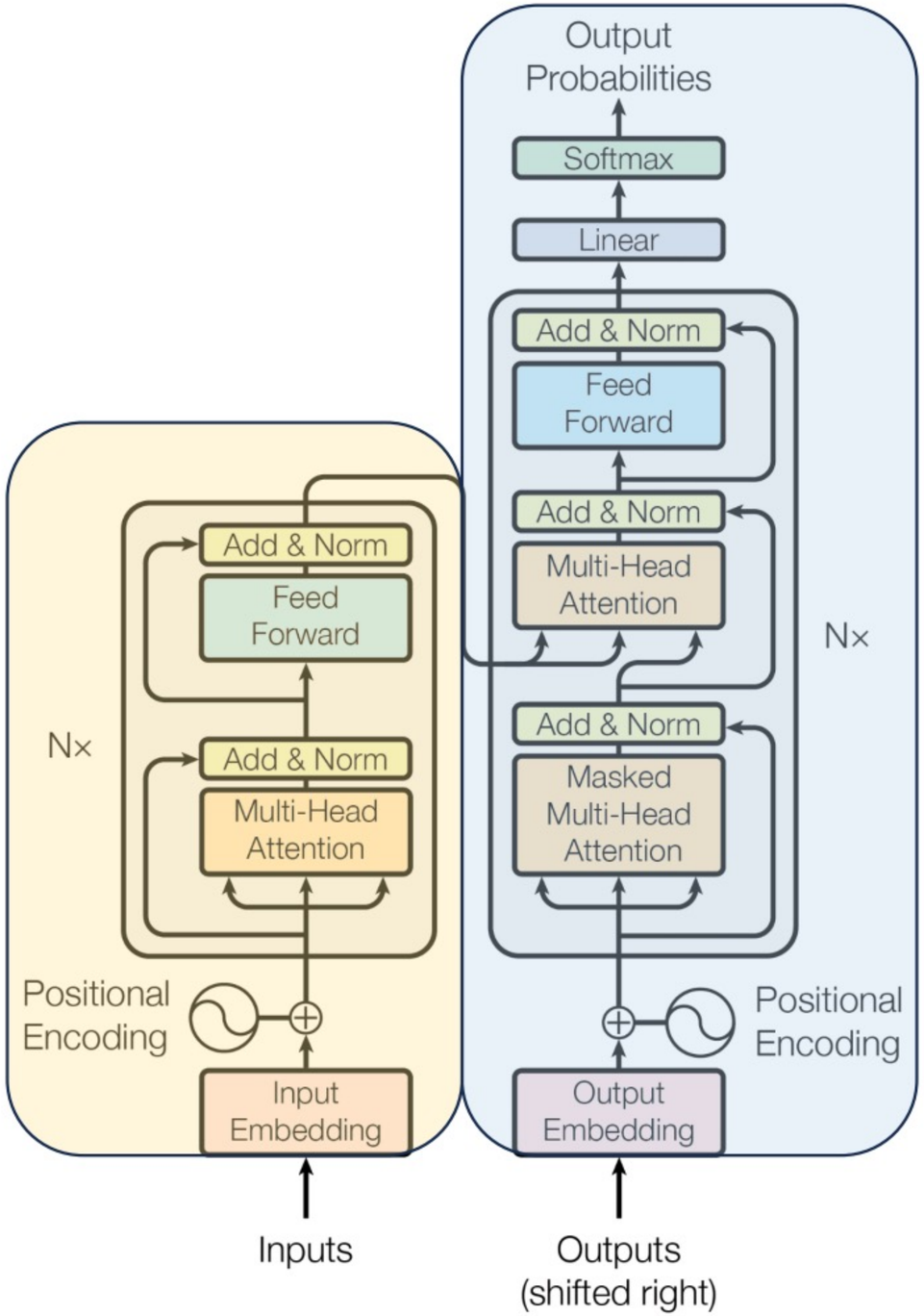
2018-
LLM Era

BERT
Oct 2018

Corpus

- English Wikipedia - 2,500 million words
- Book Corpus - 800 million words

Representation



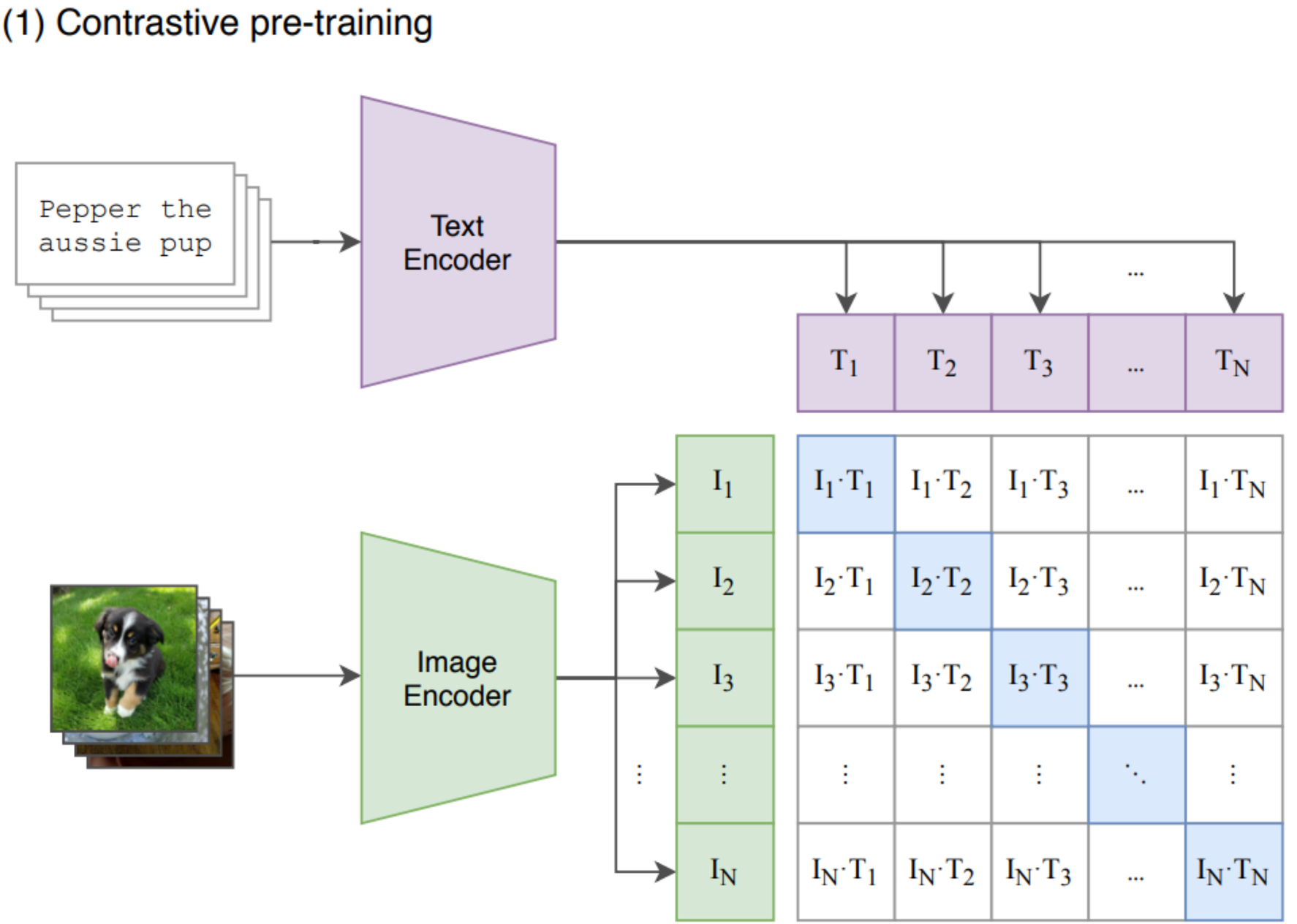
GPT
Jun 2018

Generation

CLIP

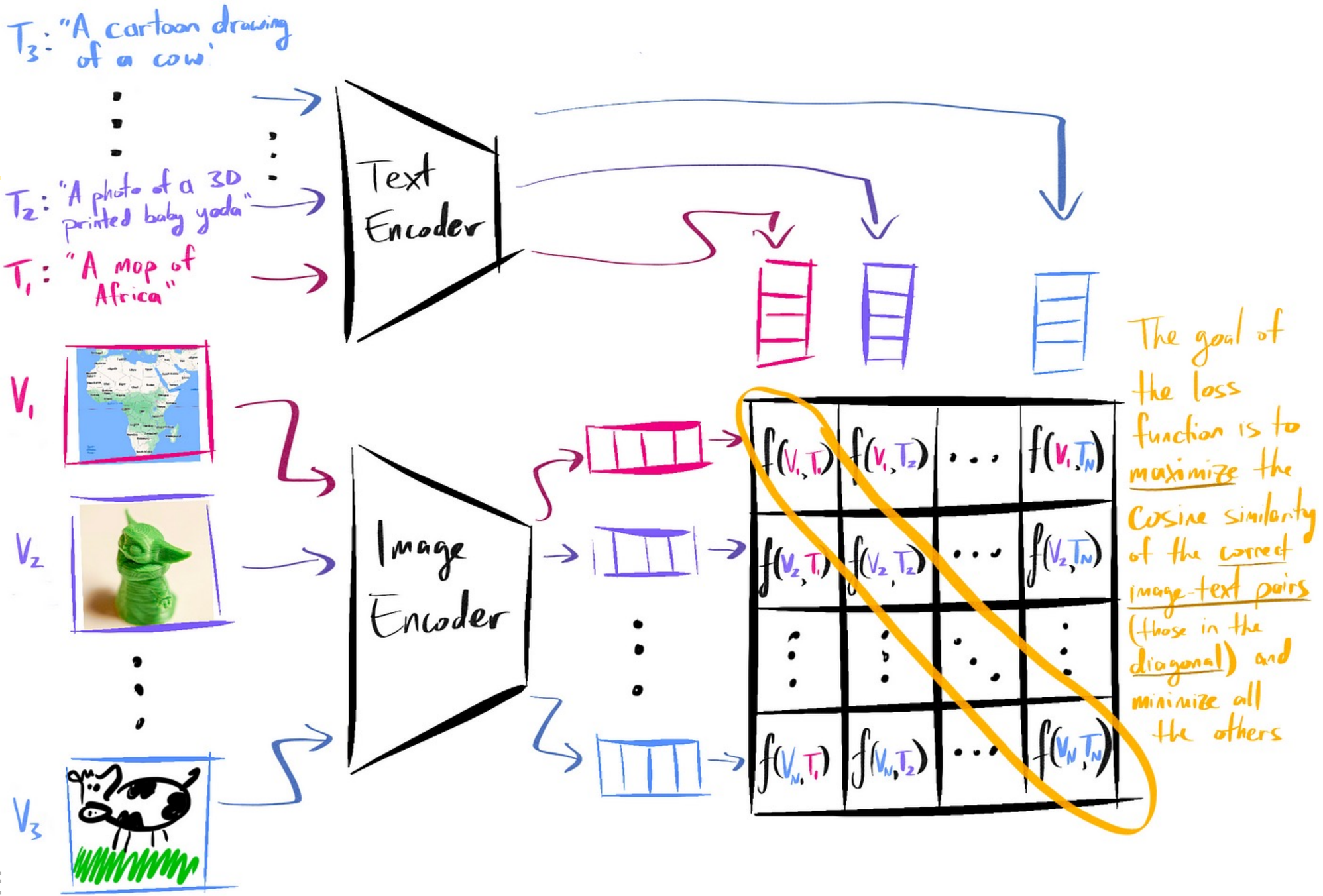
- CLIP (*Contrastive Language–Image Pre-training*)
- learning visual representations from natural language supervision

CLIP


(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$

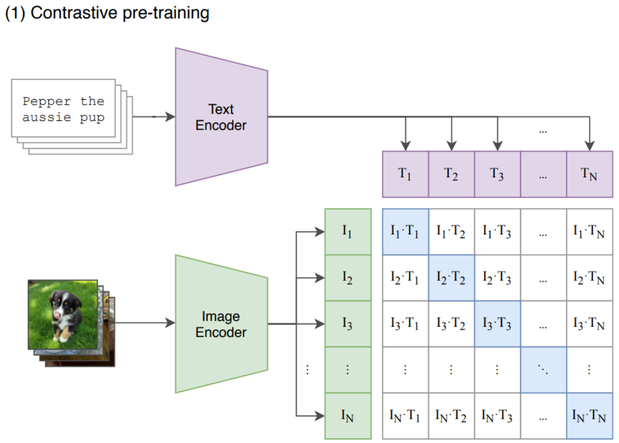
CLIP

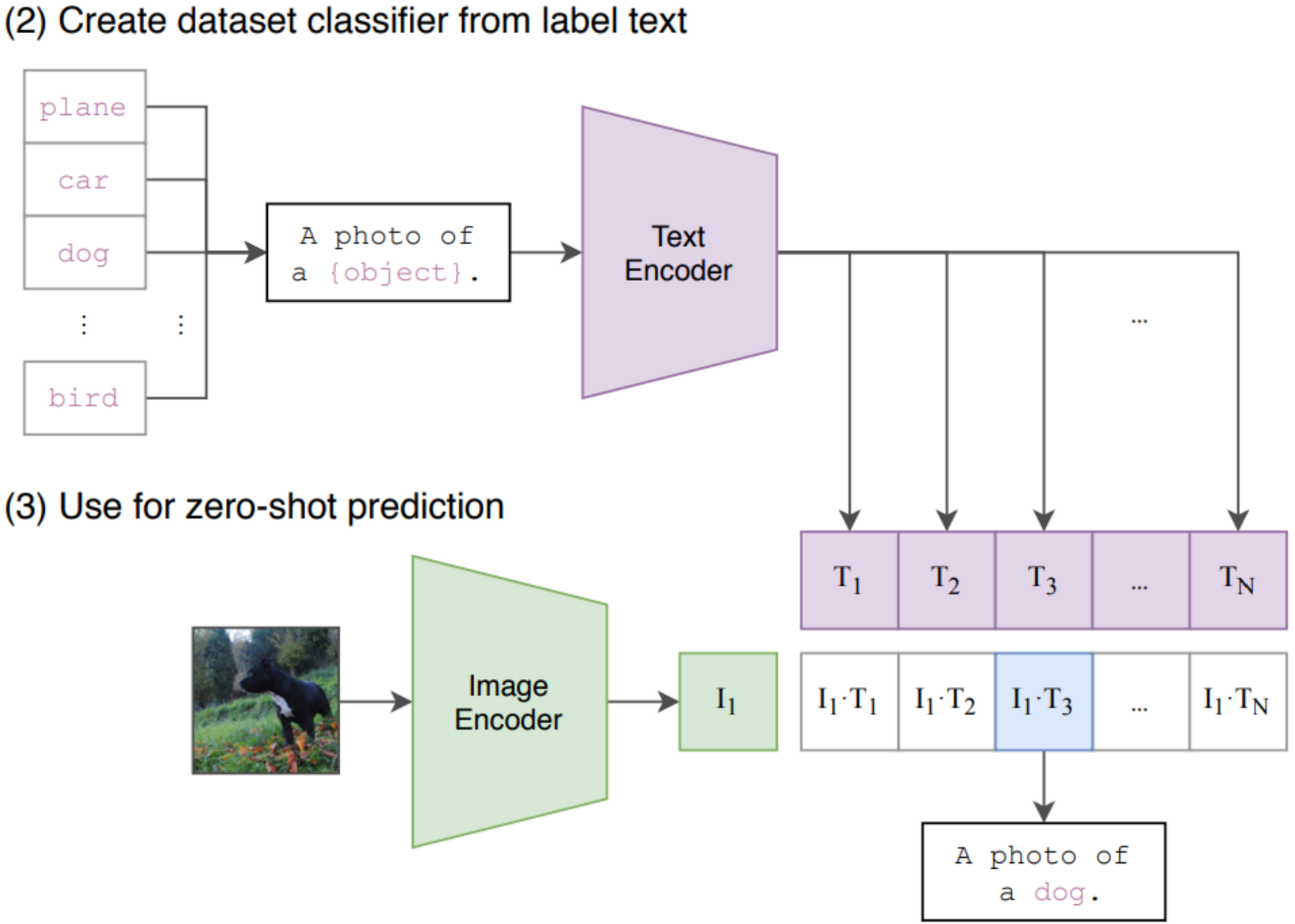
(example)



https://medium.com/one-minute-machine-learning/clip-paper-explained-easily-in-3-levels-of-detail-61959814ad13

CLIP


(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
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 $I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$


(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
T_1 T_2 T_3 ... T_N
(3) Use for zero-shot prediction
Image Encoder
I_1
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
A photo of a dog.

CLIP

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

Figure 3. Numpy-like pseudocode for the core of an implementation of CLIP.

CLIP

- learning visual representations from natural language supervision
- Pre-trained model, NOT a generative model
- Advantage:
  - does not need task-specific training data
  - bridging two modalities
- Limitations:
  - abstract or systematic tasks, complex tasks (e.g., predicting “nearest”, counting)
  - poor generalization on images not covered in pre-training

LERF

- LERF: Language Embedded Radiance Field
- *view-independent*

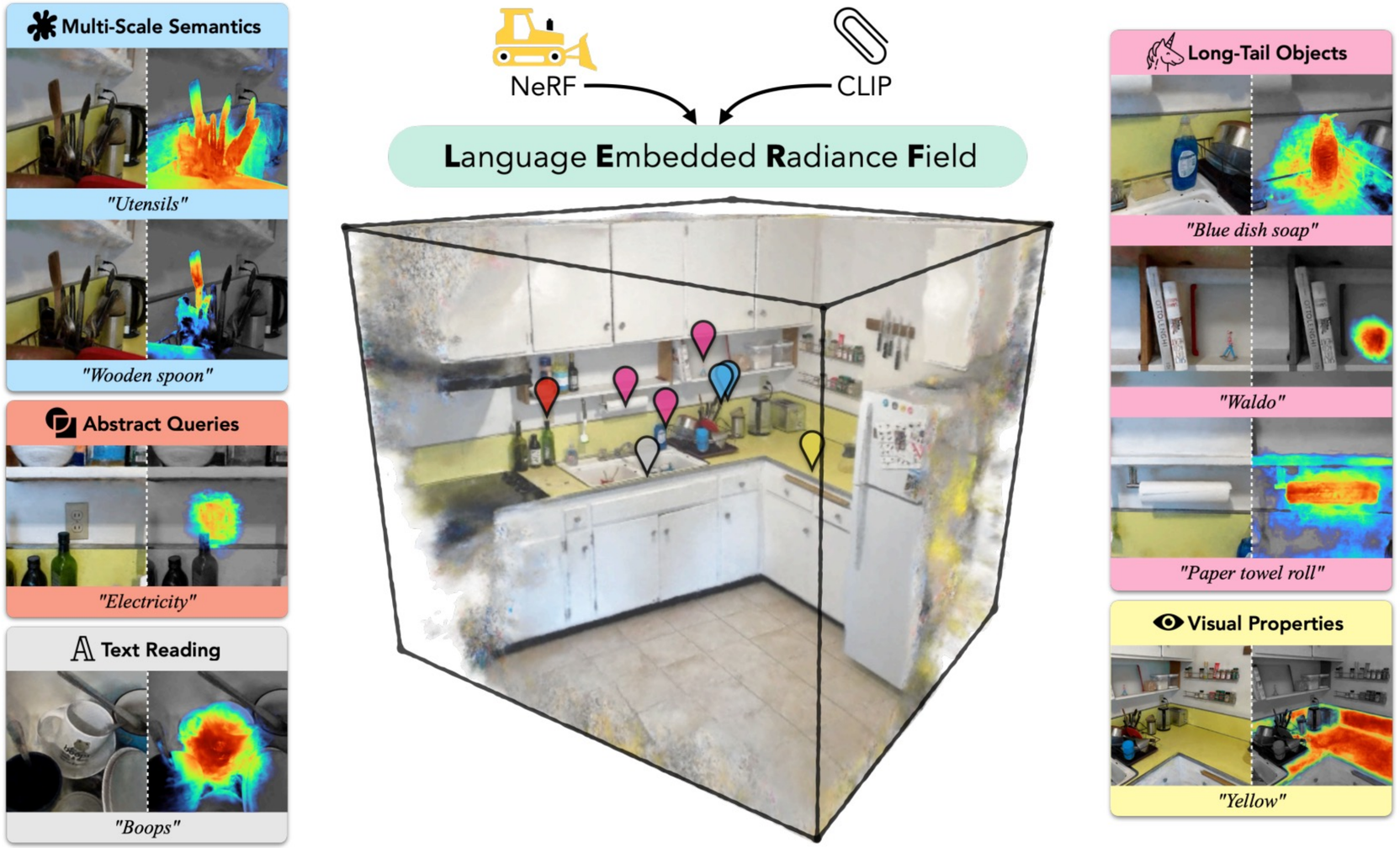
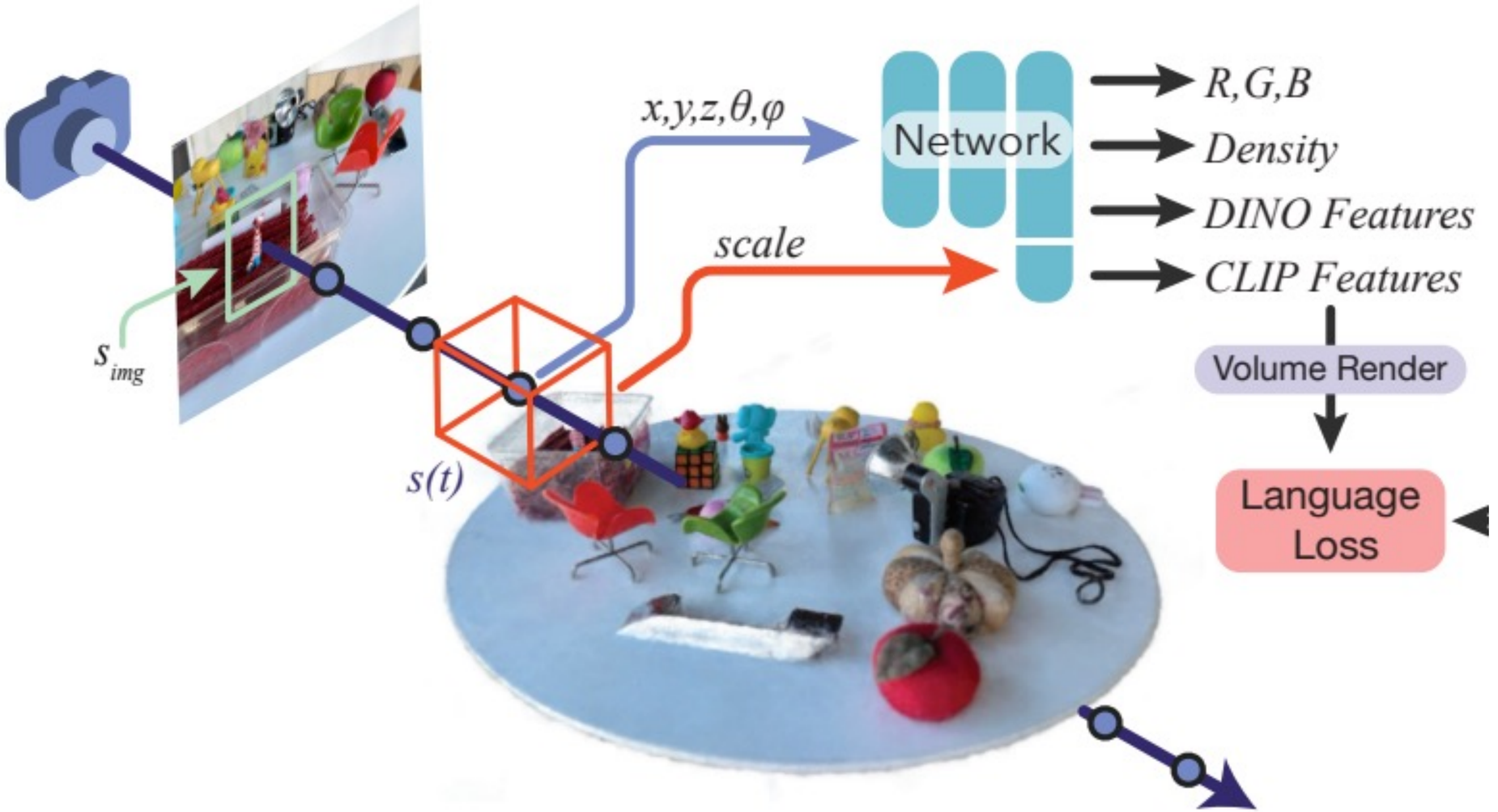
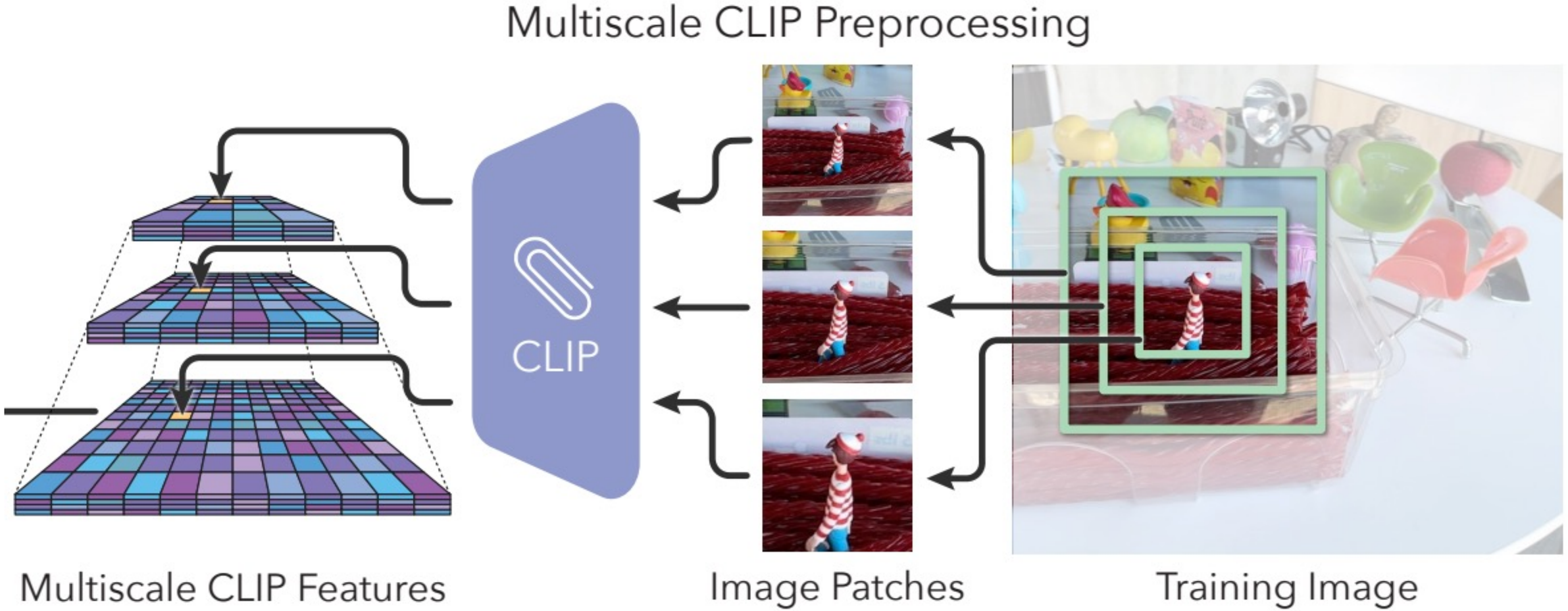


Figure 1: **Language Embedded Radiance Fields (LERF).** LERF grounds CLIP representations in a dense, multi-scale 3D field. A LERF can be reconstructed from a hand-held phone capture within 45 minutes, then can render dense relevancy maps given textual queries interactively in real-time. LERF enables a broad range of concepts to be queried via natural language, from abstract queries like *"Electricity"*, visual properties like *"Yellow"*, long-tail objects such as *"Waldo"*, and even reading text like *"Boops"* on the mug. For each prompt, an RGB image and relevancy map are rendered focusing on the location with maximum relevancy activation.

LERF Rendering




Multiscale CLIP Preprocessing
CLIP
Multiscale CLIP Features
Image Patches
Training Image

Putting it together

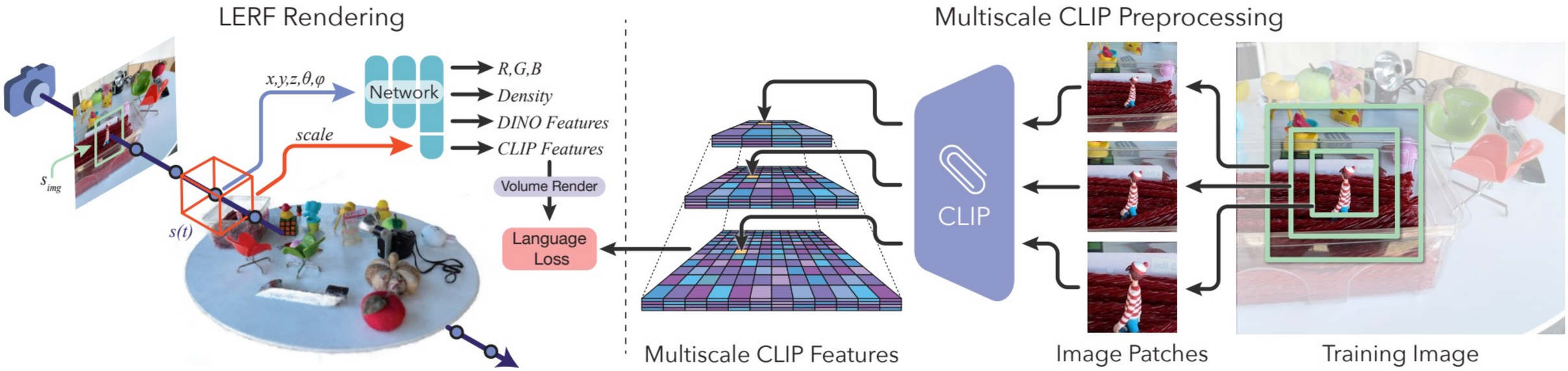


Figure 2: **LERF Optimization:** *Left*: LERF represents a field of 3D volumes, parameterized by position x, y, z and scale s (orange cube). To render a CLIP embedding along a ray, the field is sampled and averaged according to NeRF's volume rendering weights. Physical scale corresponds to an image scale s_{img} via projective geometry. *Right*: We pre-compute a multi-scale feature pyramid of CLIP embeddings over training views, and during training interpolate this pyramid with s_{img} and the ray's pixel location to obtain CLIP supervision. The CLIP loss maximizes cosine similarity, and other outputs are supervised with mean squared-error using standard per-pixel rendering.

DeepRob

Lecture 16
Language Models
University of Michigan | Department of Robotics