Lecture 2 Transformers

CSC485/2501

Announcements

- A1 release today (later in the afternoon).
- Essay 1 released – Due Sep 16.
- A1 Tutorials:
  - T2: Fri, Sep 20.
  - T3: Wed, Sep 25.

Language Model

Models that estimate:

Probability(*Some sentence over here.*)

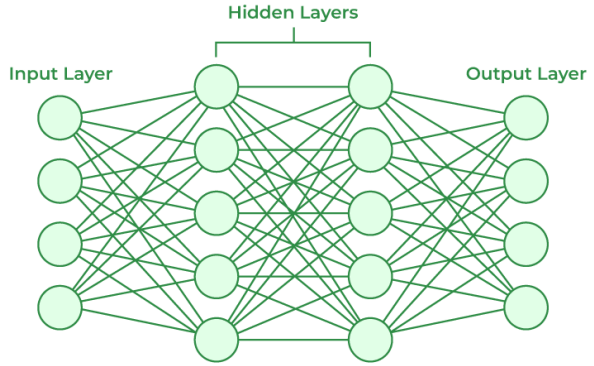
Neural Models

Dog!

Image: 863 x 625 pixels

```
tensor([[0.7913, 0.4032, 0.4254,  ..., 0.4260, 0.2358, 0.4319],
        [0.5912, 0.8657, 0.9194,  ..., 0.1741, 0.1045, 0.0873],
        [0.7583, 0.4037, 0.6144,  ..., 0.8484, 0.5134, 0.0949],
        ...,
        [0.5109, 0.5798, 0.1756,  ..., 0.9258, 0.2265, 0.8081],
        [0.0229, 0.0628, 0.7615,  ..., 0.5301, 0.6956, 0.0535],
        [0.6283, 0.3913, 0.7945,  ..., 0.3634, 0.2575, 0.0871]])
```

Vectorize

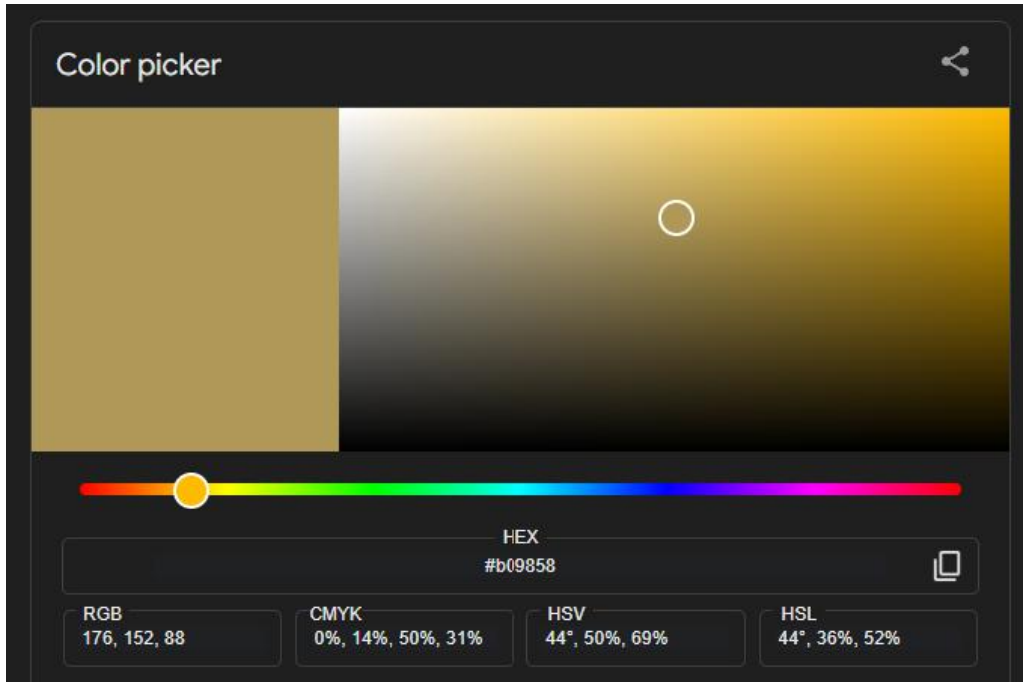


Predict

Image vs. Natural Language

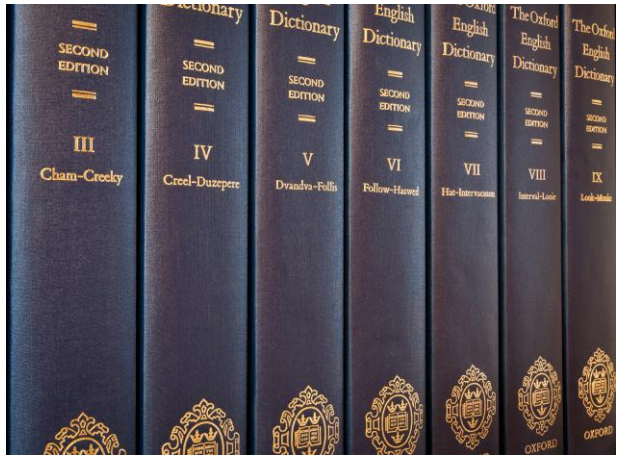
Vision (Image)

- Fixed input size (e.g., 960x720 pixels).
- Defined and continuous input range (RGB).

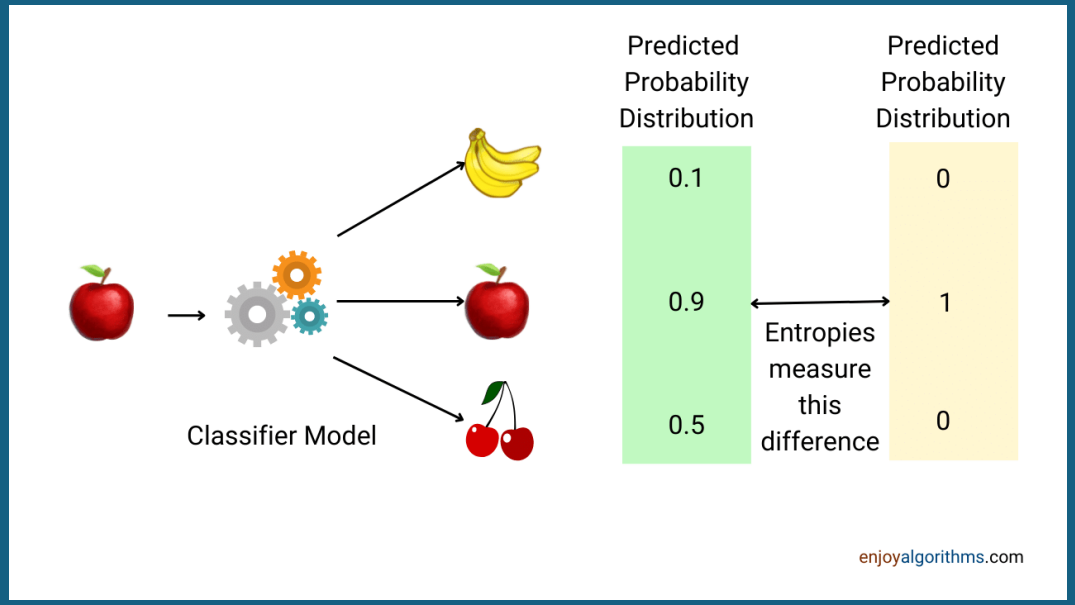


Natural Language

- Dynamic input sequence length.
- Discrete and large input space.
  - GPT-2 tokenizer: 50257 tokens.
  - LLaMA-3.1 tokenizer: 128000 tokens.



Neural Networks

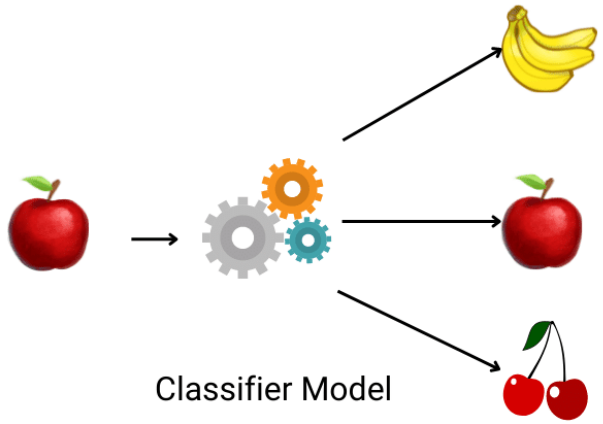


Probability(*Some sentence over here.*)?

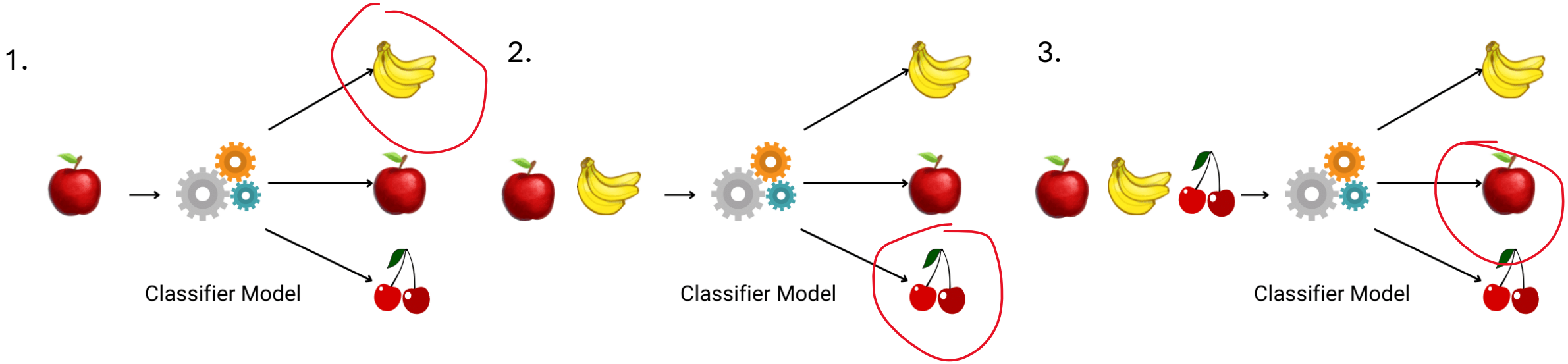
Training Task 1: Next Token Prediction

The capital city of Canada is

- Toronto
- the
- consultant
- Ottawa
- Frank
- ...



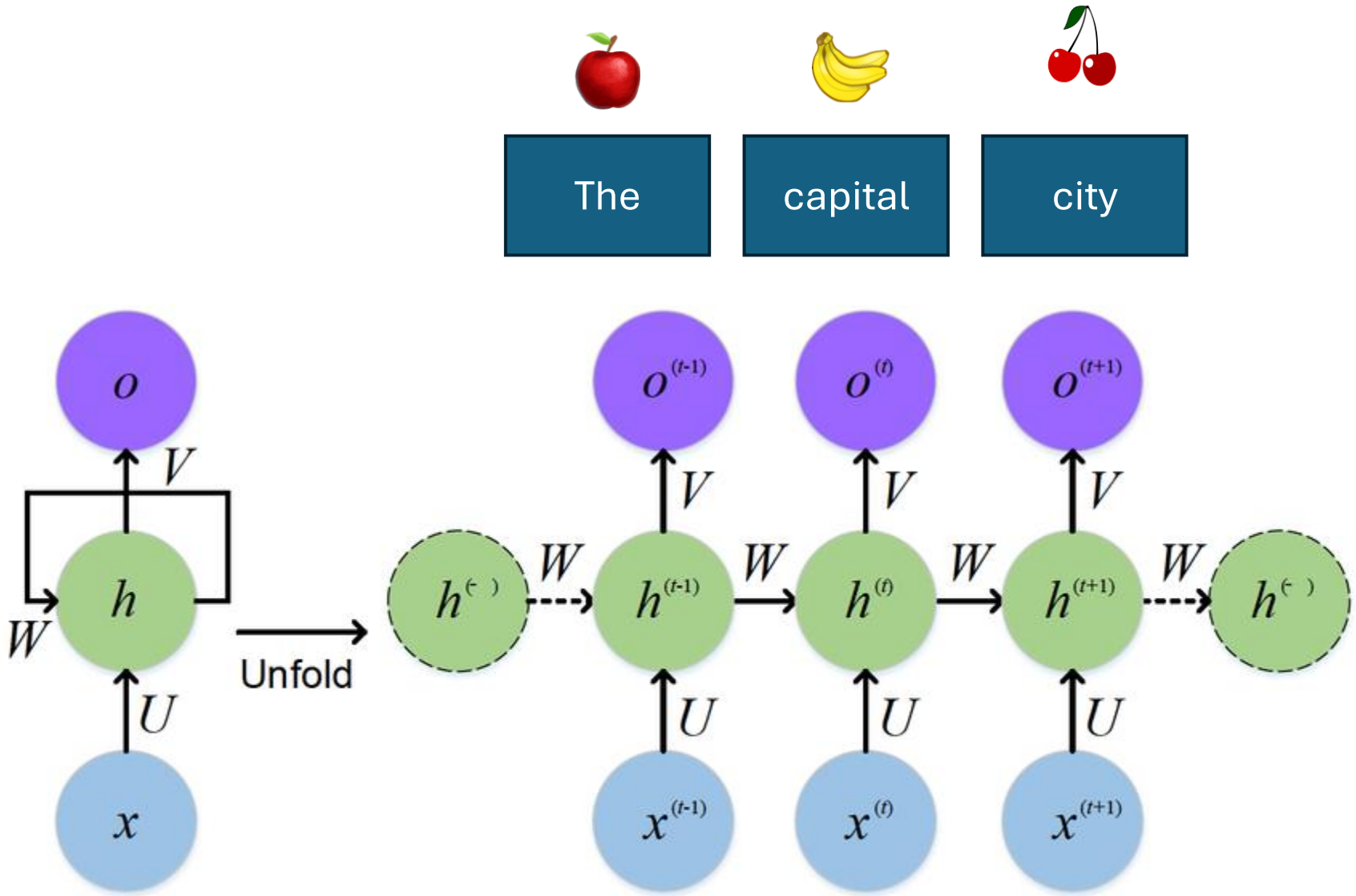
Probability(🍎 🍌 🍒 🍎)



Probability(🍎 🍌 🍒 🍎)

=P(🍎)*P(🍌 | 🍎)*P(🍒 | 🍎 🍌)*P(🍎 | 🍎 🍌 🍒)

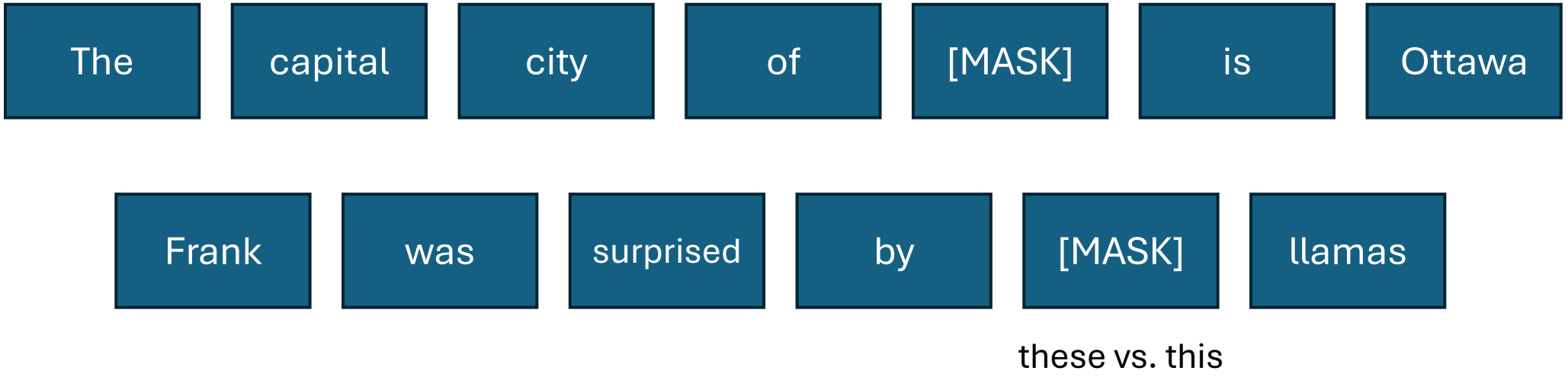
RNN


The
capital
city
o
V
h
W
U
x
Unfold
$o^{(t-1)}$
$o^{(t)}$
$o^{(t+1)}$
$h^{(t-1)}$
$h^{(t)}$
$h^{(t+1)}$
$x^{(t-1)}$
$x^{(t)}$
$x^{(t+1)}$

Training Task 1: Next Token Prediction

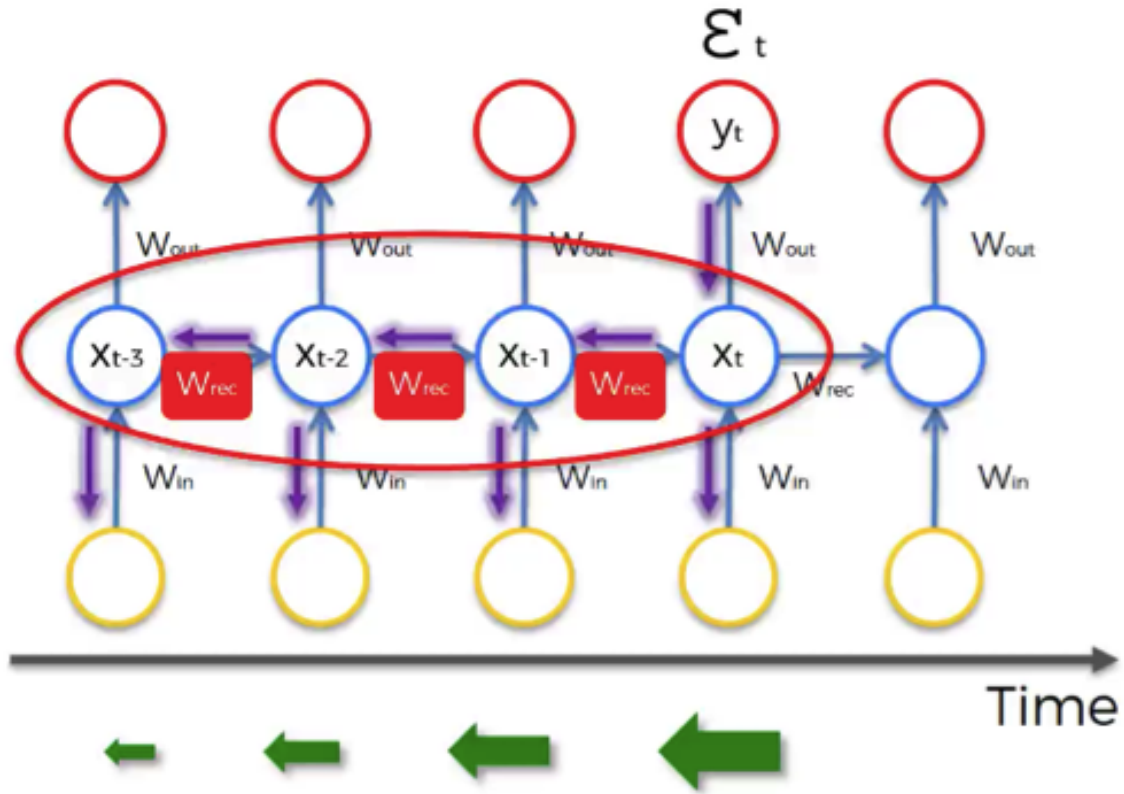
The	capital	city	of	Canada	is	Ottawa

Training Task 2: Masked Language Modelling



- MASK 15% of the tokens.

The Vanishing Gradient Problem



$$\frac{\partial \mathcal{E}}{\partial \theta} = \sum_{1 \le t \le T} \frac{\partial \mathcal{E}_t}{\partial \theta} \quad (3)$$

$$\frac{\partial \mathcal{E}_t}{\partial \theta} = \sum_{1 \le k \le t} \left(\frac{\partial \mathcal{E}_t}{\partial \mathbf{x}_t} \frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} \frac{\partial^+ \mathbf{x}_k}{\partial \theta} \right) \quad (4)$$

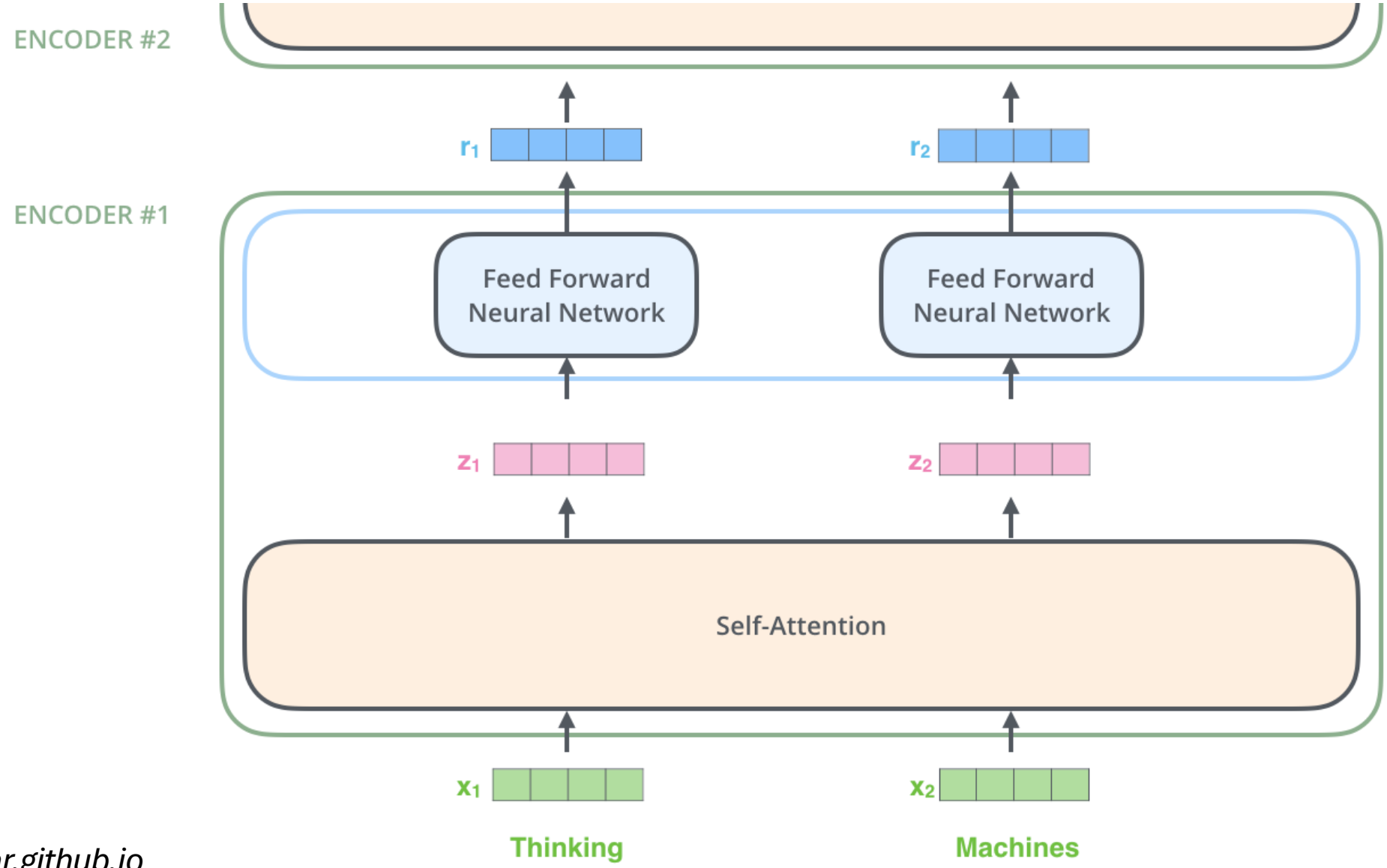
$$\frac{\partial \mathbf{x}_t}{\partial \mathbf{x}_k} = \prod_{t \ge i > k} \frac{\partial \mathbf{x}_i}{\partial \mathbf{x}_{i-1}} = \prod_{t \ge i > k} \mathbf{W}_{rec}^T diag(\sigma'(\mathbf{x}_{i-1})) \quad (5)$$

W_{rec} ~ small ⇨ Vanishing

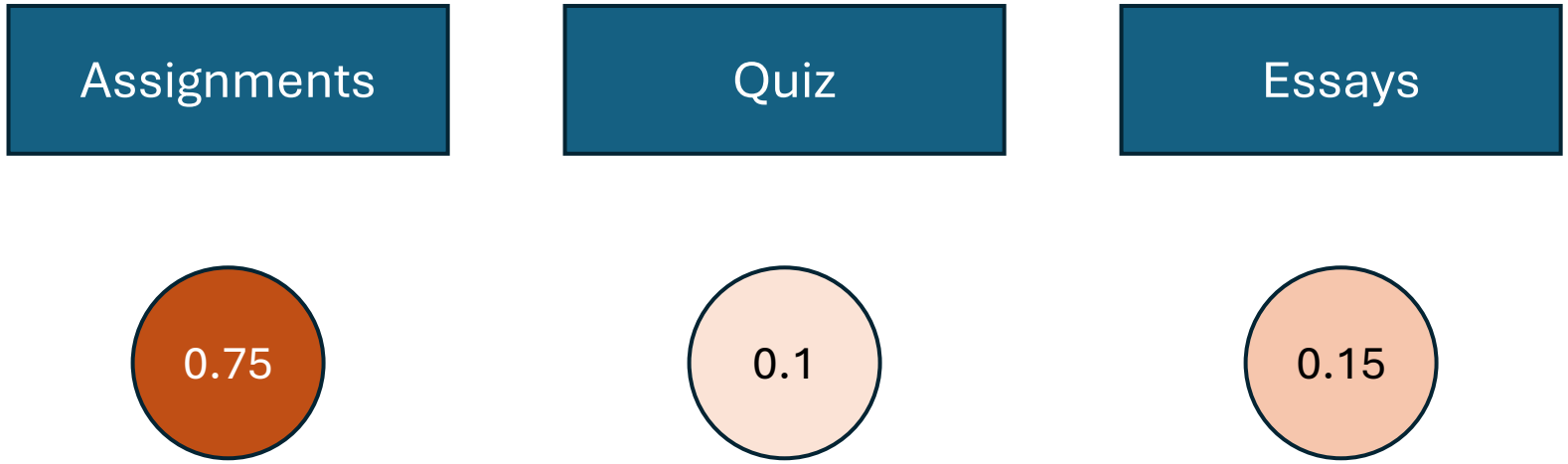
W_{rec} ~ large ⇨ Exploding

Formula Source: Razvan Pascanu et al. (2013)

Transformers

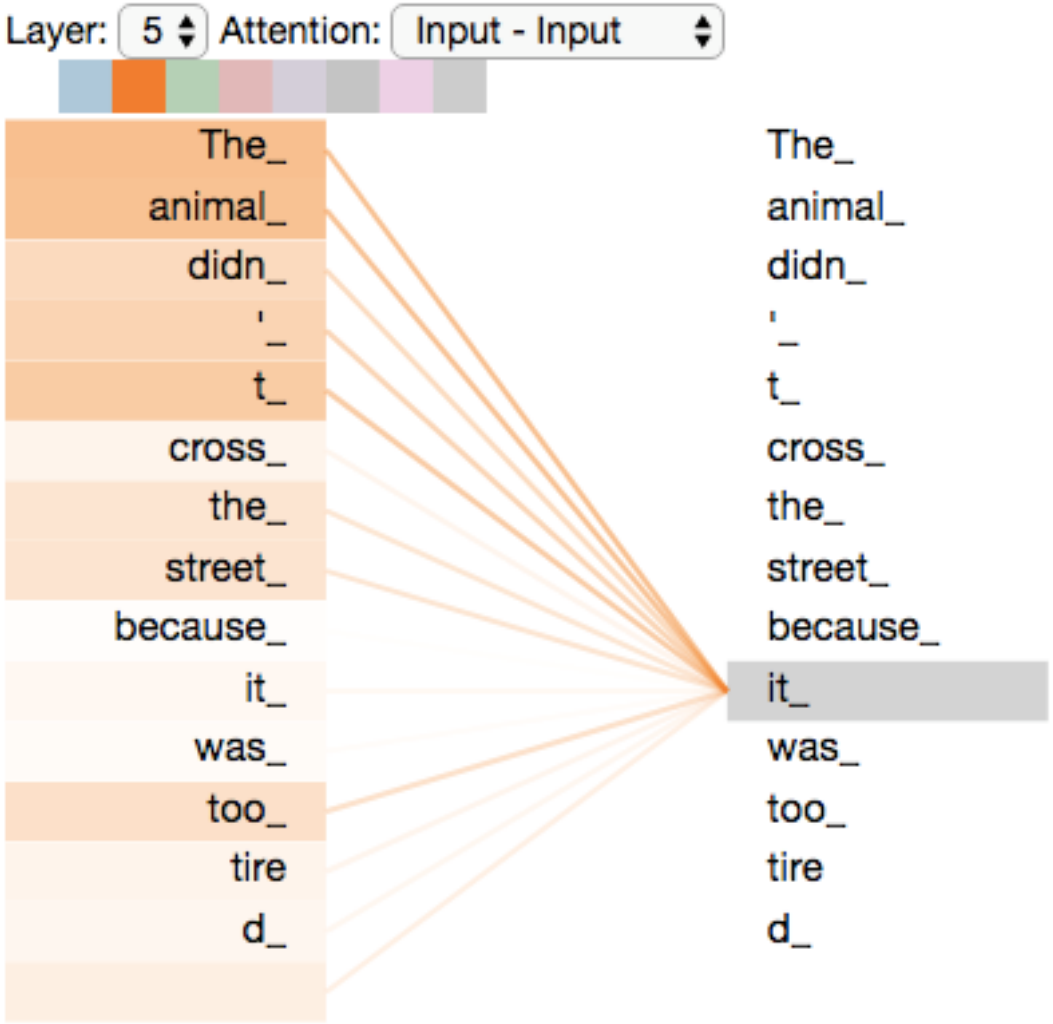


Attention Mechanism

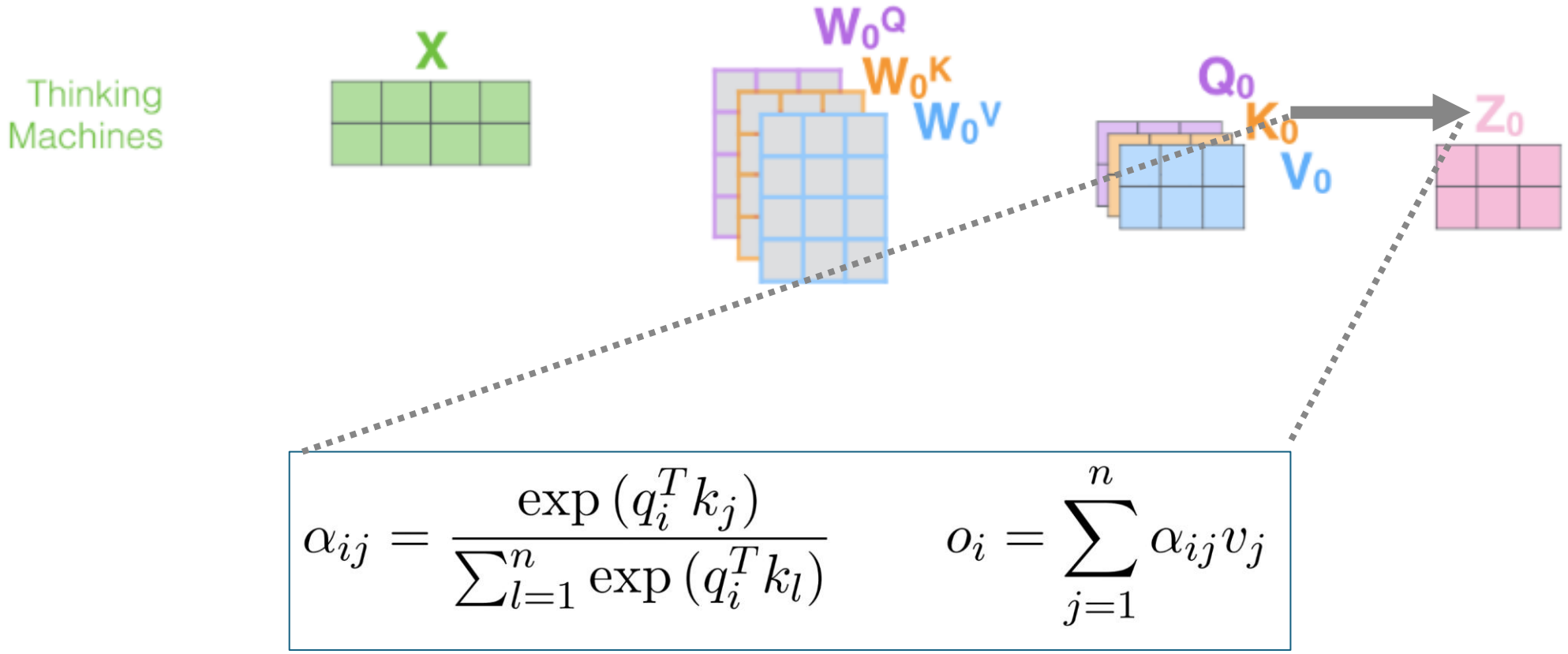


$$\text{Total} = 0.75 * A + 0.1 * Q + 0.15 * E$$

Attention Mechanism

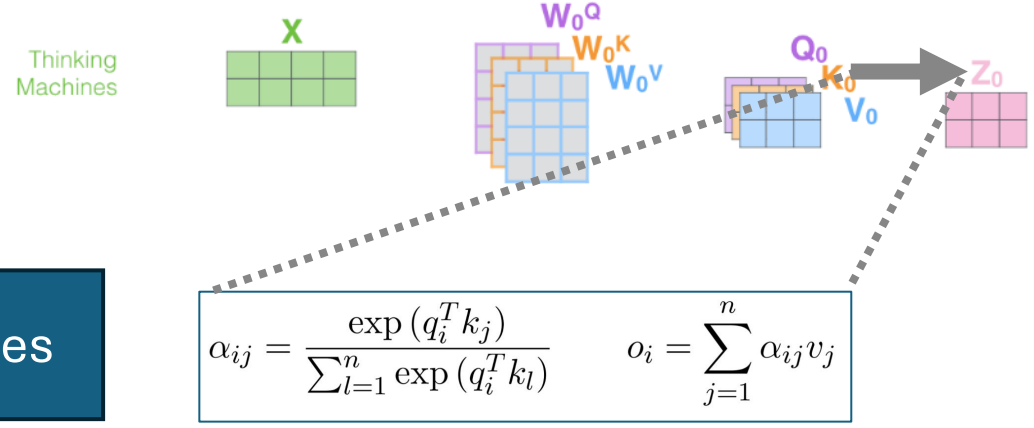


Self Attention



$$\alpha_{ij} = \frac{\exp\left(q_i^T k_j\right)}{\sum_{l=1}^{n} \exp\left(q_i^T k_l\right)} \qquad o_i = \sum_{j=1}^{n} \alpha_{ij} v_j$$

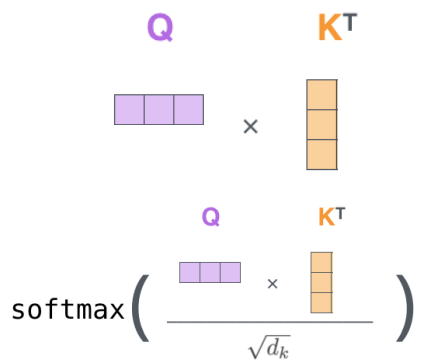
Self Attention



I	like	eating	apples

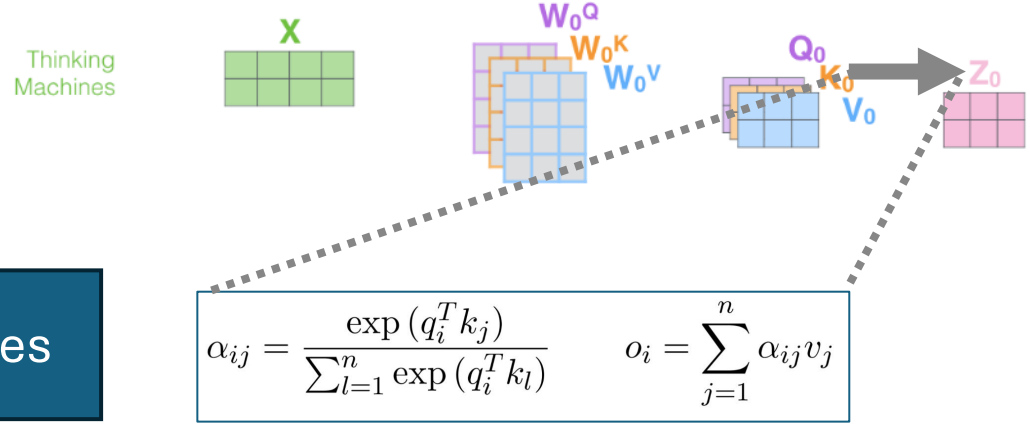
What is o(like)? I.e., the attention score from like to the sentence.

query		Q2 “like”		
key	K1 “I”	K2 “like”	K3 “eating”	K4 “apples”
attention weights*	14.2	18.1	10.3	7.9
attention scores* (o)	0.3	0.6	0.07	0.03



*: made-up numbers, not real.

Self Attention



I | like | eating | apples

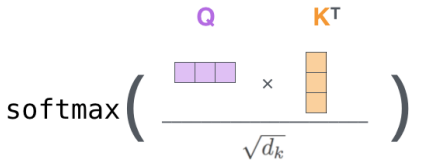
What is z(like)?

attention scores* (o): 0.3 | 0.6 | 0.07 | 0.03

values: V1 "I" | V2 "like" | V3 "eating" | V4 "apples"

Weighted sum

Z2 "like"



*: made-up numbers, not real.

Multi-Head Self Attention

1) This is our input sentence*

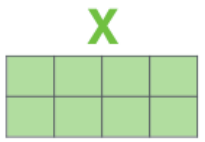
2) We embed each word*

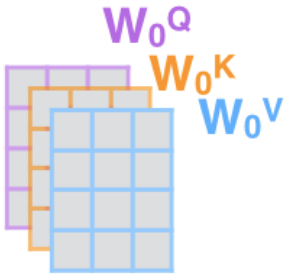
3) Split into 8 heads. We multiply X or R with weight matrices

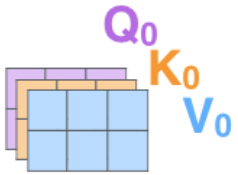
4) Calculate attention using the resulting Q/K/V matrices

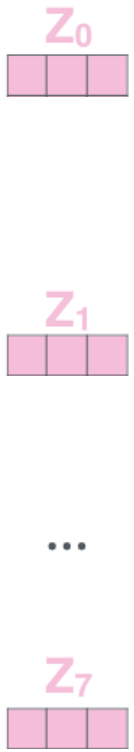
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer

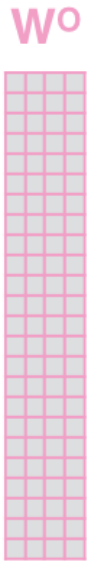
Thinking Machines

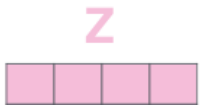




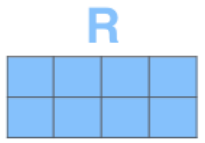




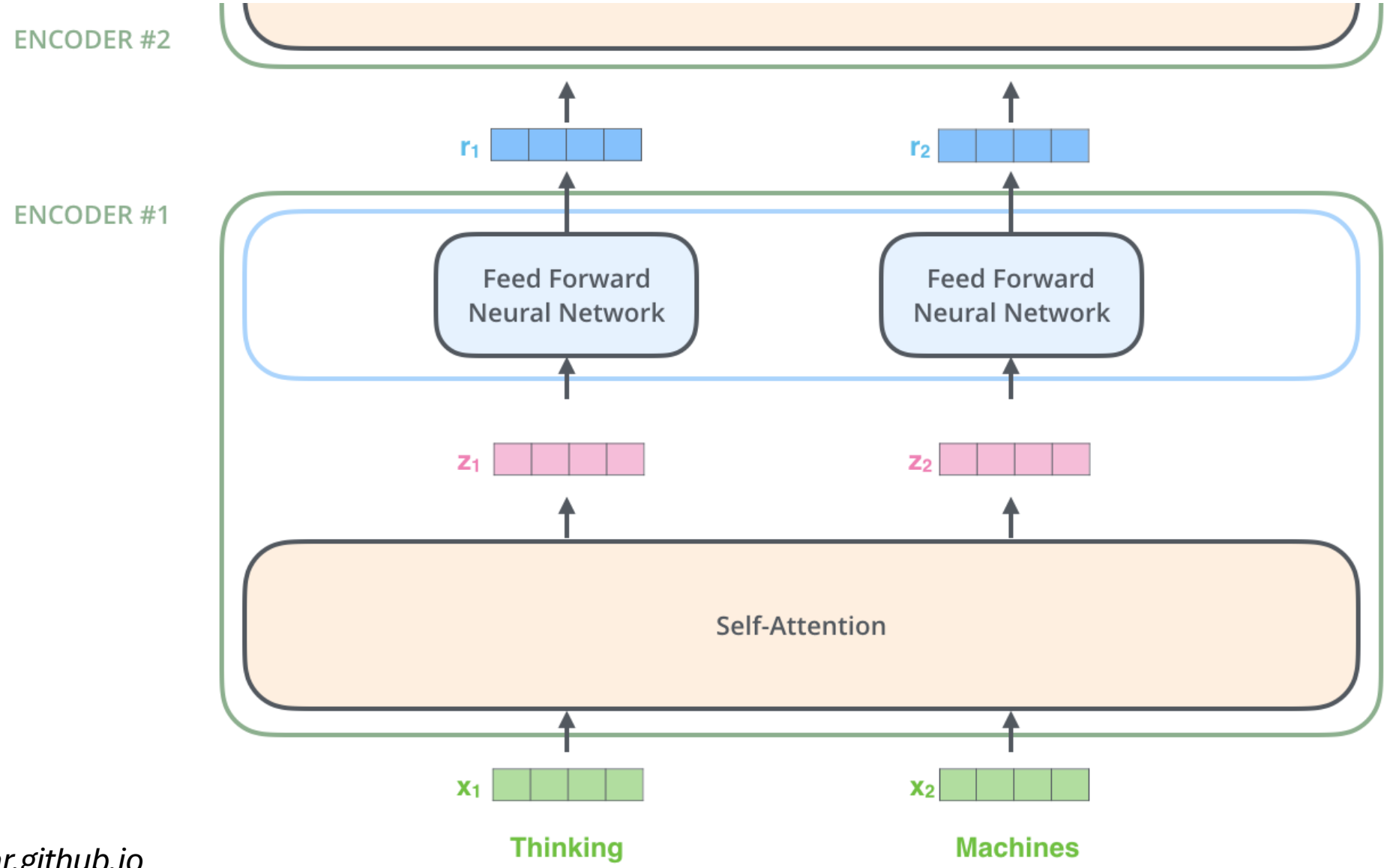




* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one



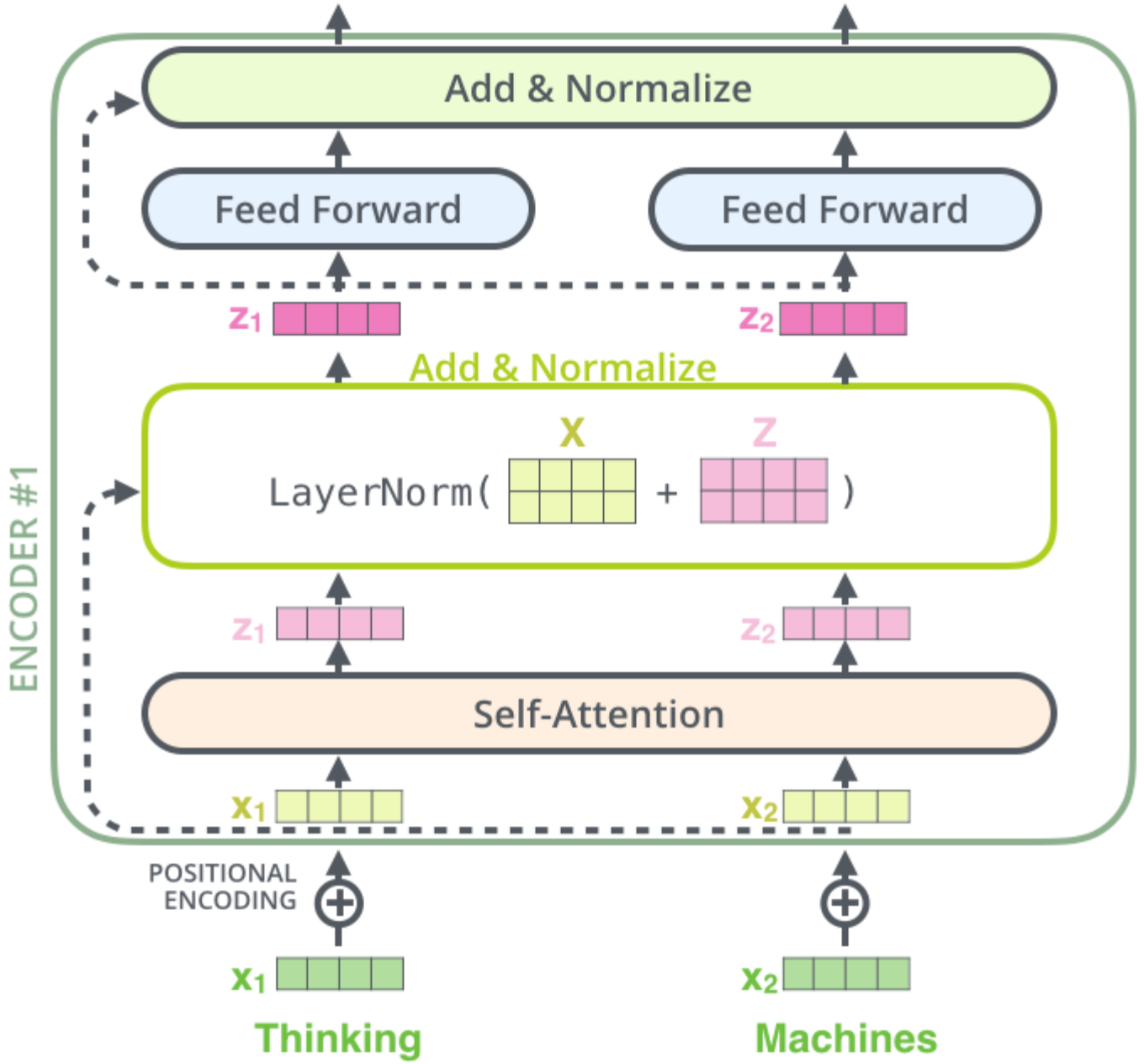
Transformers



Typically
2-layer MLP

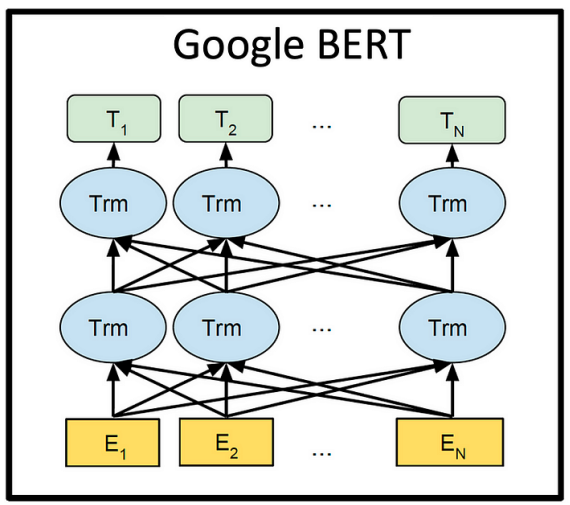
mlp_out + residual_mid

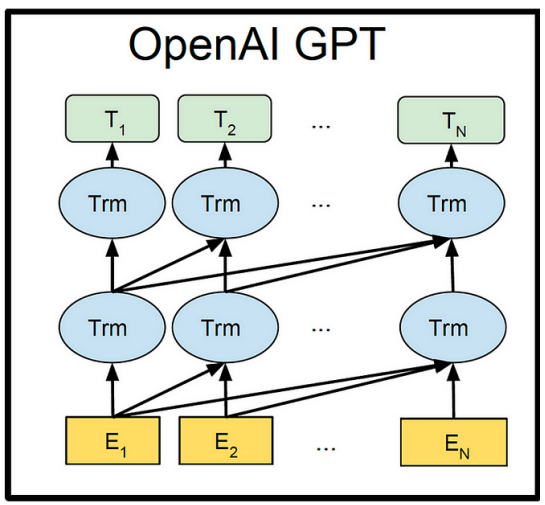
residual_mid



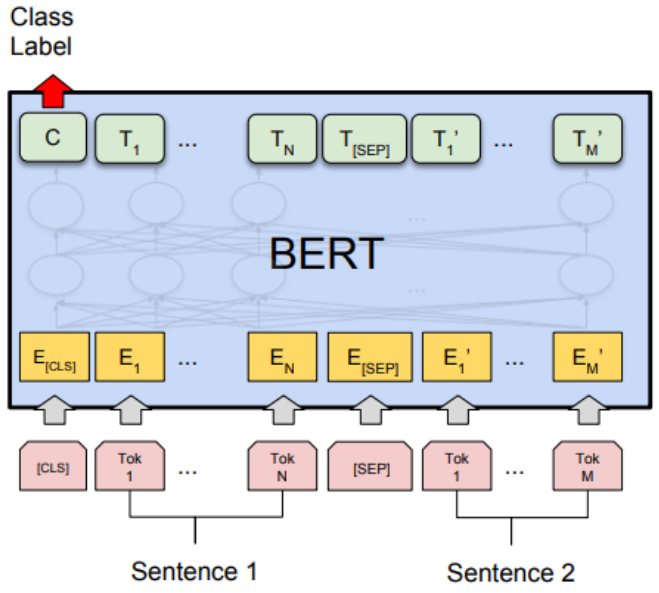
Residuals


“Ogres are like onions ....
they have layers”
- Shrek

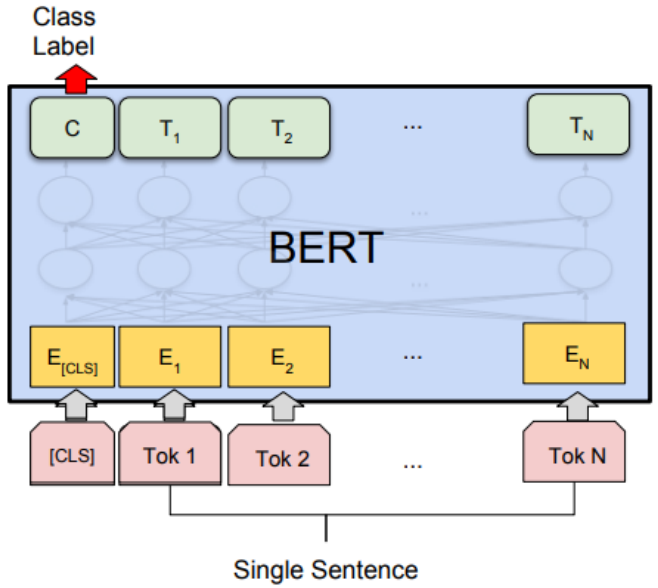

Google BERT
T1
T2
...
TN
Trm
Trm
...
Trm
Trm
Trm
...
Trm
E1
E2
...
EN


OpenAI GPT
T1
T2
...
TN
Trm
Trm
...
Trm
Trm
Trm
...
Trm
E1
E2
...
EN

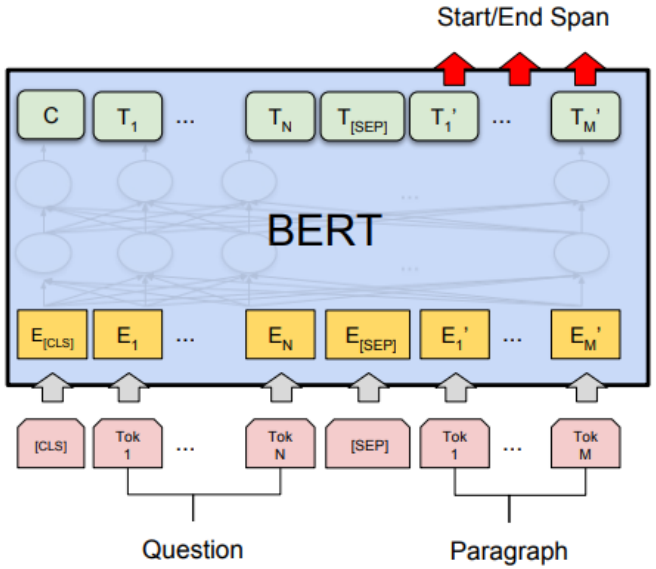
BERT: Pretrain-Finetune Paradigm

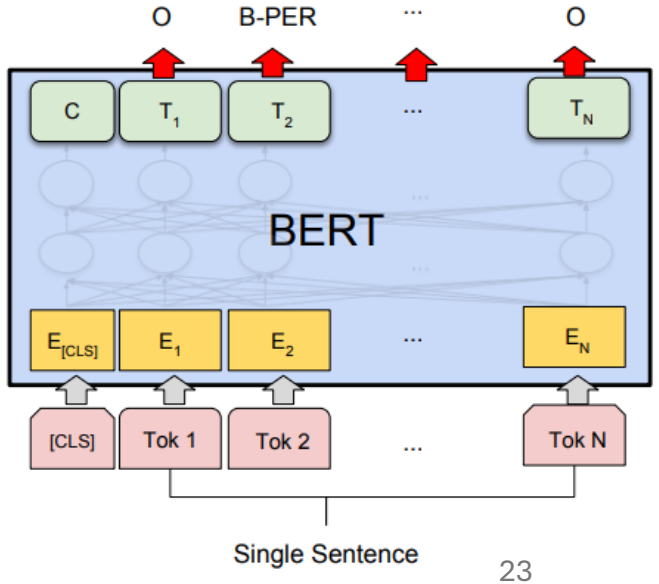


(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG

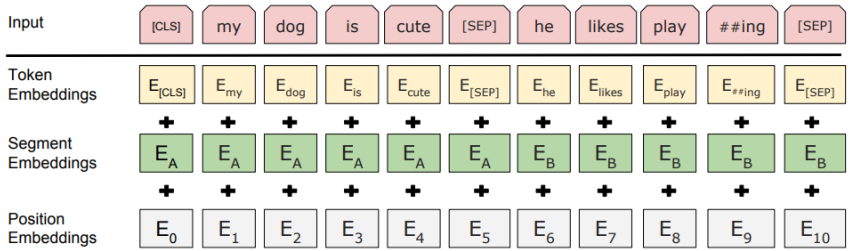


(b) Single Sentence Classification Tasks: SST-2, CoLA





Position Encoding



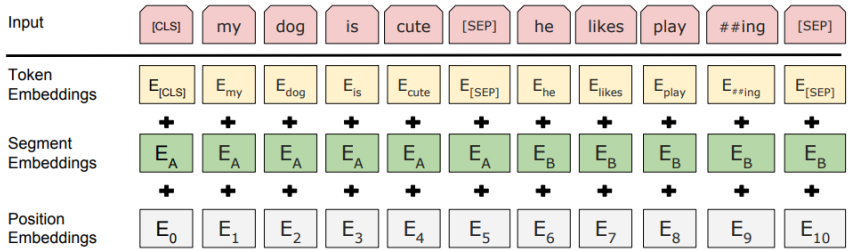
$$\overrightarrow{p_t} = \begin{bmatrix} \sin(\omega_1 . t) \\ \cos(\omega_1 . t) \\ \\ \sin(\omega_2 . t) \\ \cos(\omega_2 . t) \\ \\ \vdots \\ \\ \sin(\omega_{d/2} . t) \\ \cos(\omega_{d/2} . t) \end{bmatrix}_{d \times 1}$$

where

$$\overrightarrow{p_t}^{(i)} = f(t)^{(i)} := \begin{cases} \sin(\omega_k . t), & \text{if } i = 2k \\ \cos(\omega_k . t), & \text{if } i = 2k+1 \end{cases}$$

$$\omega_k = \frac{1}{10000^{2k/d}}$$

Position Encoding



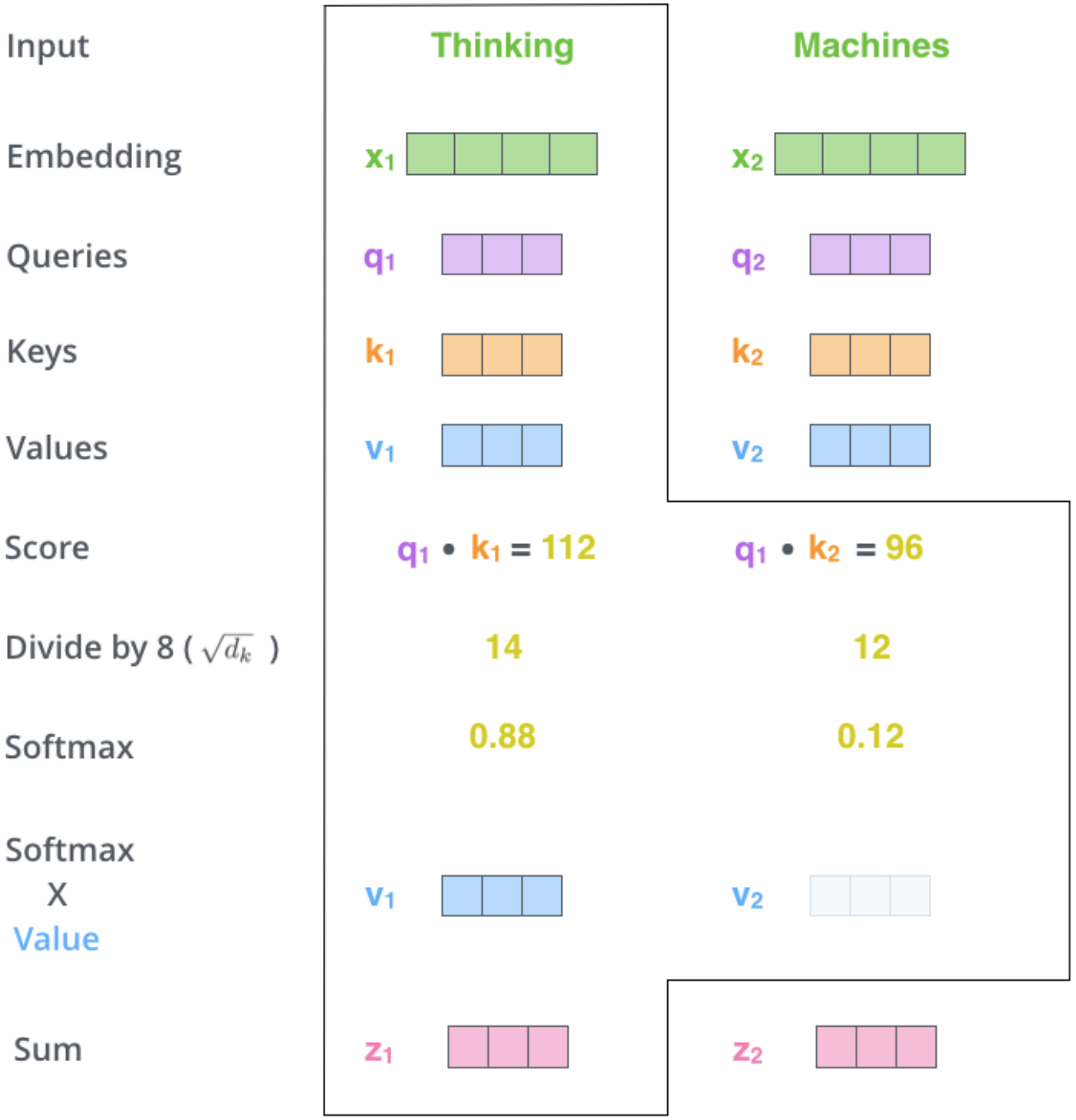
- Encodings of any two distinct positions are distinct
- Each position maps to only one encoding
- Test sentences may be longer than training
- Distance between two positions should be constant across sentences (of varying lengths).

$$\vec{p_t} = \begin{bmatrix} \sin(\omega_1 . t) \\ \cos(\omega_1 . t) \\ \\ \sin(\omega_2 . t) \\ \cos(\omega_2 . t) \\ \\ \vdots \\ \\ \sin(\omega_{d/2} . t) \\ \cos(\omega_{d/2} . t) \end{bmatrix}_{d \times 1}$$

where

$$\vec{p_t}^{(i)} = f(t)^{(i)} := \begin{cases} \sin(\omega_k . t), & \text{if } i = 2k \\ \cos(\omega_k . t), & \text{if } i = 2k+1 \end{cases}$$

$$\omega_k = \frac{1}{10000^{2k/d}}$$



Review

- Embeddings:
  - Token + Sentence + Position
- Multi-Head Attention
- Feed forward module (MLP)
- Layers

More Transformer in This Course

- Use Transformer Models to build all kinds of applications.
  - Parser (A1!), Word sense, QA, LLMs…
- Why do Transformers work so well?
  - Interpretability.
- How can we control them.
  - Interpretability, model editing.