Foundation Models

(Large Pre-trained Models)

Sarah Pratt

Language Models

Encoders vs Decoders vs Encoder-Decoder Models

Prompting (zero-shot, in-context, chain-of-thought)

Vision + Language Models

CLIP training + inference

Results + Robustness

My prior work

Language Models

Language Models

It’s cold today! Don’t forget to wear a _______.

The ______________ is a popular tourist attraction in Seattle.

I missed ___ bus.

I had 3 pencils and lost one so now I have _______ pencils.

Language Models

It’s cold today! Don’t forget to wear a **jacket**.

The ______________ is a popular tourist attraction in Seattle.

I missed ___ bus.

I had 3 pencils and lost one so now I have _______ pencils.

Language Models

It’s cold today! Don’t forget to wear a **jacket**.

The **Space Needle** is a popular tourist attraction in Seattle.

I missed ___ bus.

I had 3 pencils and lost one so now I have _______ pencils.

Language Models

It’s cold today! Don’t forget to wear a **jacket**.

The **Space Needle** is a popular tourist attraction in Seattle.

I missed **the** bus.

I had 3 pencils and lost one so now I have _______ pencils.

Language Models

It’s cold today! Don’t forget to wear a **jacket**.

The **Space Needle** is a popular tourist attraction in Seattle.

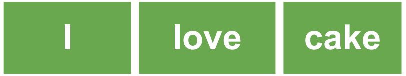
I missed **the** bus.

I had 3 pencils and lost one so now I have **two** pencils.

<u>Encoder Only:</u> Capture the meaning of an entire sequence

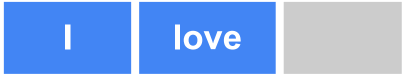
I	love	cake

Encoder Only: Capture the meaning of an entire sequence



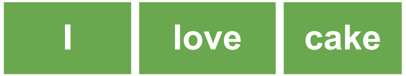
I love cake

Decoder Only: Generate text based on previously generated text



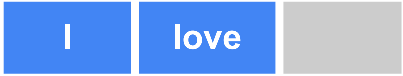
I love

Encoder Only: Capture the meaning of an entire sequence



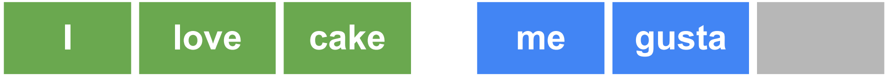
I love cake

Decoder Only: Generate text based on previously generated text



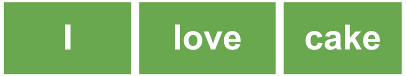
I love

Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence

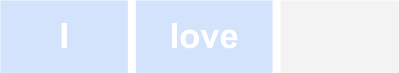


I love cake me gusta

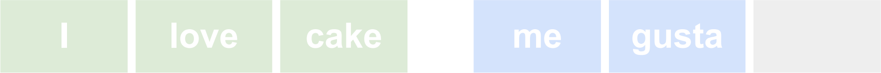
Encoder Only: Capture the meaning of an entire sequence



Decoder Only: Generate text based on previously generated text



Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence



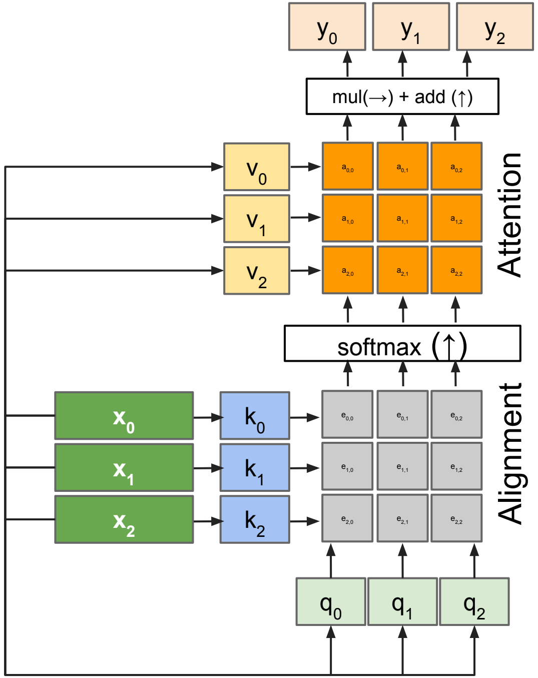
Encoder Only: Capture the meaning of an entire sequence

Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Encoder Only: Capture the meaning of an entire sequence

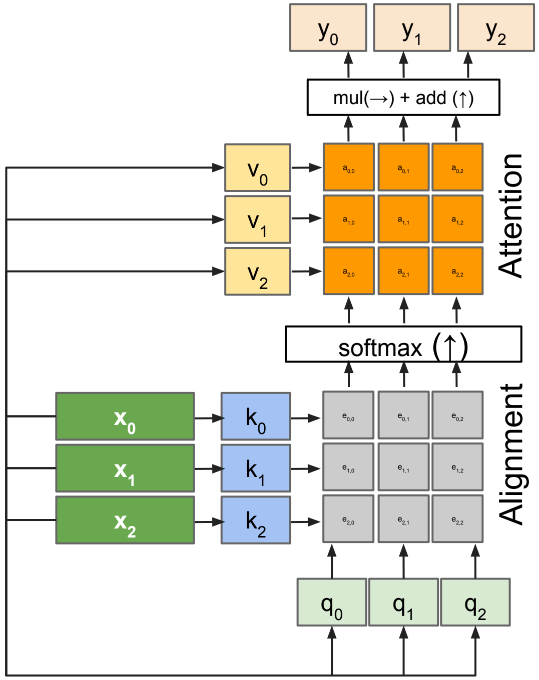
Example Model: BERT



Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence

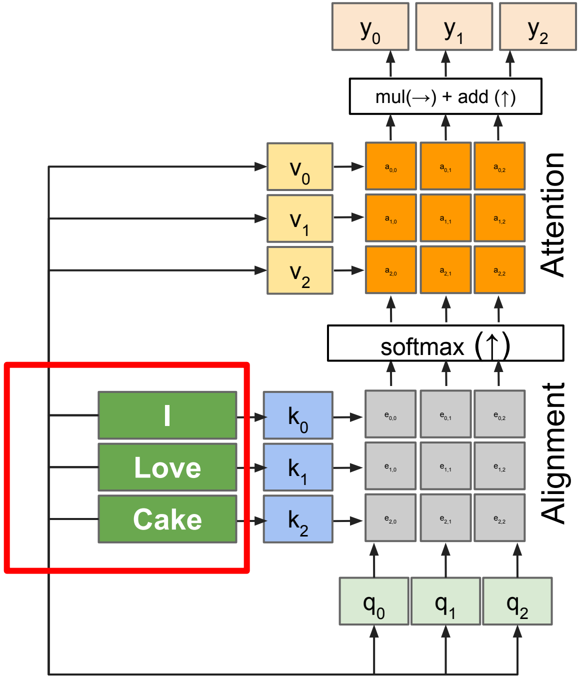


y_0
y_1
y_2
mul(→) + add (↑)
v_0
v_1
v_2
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$
Attention
softmax (↑)
x_0
x_1
x_2
k_0
k_1
k_2
$e_{0,0}$ $e_{0,1}$ $e_{0,2}$
$e_{1,0}$ $e_{1,1}$ $e_{1,2}$
$e_{2,0}$ $e_{2,1}$ $e_{2,2}$
Alignment
q_0
q_1
q_2

Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence



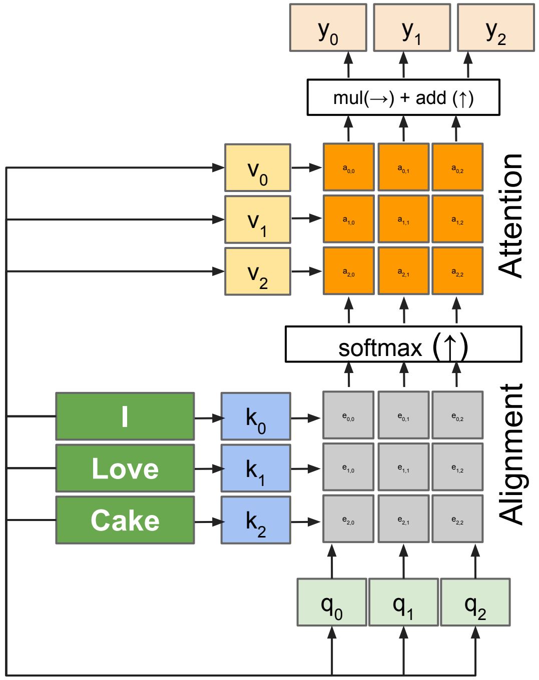
Input = text token embeddings (and positional embedding)

Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence

Output: Feature Vector



Outputs:
context vectors: **y** (shape: D_v)

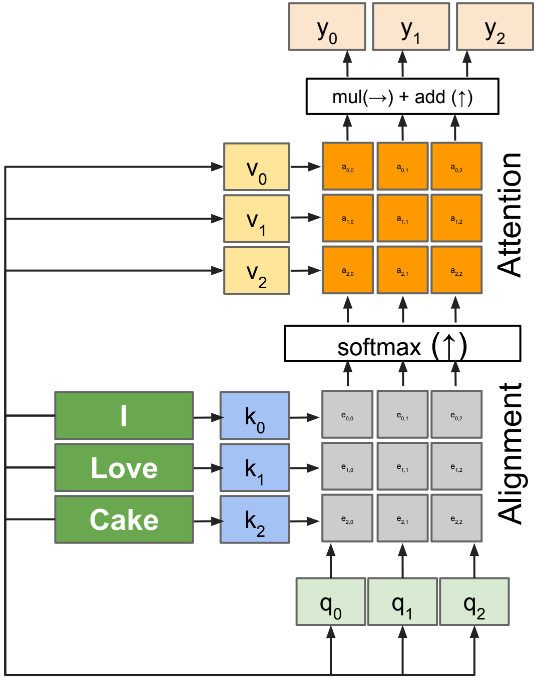
Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence

Output: Feature Vector

What information do the y vectors contain?



y_0
y_1
y_2
mul(→) + add (↑)
V_0
V_1
V_2
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$
Attention
softmax (↑)
I
Love
Cake
k_0
k_1
k_2
$e_{0,0}$ $e_{0,1}$ $e_{0,2}$
$e_{1,0}$ $e_{1,1}$ $e_{1,2}$
$e_{2,0}$ $e_{2,1}$ $e_{2,2}$
Alignment
q_0
q_1
q_2

Outputs:
context vectors: **y** (shape: D_v)

Encoder Only: Capture the meaning of an entire sequence

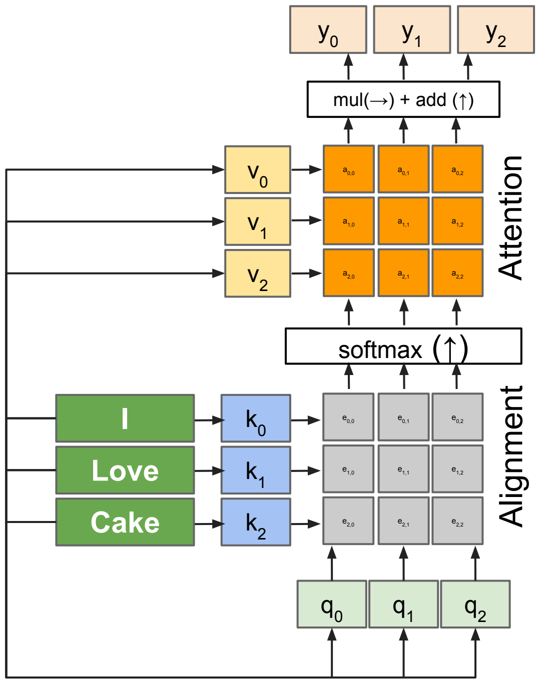
Example Model: BERT

Input: Text sequence

Output: Feature Vector

What information do the y vectors contain?

Nothing, yet!



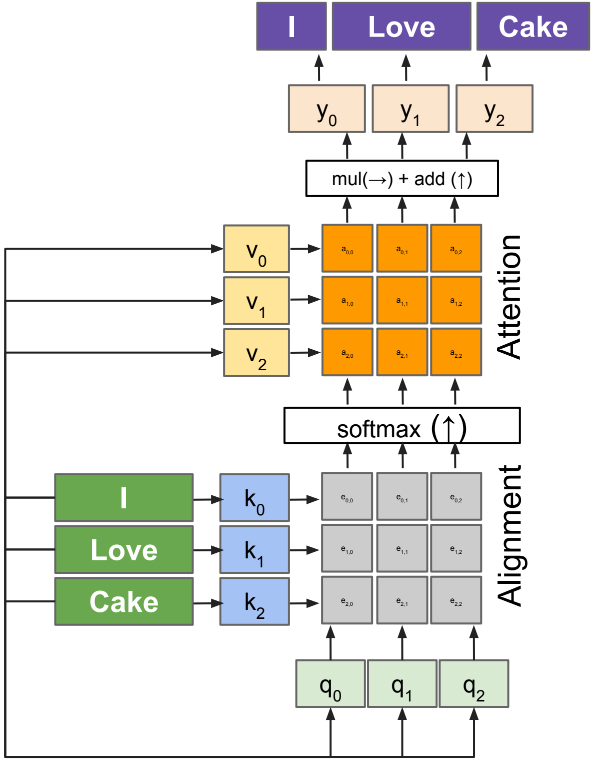
Outputs:
context vectors: **y** (shape: D_v)

Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence

Output: Feature Vector



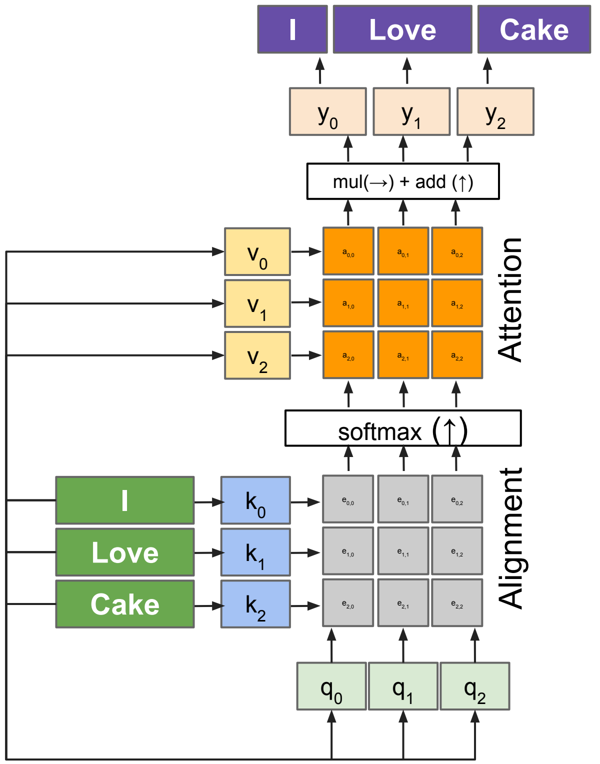
Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence

Output: Feature Vector

What information do the y vectors contain?



Encoder Only: Capture the meaning of an entire sequence

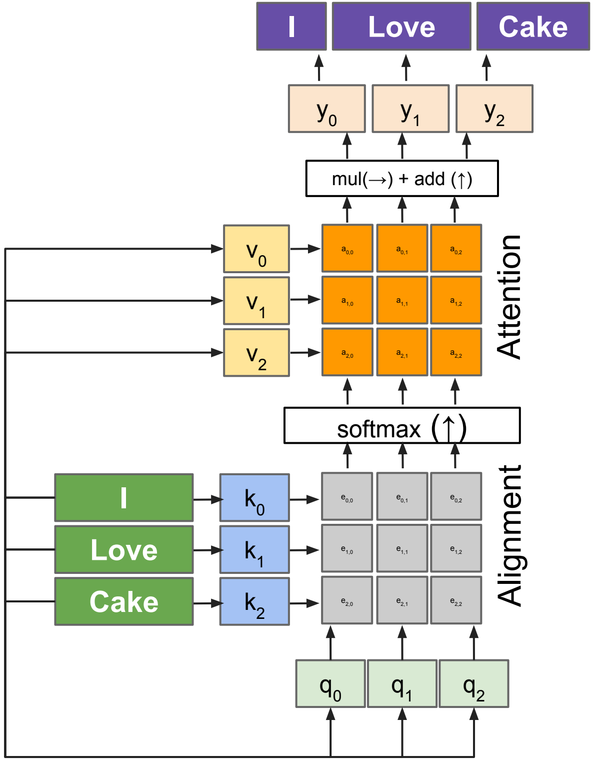
Example Model: BERT

Input: Text sequence

Output: Feature Vector

What information do the y vectors contain?

Just copying input



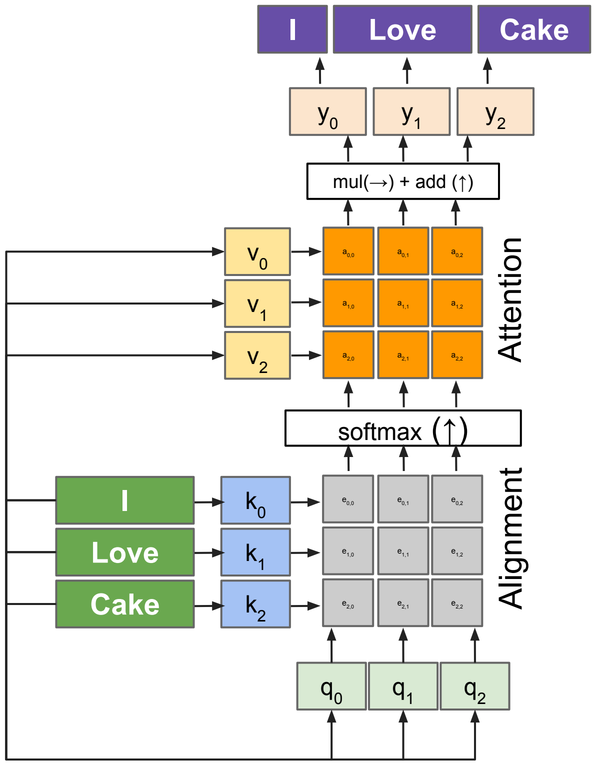
Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence

Output: Feature Vector

How to we force this model to learn semantic/factual/grammatical/logical information?



Language Models

It’s cold today! Don’t forget to wear a **jacket**.

The **Space Needle** is a popular tourist attraction in Seattle.

I missed **the** bus.

I had 3 pencils and lost one so now I have **two** pencils.

Language Models

It’s cold today! Don’t forget to wear a **jacket**. **Semantic**

The **Space Needle** is a popular tourist attraction in Seattle. **Factual**

I missed **the** bus. **Grammatical**

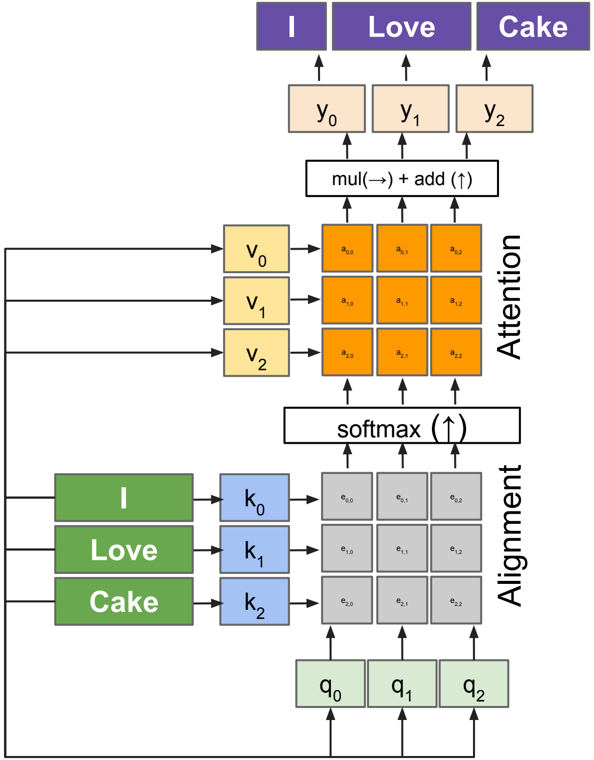
I had 3 pencils and lost one so now I have **two** pencils. **Logical**

Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence

Output: Feature Vector



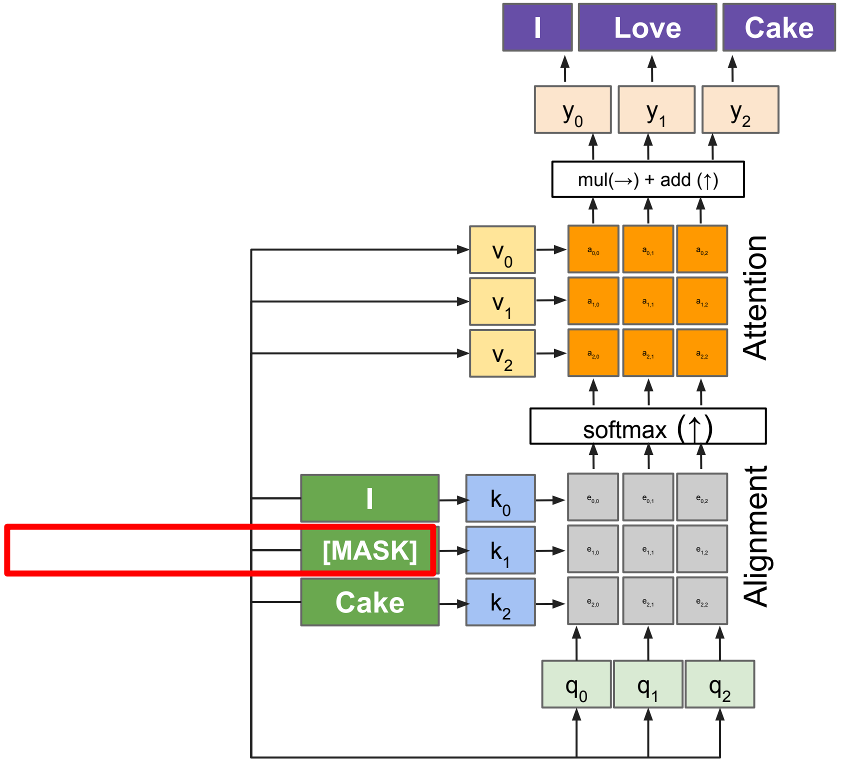
I
Love
Cake
y_0
y_1
y_2
mul(→) + add (↑)
v_0
v_1
v_2
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$
Attention
softmax (↑)
I
Love
Cake
k_0
k_1
k_2
$e_{0,0}$ $e_{0,1}$ $e_{0,2}$
$e_{1,0}$ $e_{1,1}$ $e_{1,2}$
$e_{2,0}$ $e_{2,1}$ $e_{2,2}$
Alignment
q_0
q_1
q_2

Encoder Only: Capture the meaning of an entire sequence

Example Model: BERT

Input: Text sequence

Output: Feature Vector



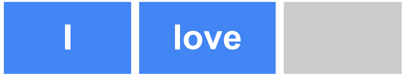
Randomly select 15% of tokens.

- **80% - [MASK]**
- **10% - random token**
- **10% - keep same**

Encoder Only: Capture the meaning of an entire sequence

I love cake

Decoder Only: Generate text based on previously generated text

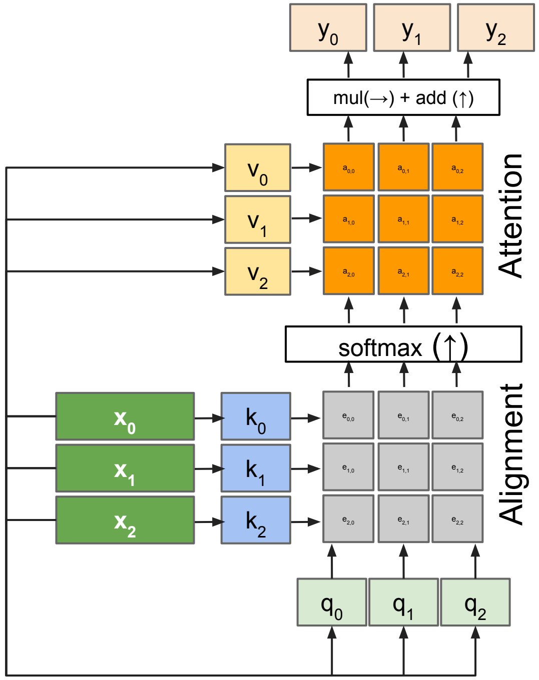

I love

Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence sequence

I love cake me gusta

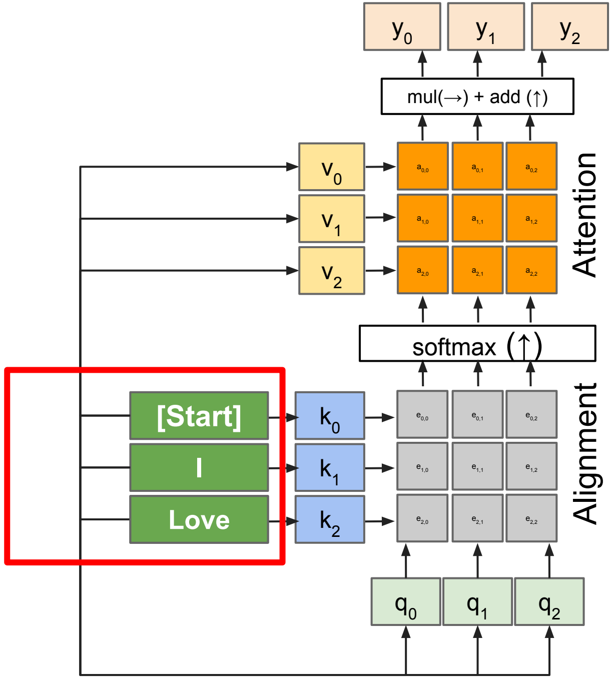
Decoder Only: Generate text based on previously generated text

Decoder Only: Generate text based on previously generated text



Decoder Only: Generate text based on previously generated text

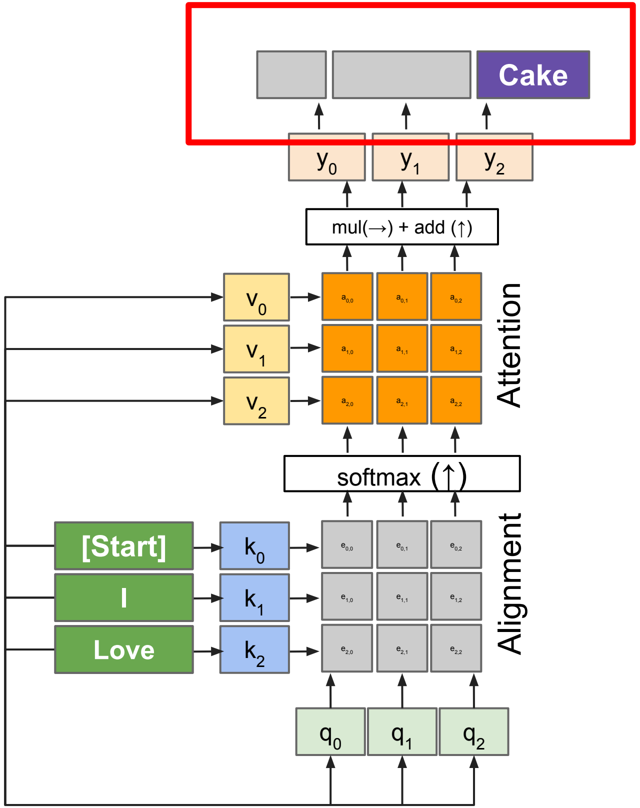
Input: Text sequence



Decoder Only: Generate text based on previously generated text

Input: Text sequence

Output: Completed text sequence

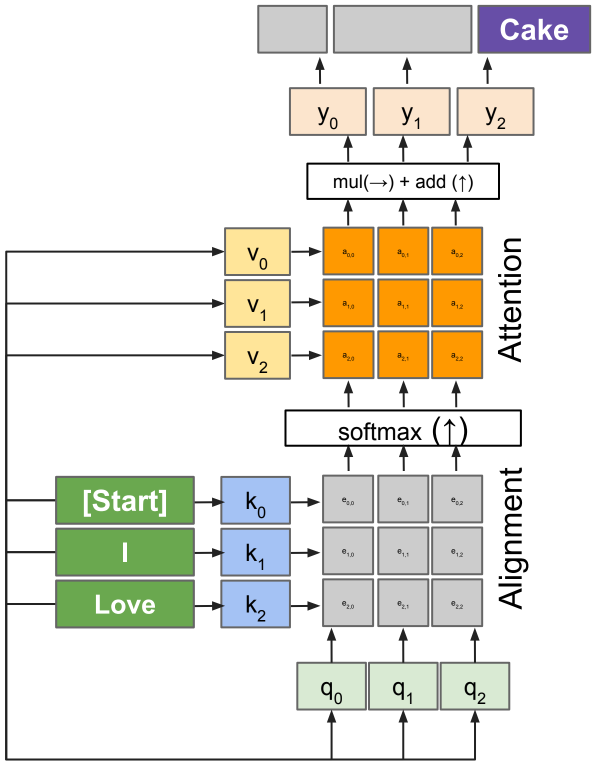


Decoder Only: Generate text based on previously generated text

Input: Text sequence

Output: Completed text sequence

Cons: Need to process entire sentence in order to get loss from one word - not very much signal for the amount of processing



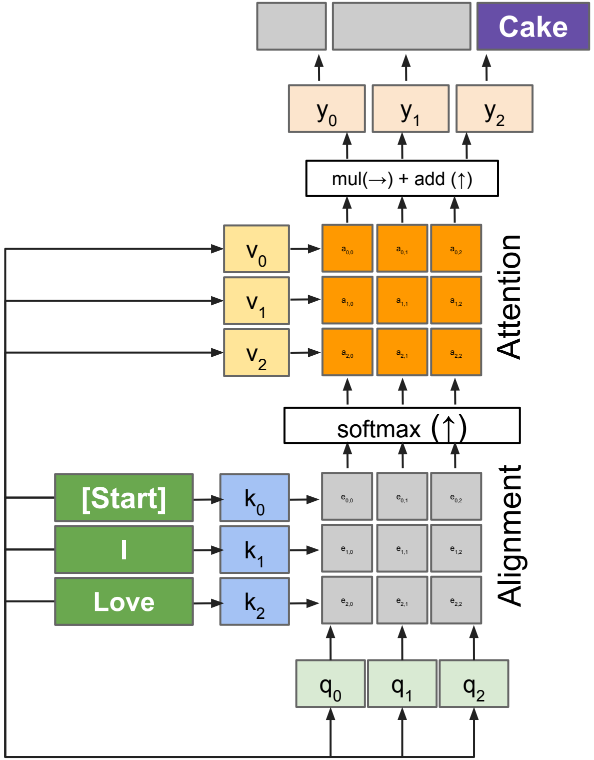
Decoder Only: Generate text based on previously generated text

Input: Text sequence

Output: Completed text sequence

Cons: Need to process entire sentence in order to get loss from one word - not very much signal for the amount of processing

Solution: predict each word given previous words so far



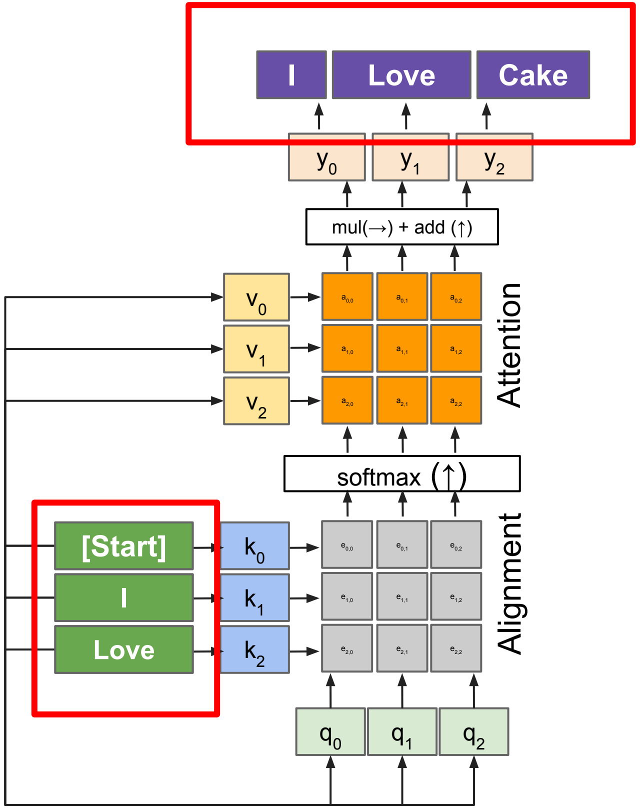
Decoder Only: Generate text based on previously generated text

Input: Text sequence

Output: Completed text sequence

Cons: Need to process entire sentence in order to get loss from one word - not very much signal for the amount of processing

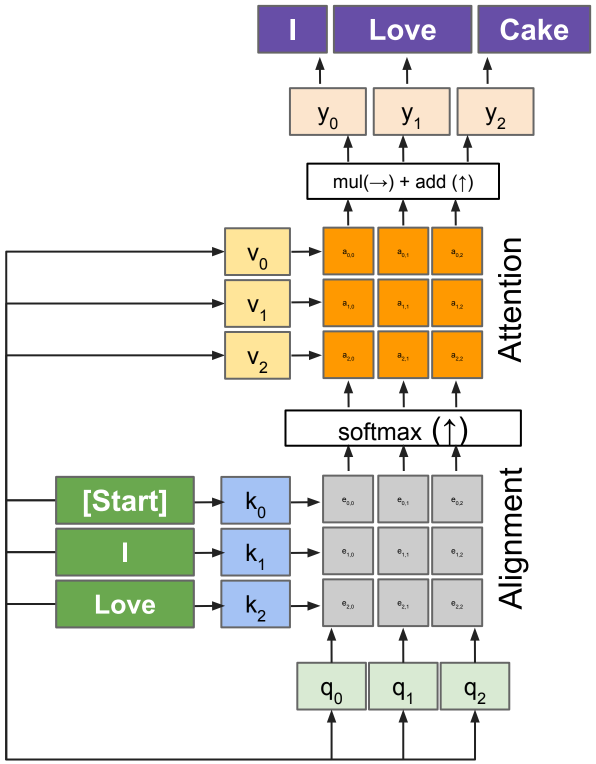
Solution: predict each word given previous words so far



Decoder Only: Generate text based on previously generated text

Input: Text sequence

Output: Completed text sequence

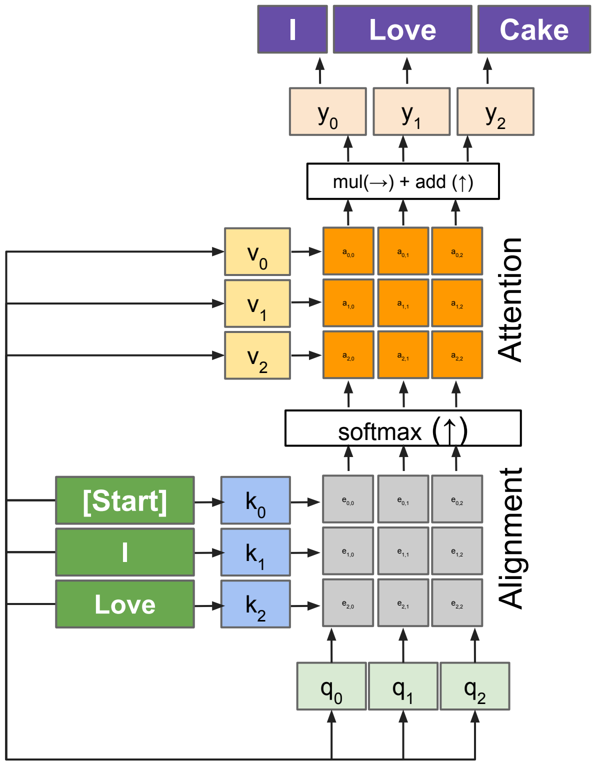


Decoder Only: Generate text based on previously generated text

Input: Text sequence

Output: Completed text sequence

What's wrong with this?



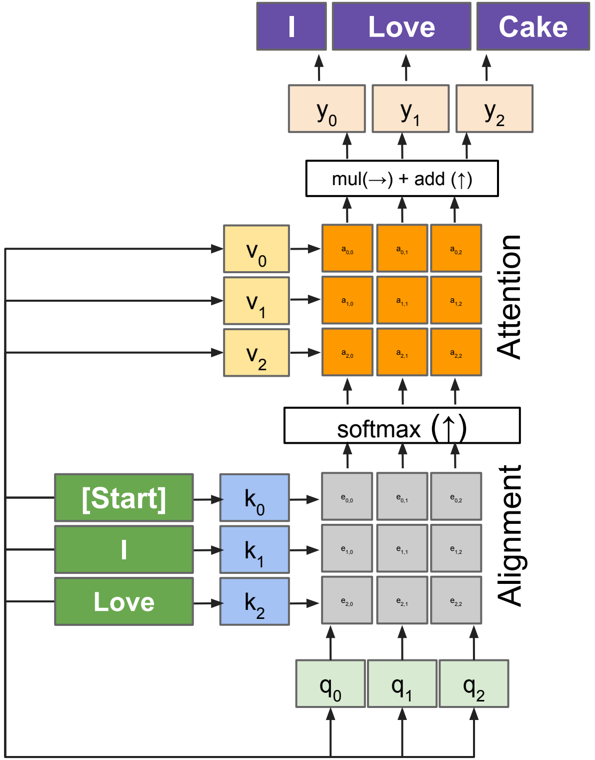
Decoder Only: Generate text based on previously generated text

Input: Text sequence

Output: Completed text sequence

What's wrong with this?

It can see the answer!



Decoder Only: Generate text based on previously generated text

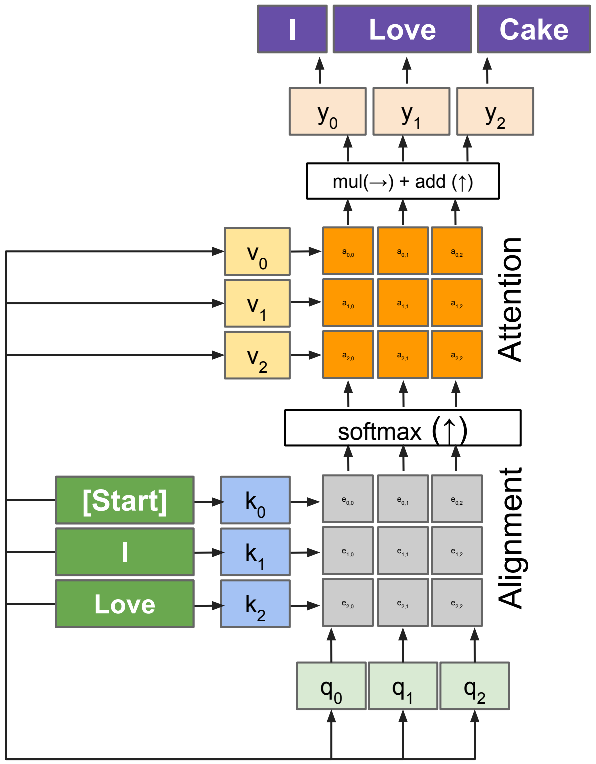
Input: Text sequence

Output: Completed text sequence

What’s wrong with this?

It can see the answer!

Solution: zero out values from future words



Decoder Only: Generate text based on previously generated text

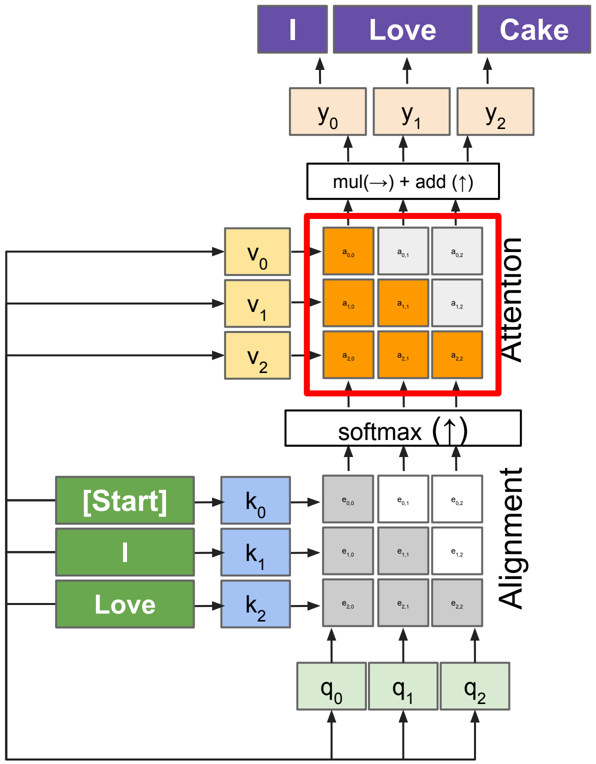
Input: Text sequence

Output: Completed text sequence

What's wrong with this?

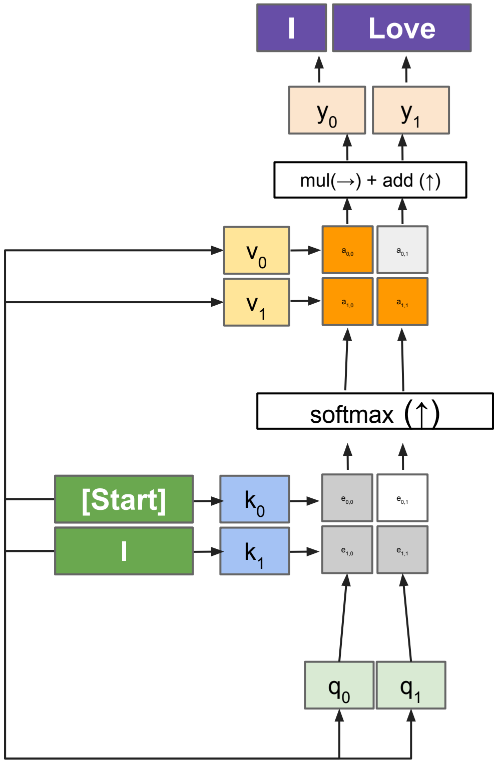
It can see the answer!

Solution: zero out values from future words

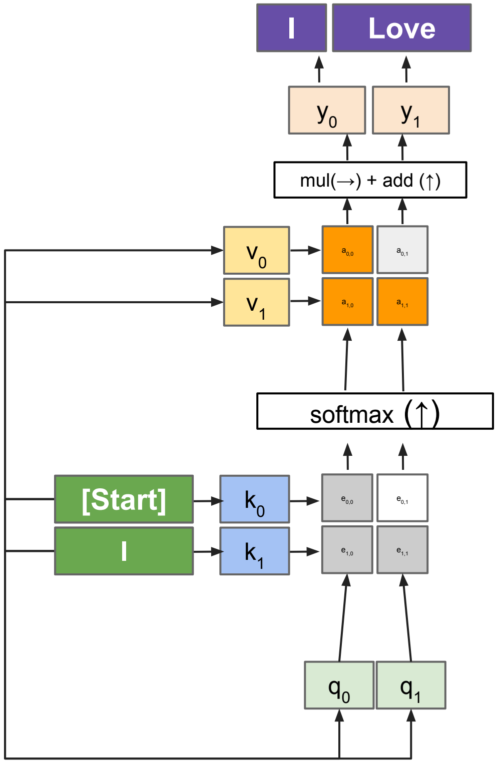


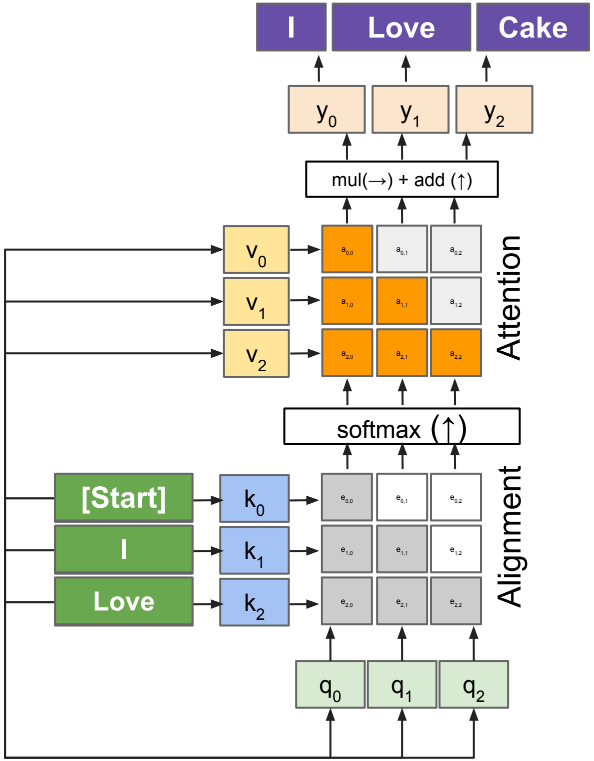
Decoder Only: Inference

Decoder Only: Inference

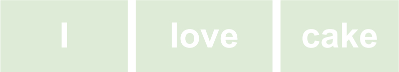


Decoder Only: Inference

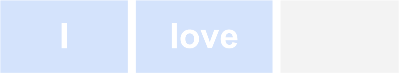




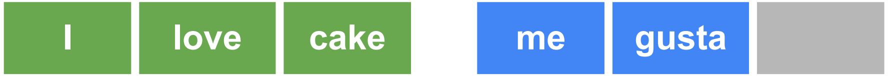
Encoder Only: Capture the meaning of an entire sequence



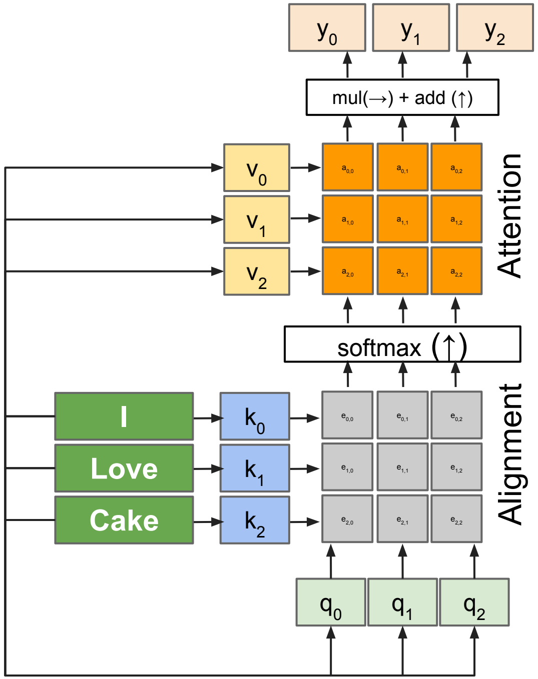
Decoder Only: Generate text based on previously generated text

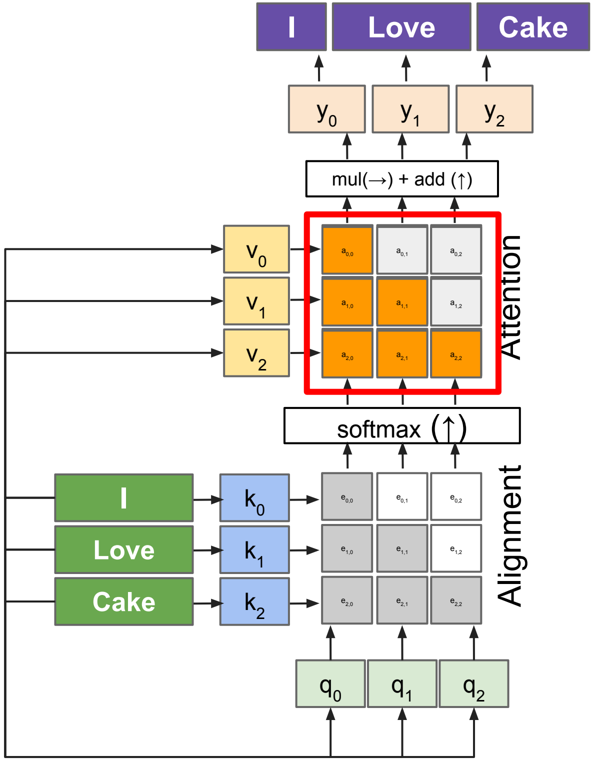


Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence

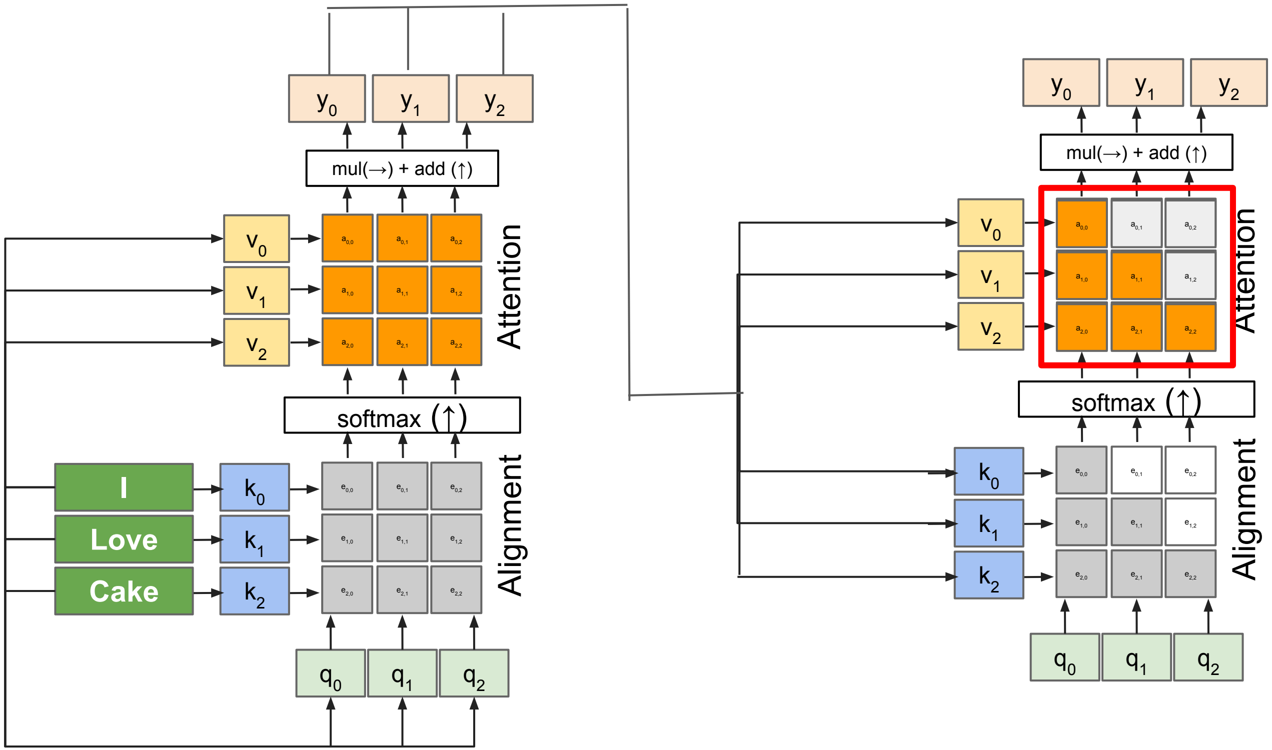


Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence


y_0
y_1
y_2
mul(→) + add (↑)
v_0
v_1
v_2
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$
Attention
softmax (↑)
I
Love
Cake
k_0
k_1
k_2
$e_{0,0}$ $e_{0,1}$ $e_{0,2}$
$e_{1,0}$ $e_{1,1}$ $e_{1,2}$
$e_{2,0}$ $e_{2,1}$ $e_{2,2}$
Alignment
q_0
q_1
q_2

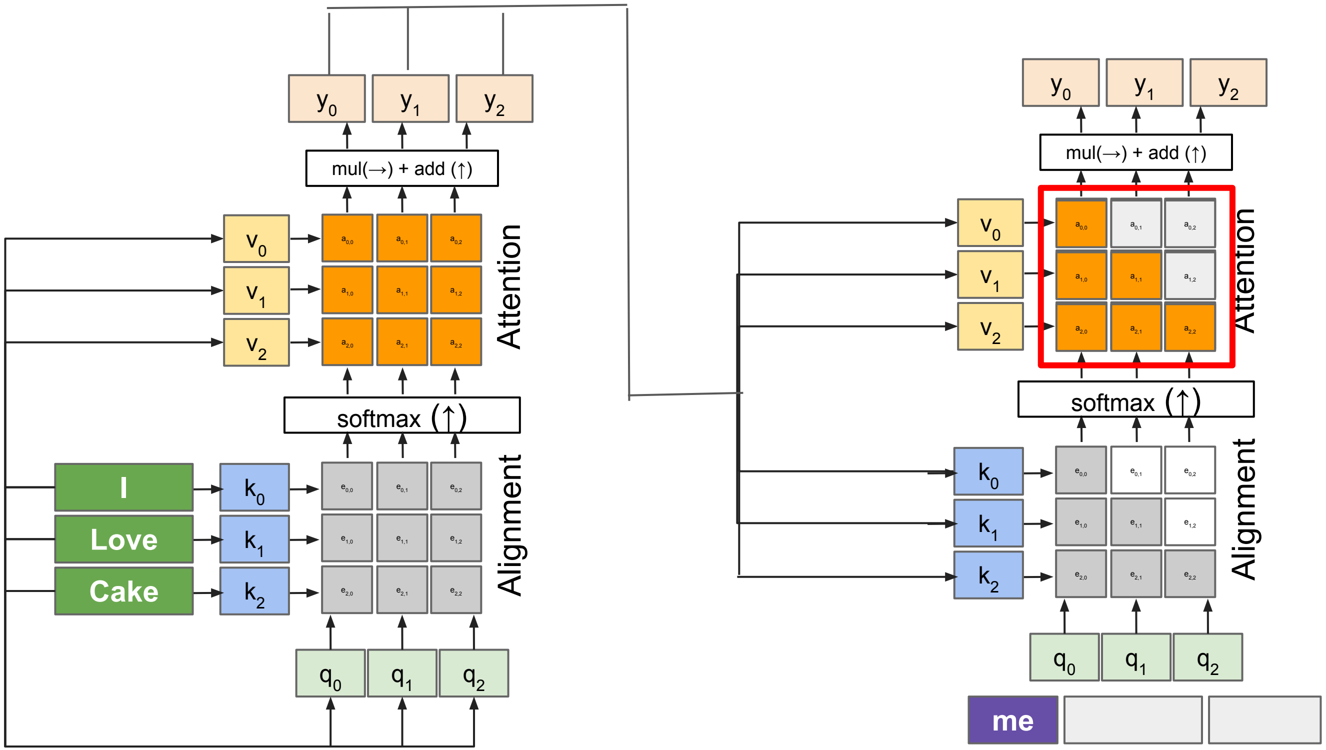

I
Love
Cake
y_0
y_1
y_2
mul(→) + add (↑)
v_0
v_1
v_2
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$
Attention
softmax (↑)
I
Love
Cake
k_0
k_1
k_2
$e_{0,0}$ $e_{0,1}$ $e_{0,2}$
$e_{1,0}$ $e_{1,1}$ $e_{1,2}$
$e_{2,0}$ $e_{2,1}$ $e_{2,2}$
Alignment
q_0
q_1
q_2

Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence

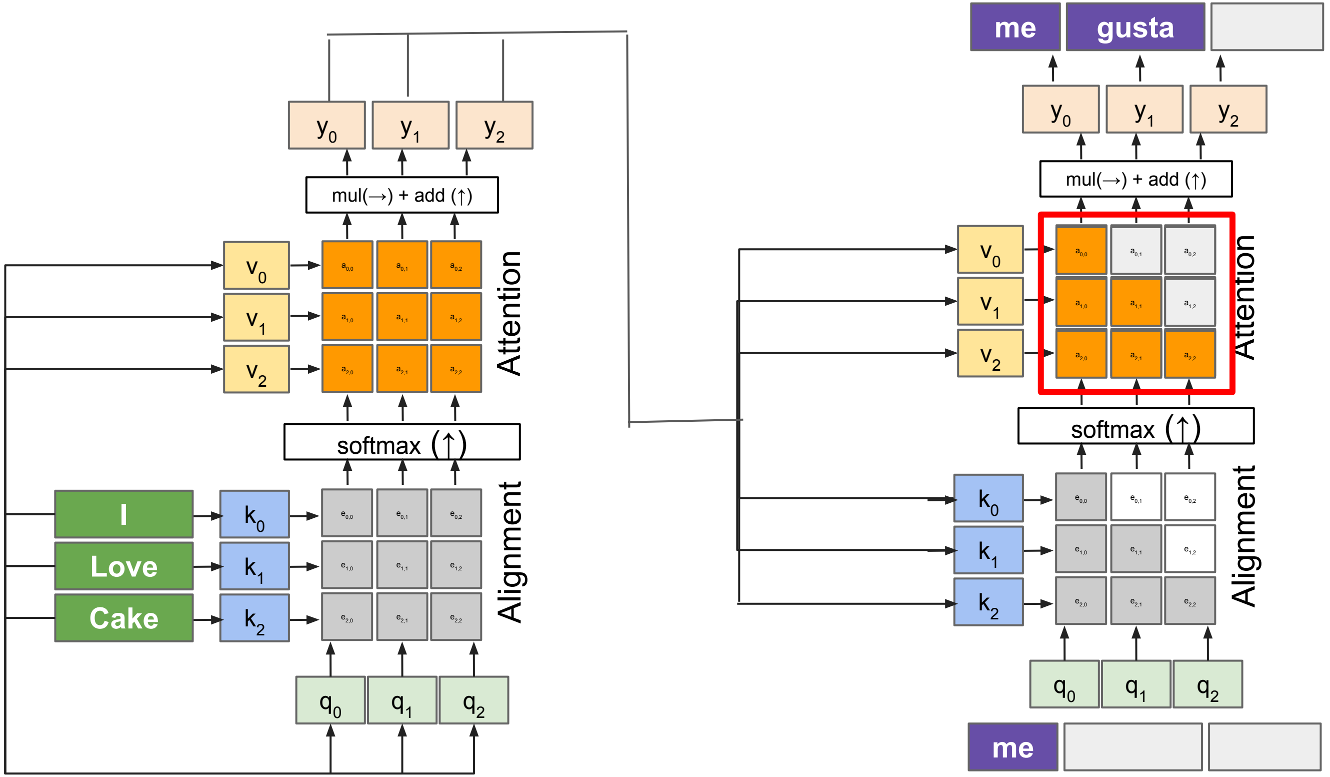


y_0
y_1
y_2
mul(→) + add (↑)
v_0
v_1
v_2
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$
Attention
softmax (↑)
I
Love
Cake
k_0
k_1
k_2
$e_{0,0}$ $e_{0,1}$ $e_{0,2}$
$e_{1,0}$ $e_{1,1}$ $e_{1,2}$
$e_{2,0}$ $e_{2,1}$ $e_{2,2}$
Alignment
q_0
q_1
q_2
y_0
y_1
y_2
mul(→) + add (↑)
v_0
v_1
v_2
$a_{0,0}$ $a_{0,1}$ $a_{0,2}$
$a_{1,0}$ $a_{1,1}$ $a_{1,2}$
$a_{2,0}$ $a_{2,1}$ $a_{2,2}$
Attention
softmax (↑)
k_0
k_1
k_2
$e_{0,0}$ $e_{0,1}$ $e_{0,2}$
$e_{1,0}$ $e_{1,1}$ $e_{1,2}$
$e_{2,0}$ $e_{2,1}$ $e_{2,2}$
Alignment
q_0
q_1
q_2

Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence



Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence

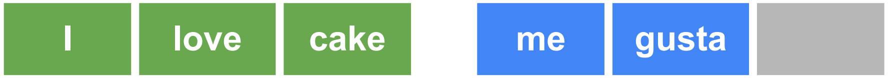


Which of the three options is GPT?

Which of the three options is GPT? **Decoder Only!**

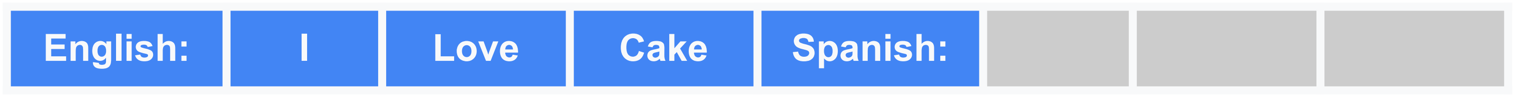
Which of the three options is GPT? Decoder Only!

Encoder-Decoder: Generate text based on previously generated text and the meaning of a separate sequence sequence



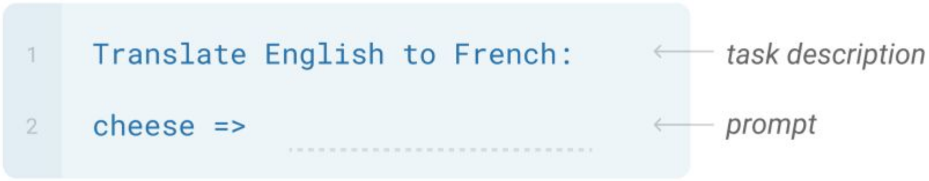
I love cake me gusta

Decoder Only: Generate text based on previously generated text



English: I Love Cake Spanish:

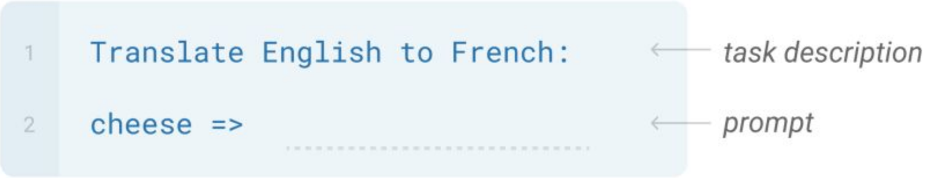
GPT-3



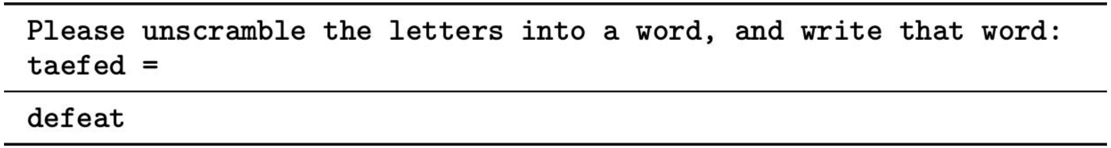
```
Translate English to French:
cheese =>
```

Brown et al “Language Models are Few-Shot Learners”

GPT-3

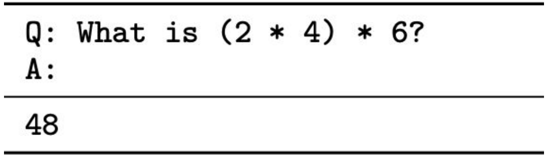


```
Translate English to French:
cheese =>
```



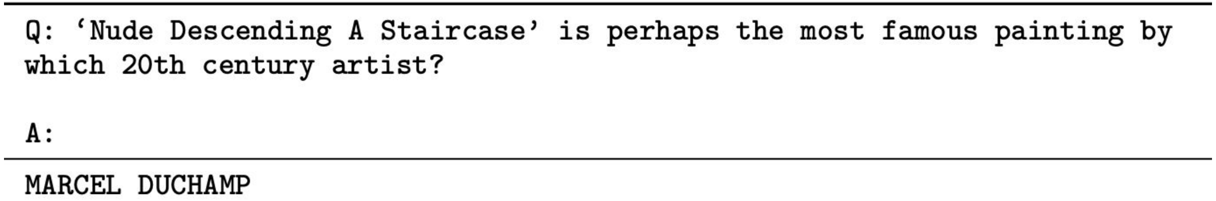
```
Please unscramble the letters into a word, and write that word:
taefed =
```

```
defeat
```



```
Q: What is (2 * 4) * 6?
A:
```

```
48
```

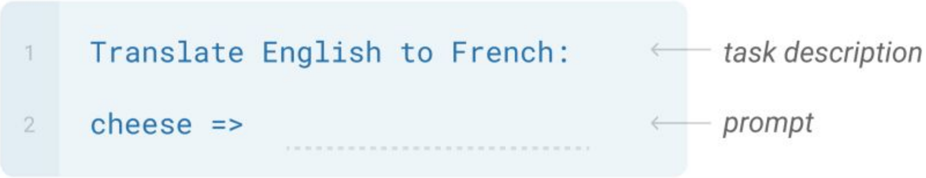


```
Q: ‘Nude Descending A Staircase’ is perhaps the most famous painting by
which 20th century artist?

A:
```

```
MARCEL DUCHAMP
```

Brown et al “Language Models are Few-Shot Learners”

GPT-3



```
Translate English to French:
cheese =>
```

task description

prompt

```
Please unscramble the letters into a word, and write that word:
taefed =
```

```
defeat
```

```
Q: What is (2 * 4) * 6?
A:
```

```
48
```

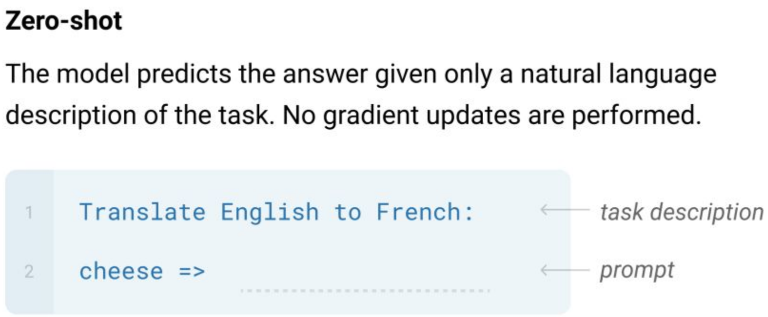
```
Q: ‘Nude Descending A Staircase’ is perhaps the most famous painting by
which 20th century artist?

A:
```

```
MARCEL DUCHAMP
```

Dataset
Common Crawl (filtered)
WebText2
Books1
Books2
Wikipedia

Brown et al “Language Models are Few-Shot Learners”

GPT-3



Zero-shot

The model predicts the answer given only a natural language description of the task. No gradient updates are performed.

```
Translate English to French:
cheese =>
```

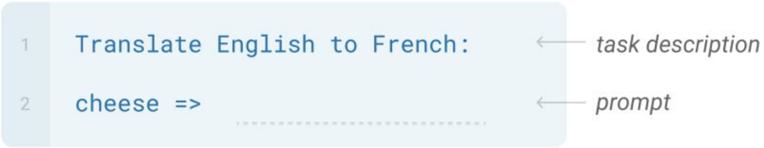
task description
prompt

Brown et al “Language Models are Few-Shot Learners”

GPT-3

Zero-shot

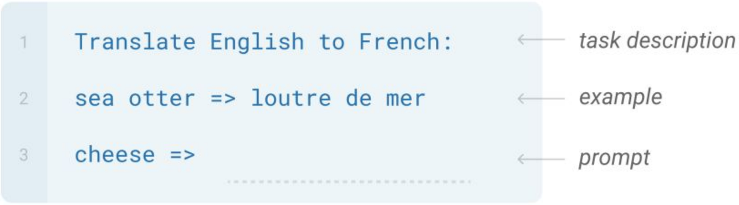
The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



```
Translate English to French:
cheese =>
```

One-shot

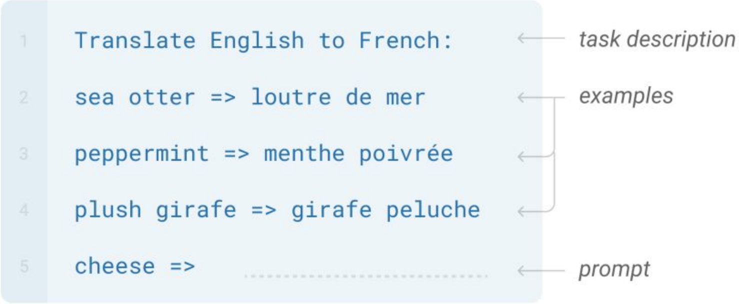
In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



```
Translate English to French:
sea otter => loutre de mer
cheese =>
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.



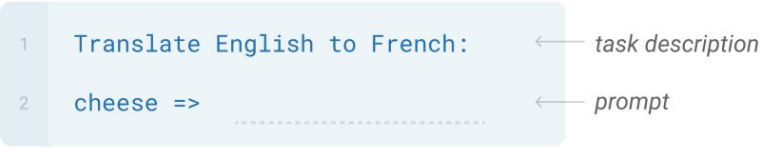
```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

Brown et al “Language Models are Few-Shot Learners”

GPT-3

Zero-shot

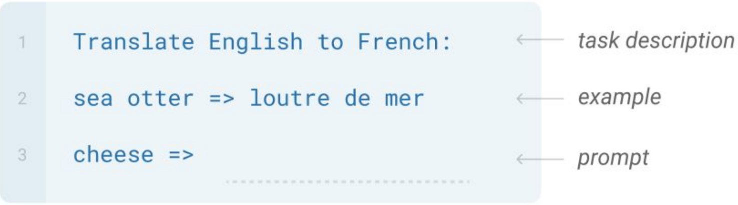
The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



```
Translate English to French:
cheese =>
```

One-shot

In addition to the task description, the model sees a single example of the task. No gradient updates are performed.

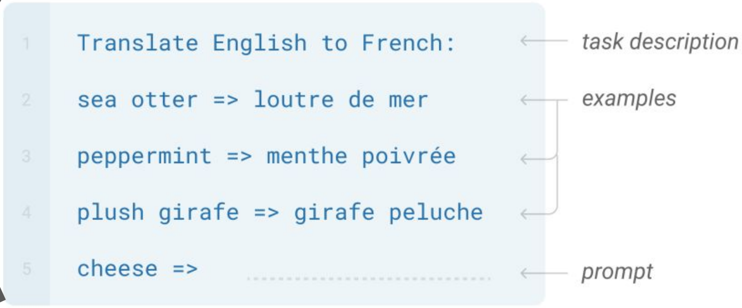


```
Translate English to French:
sea otter => loutre de mer
cheese =>
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

“Context”



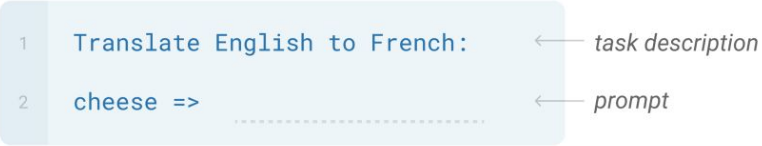
```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese =>
```

Brown et al “Language Models are Few-Shot Learners”

GPT-3

Zero-shot

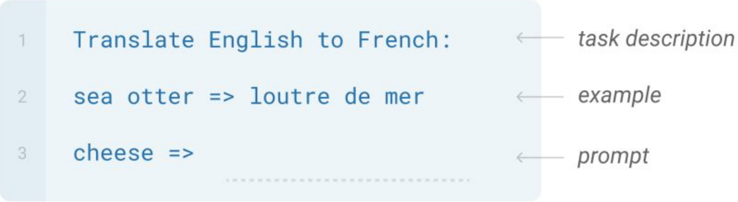
The model predicts the answer given only a natural language description of the task. No gradient updates are performed.



```
Translate English to French:
cheese => ..............................
```

One-shot

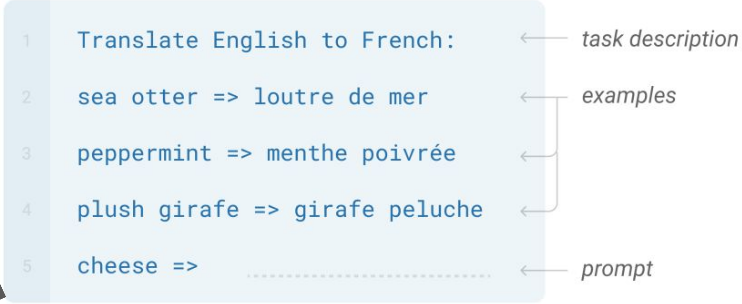
In addition to the task description, the model sees a single example of the task. No gradient updates are performed.



```
Translate English to French:
sea otter => loutre de mer
cheese => ..............................
```

Few-shot

In addition to the task description, the model sees a few examples of the task. No gradient updates are performed.

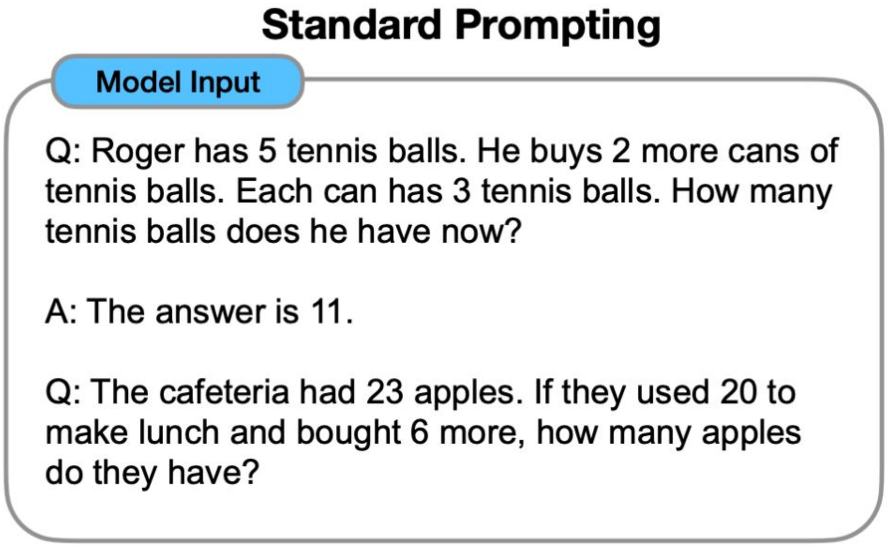


```
Translate English to French:
sea otter => loutre de mer
peppermint => menthe poivrée
plush girafe => girafe peluche
cheese => ..............................
```

“Context”

In-Context Learning

Brown et al “Language Models are Few-Shot Learners”



Standard Prompting

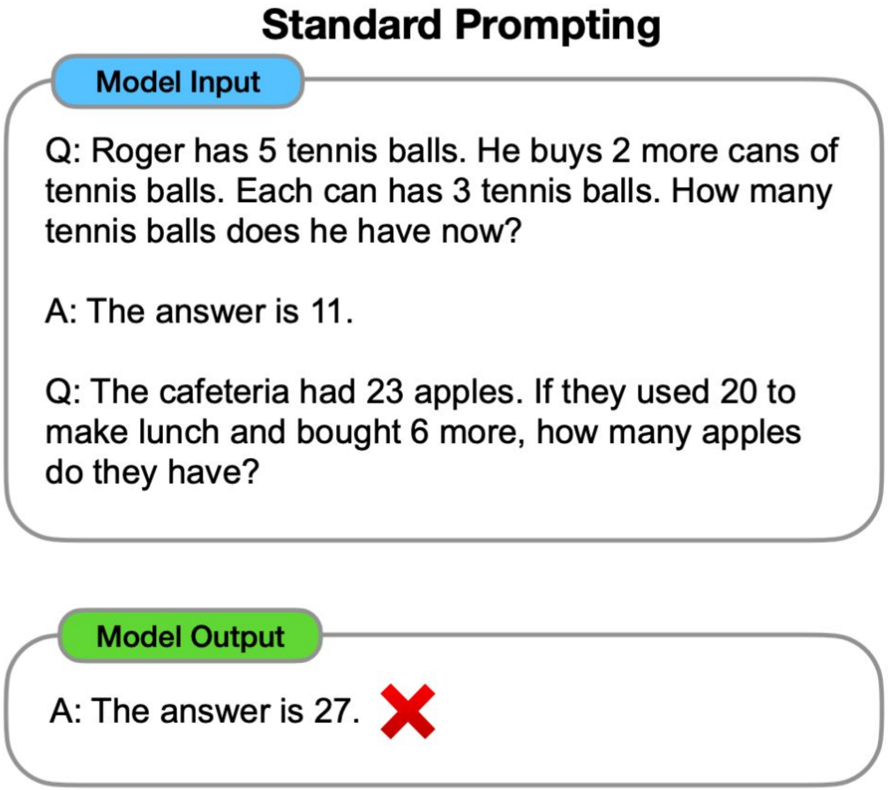
Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Wei et al “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”



Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

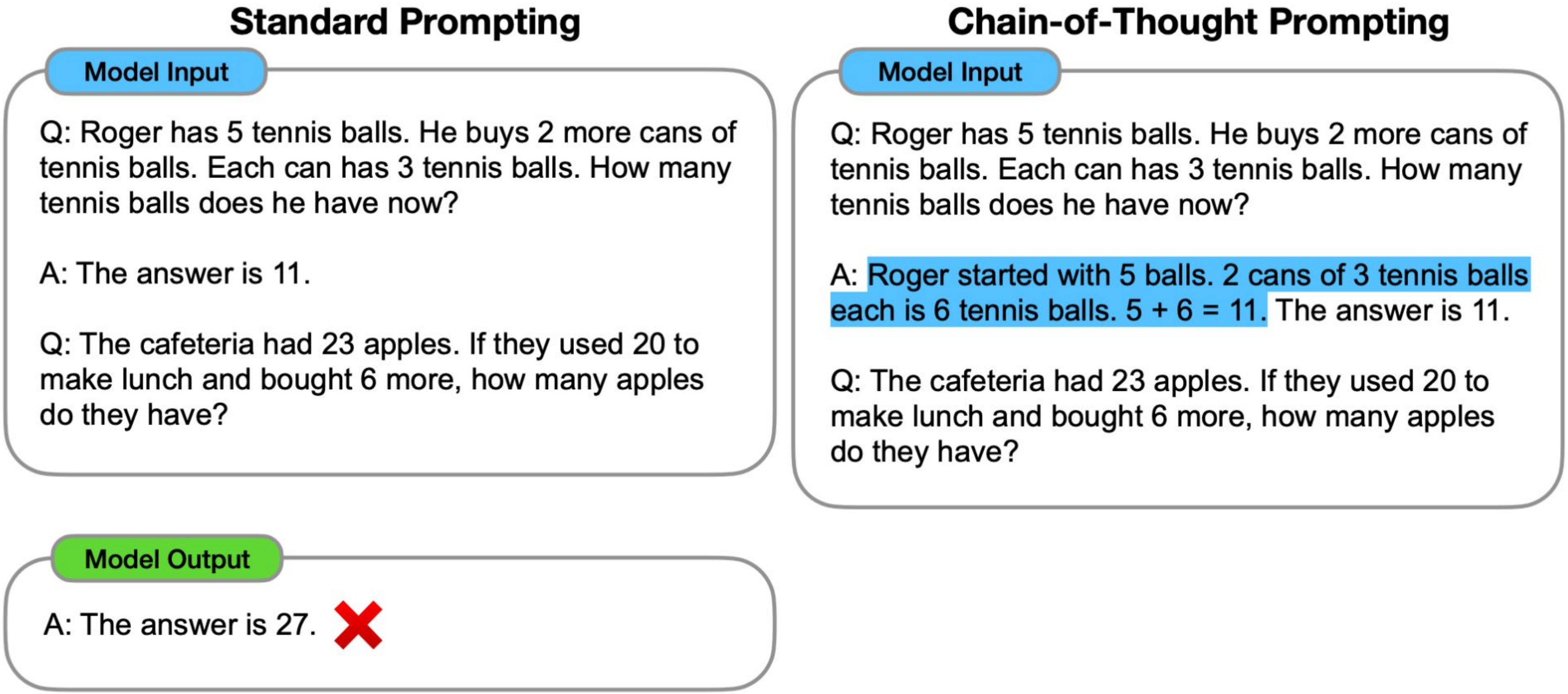
A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Wei et al “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”



Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

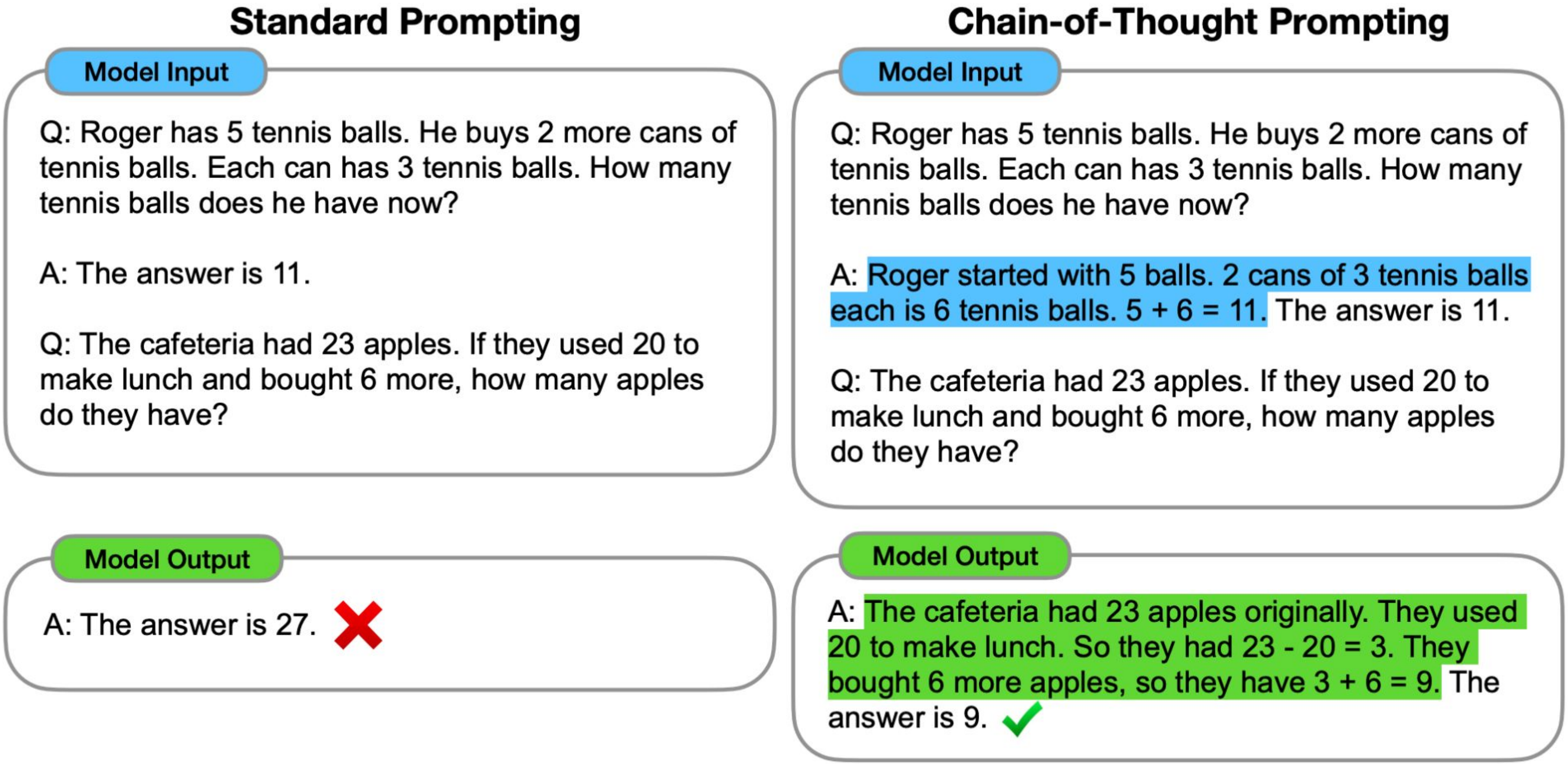
Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Wei et al “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”



Standard Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The answer is 27. ✗

Chain-of-Thought Prompting

Model Input

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

Model Output

A: The cafeteria had 23 apples originally. They used 20 to make lunch. So they had 23 - 20 = 3. They bought 6 more apples, so they have 3 + 6 = 9. The answer is 9. ✓

Wei et al “Chain-of-Thought Prompting Elicits Reasoning in Large Language Models”

Language Models

Encoders vs Decoders vs Encoder-Decoder Models

Prompting (zero-shot, in-context, chain-of-thought)

Vision + Language Models

CLIP training + inference

Results + Robustness

My prior work

Vision + Language

Vision + Language

Text pretraining task: Given previous words, pick the next word

Vision + Language

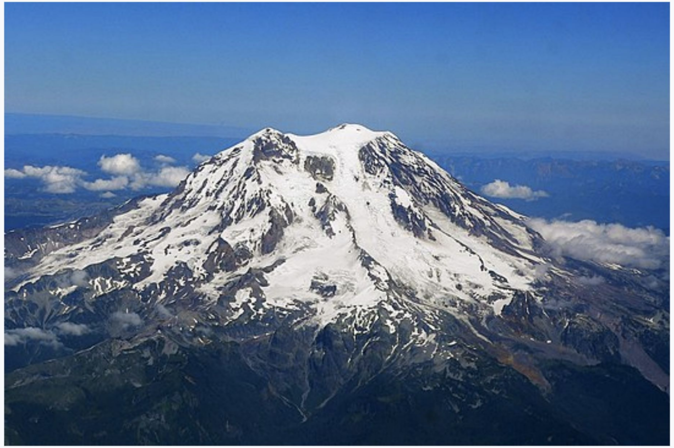
Text pretraining task: Given previous words, pick the next word

Text + Vision: Where can we get large amounts of image and text data?

Vision + Language

Text pretraining task: Given previous words, pick the next word

Text + Vision: Where can we get large amounts of image and text data?

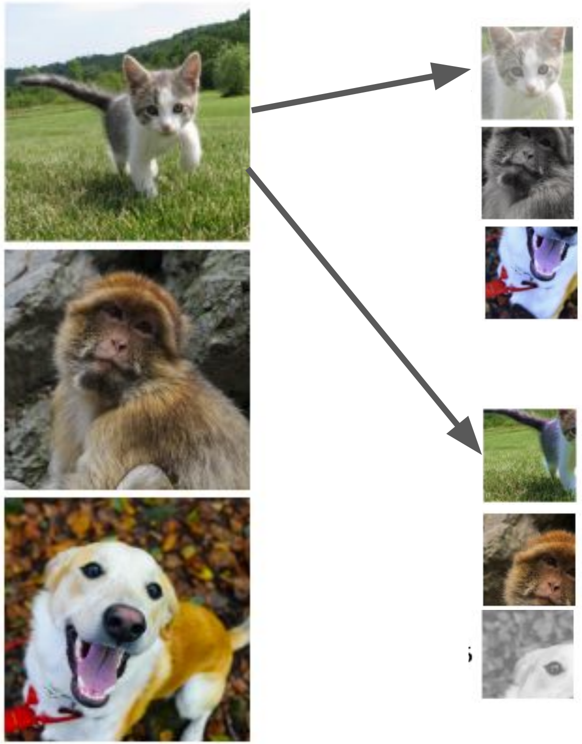


The western slope of Mount Rainier in 2005

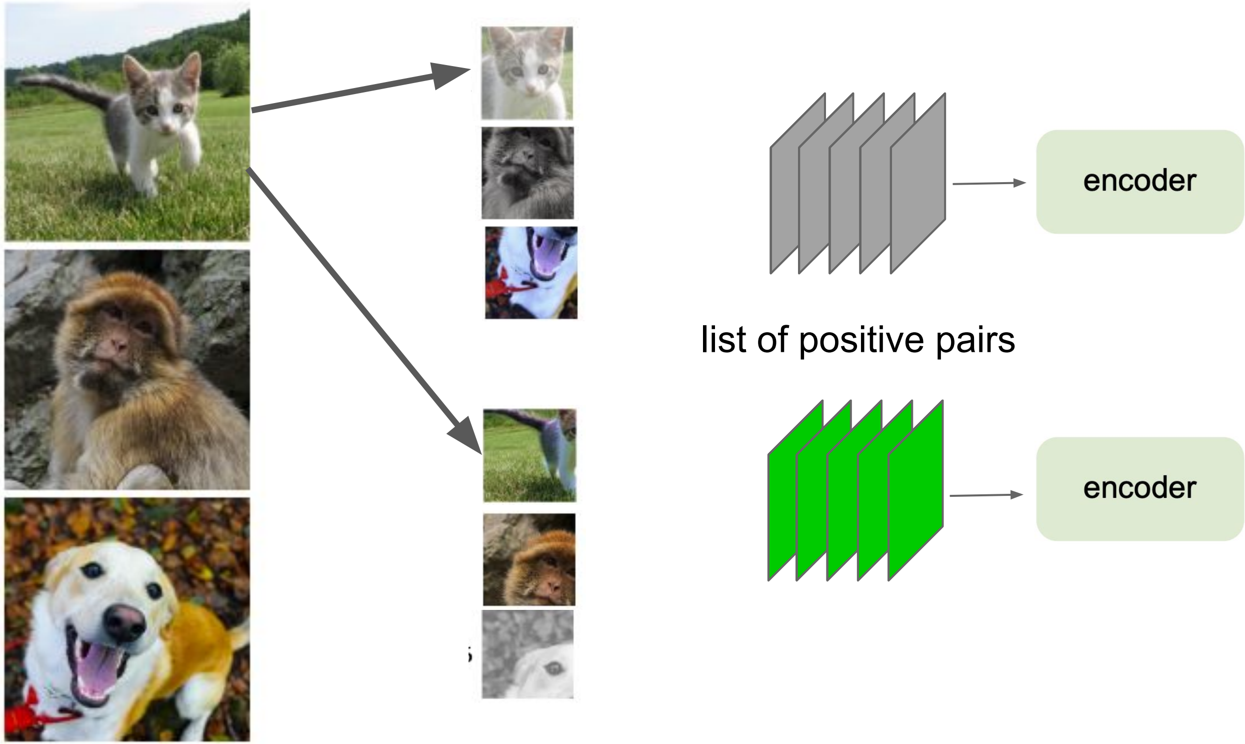
https://en.wikipedia.org/wiki/Mount_Rainier

SimCLR

SimCLR



SimCLR

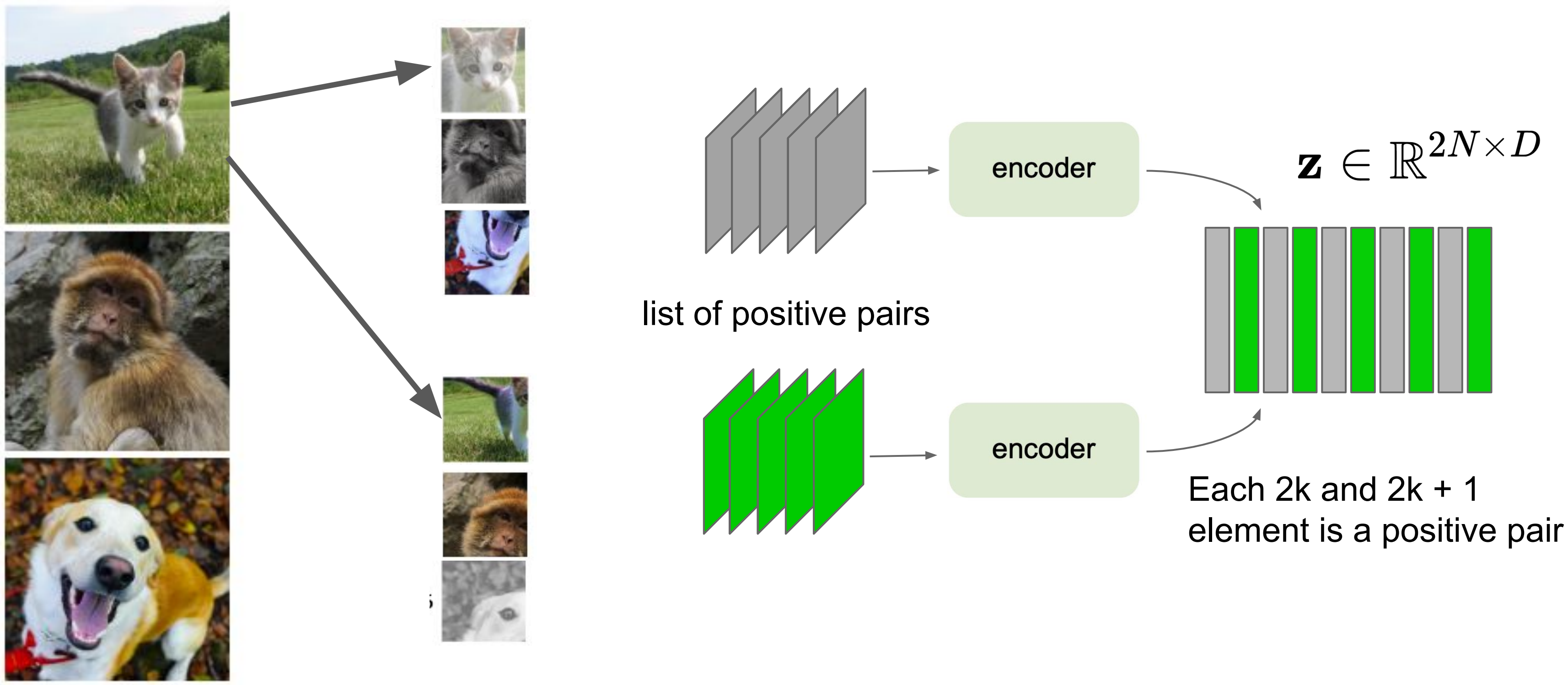


encoder

list of positive pairs

encoder

SimCLR



$$\mathbf{z} \in \mathbb{R}^{2N \times D}$$

encoder

list of positive pairs

encoder

Each 2k and 2k + 1 element is a positive pair

CLIP Training

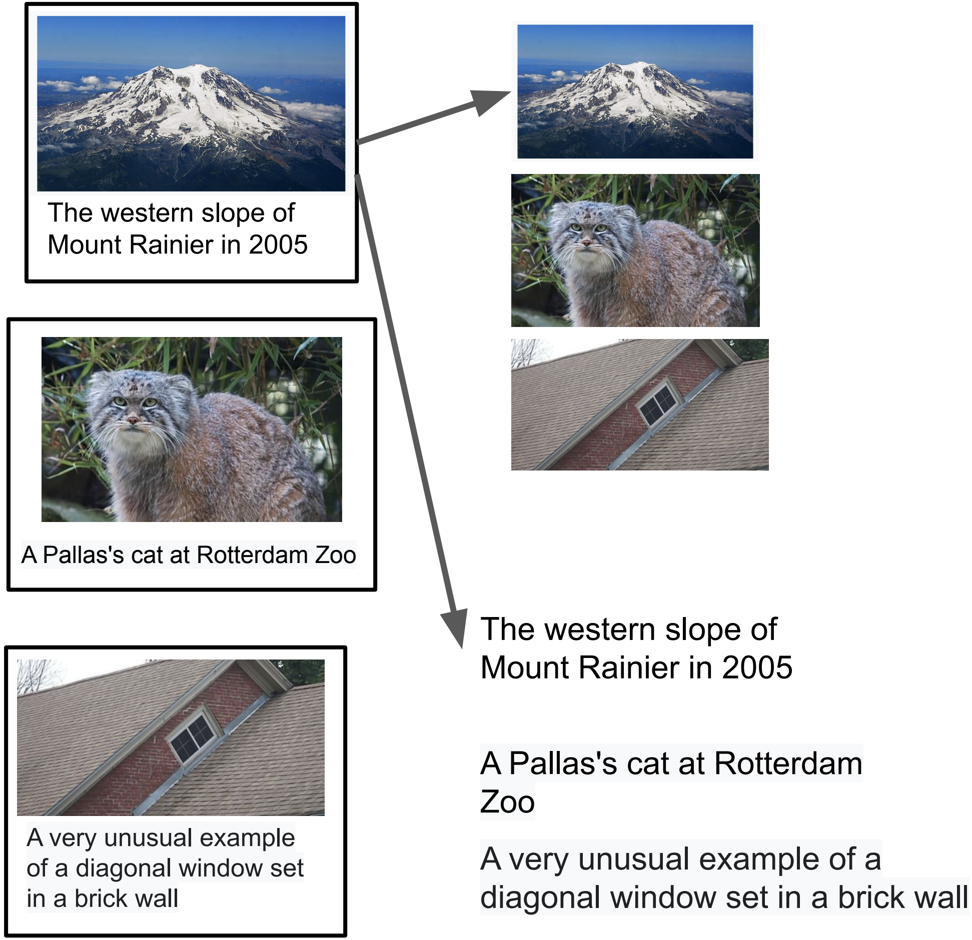
CLIP Training

The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

CLIP Training



The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

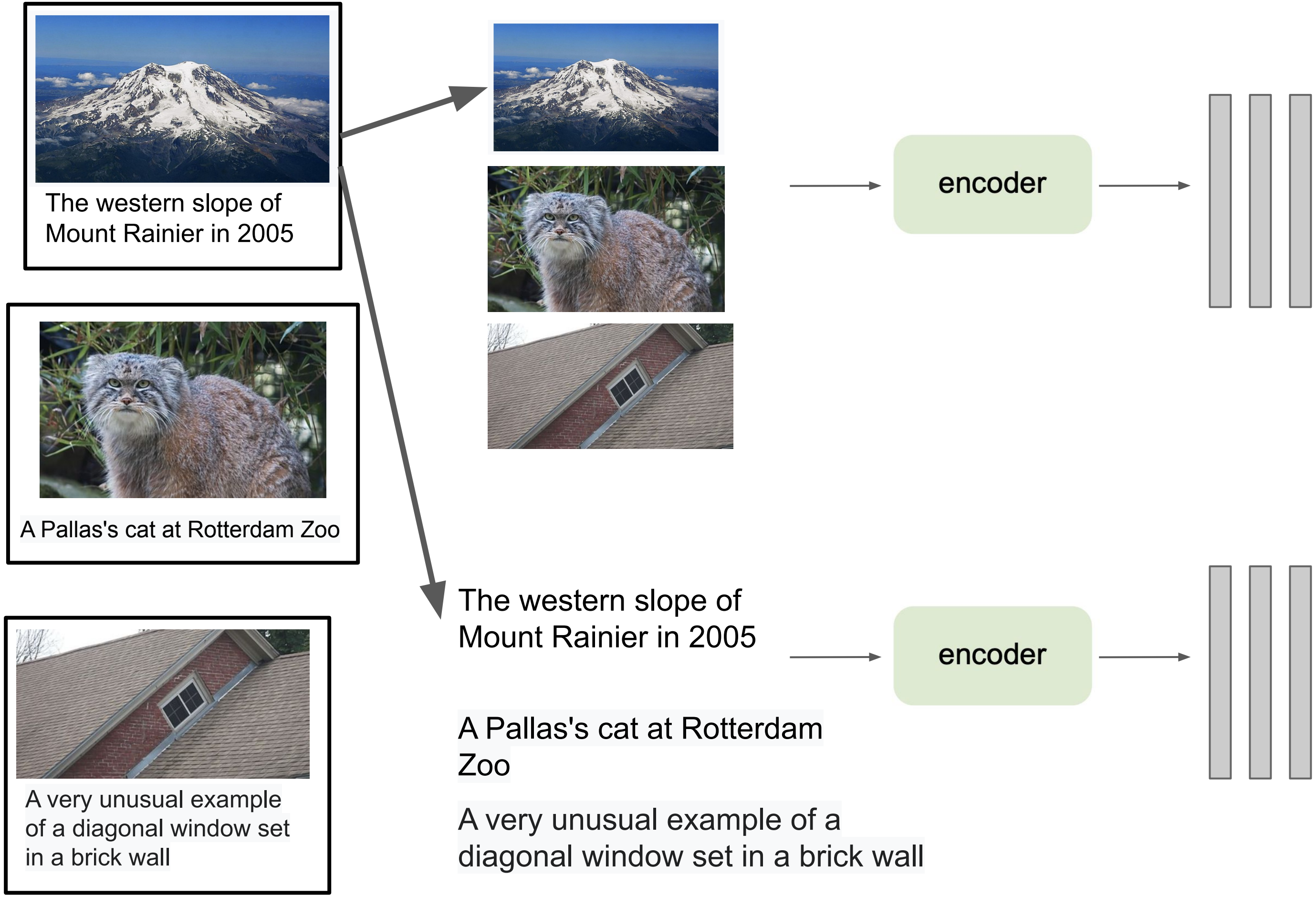
A very unusual example of a diagonal window set in a brick wall

The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

CLIP Training



The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

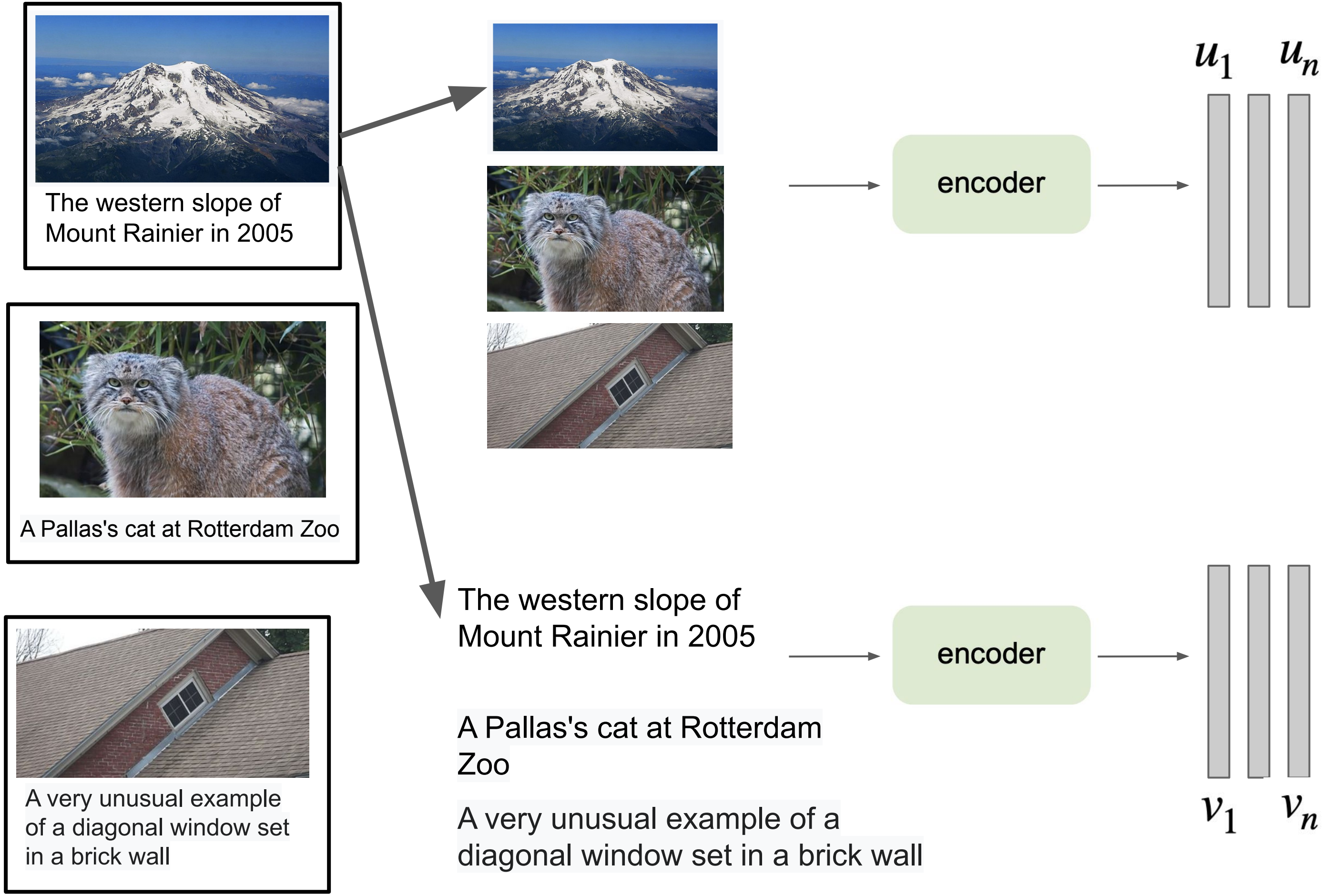
A very unusual example of a diagonal window set in a brick wall

The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

CLIP Training



The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

encoder

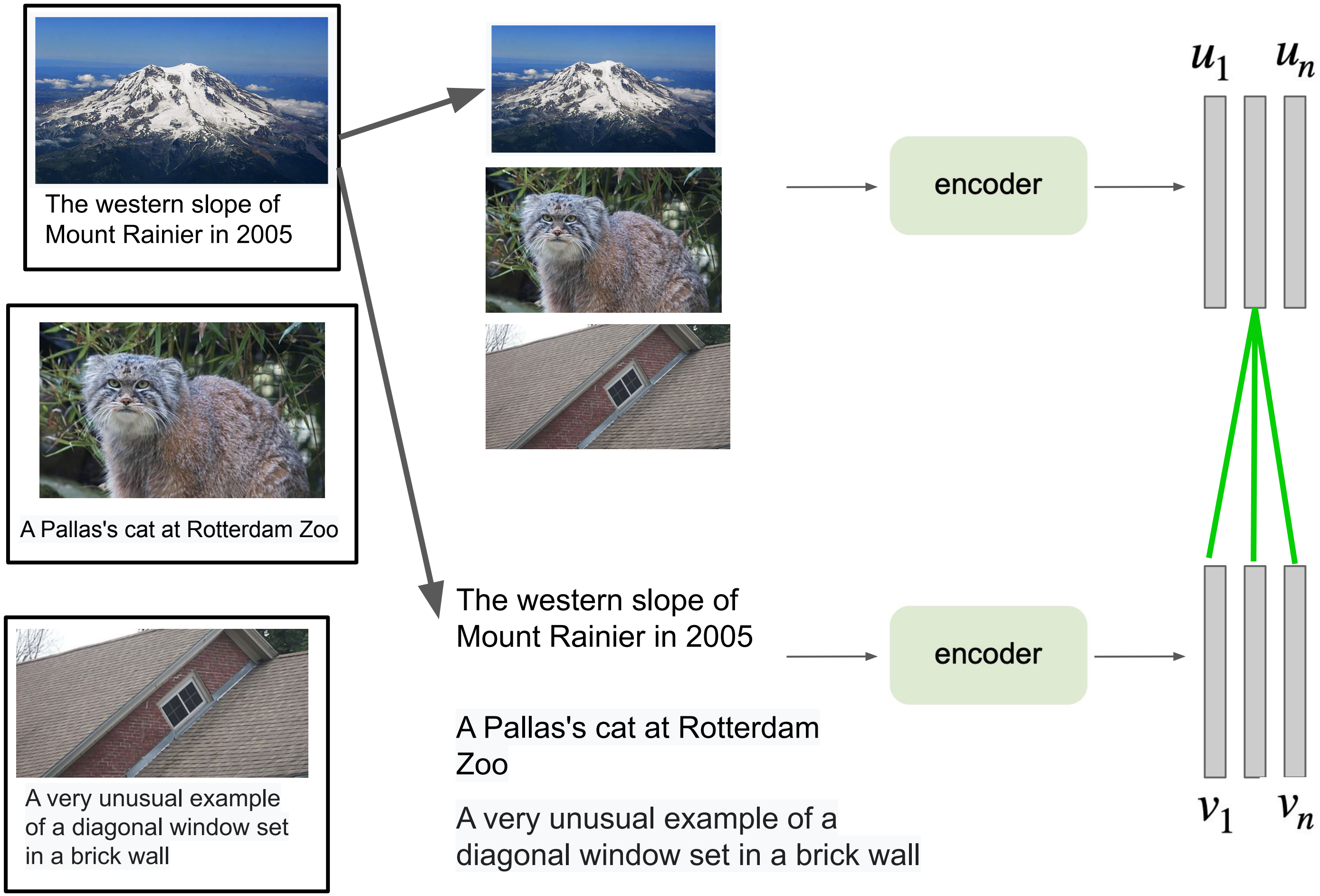
The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

encoder

CLIP Training



The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

u_1 u_n

encoder

The western slope of Mount Rainier in 2005

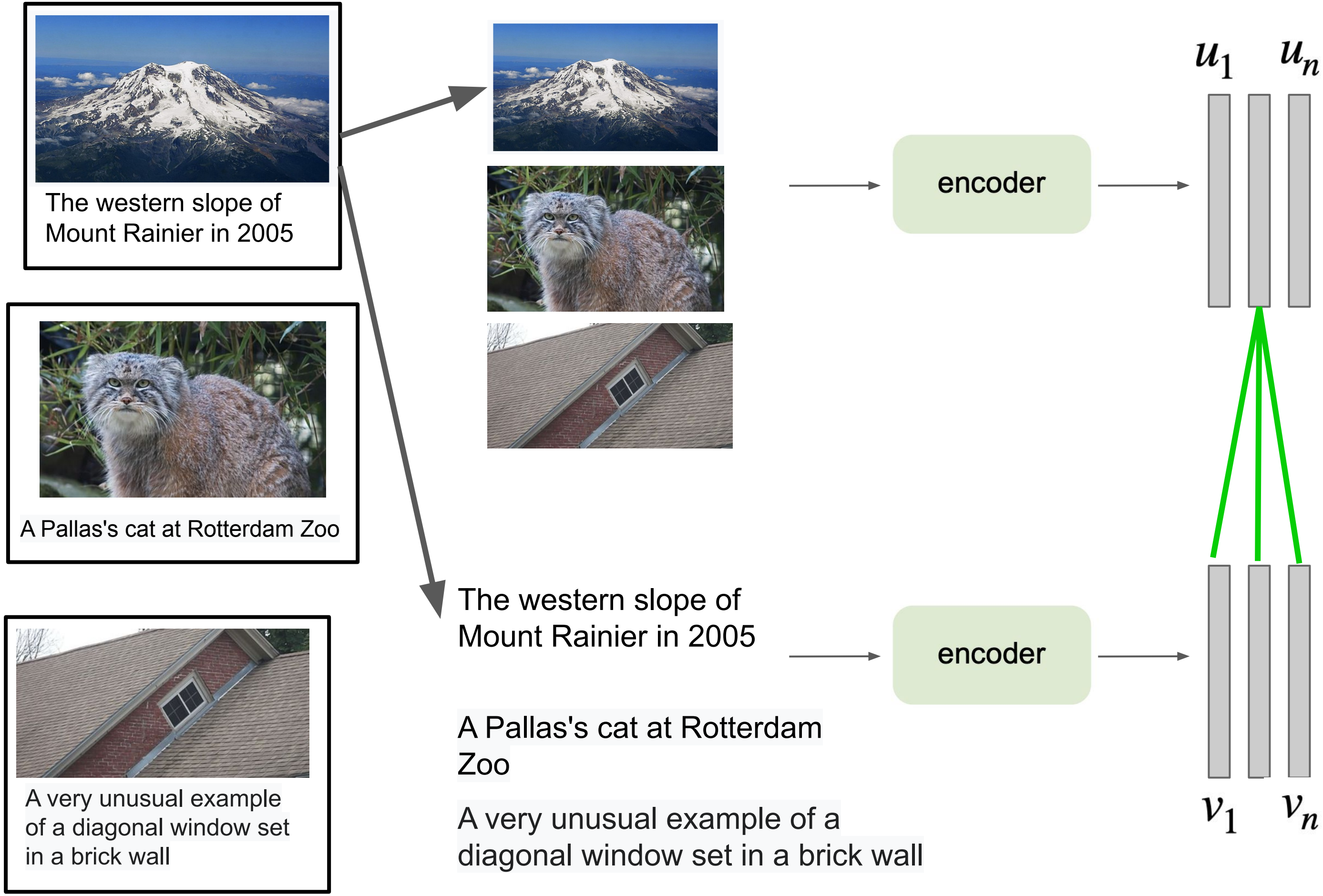
A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

encoder

v_1 v_n

CLIP Training



The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

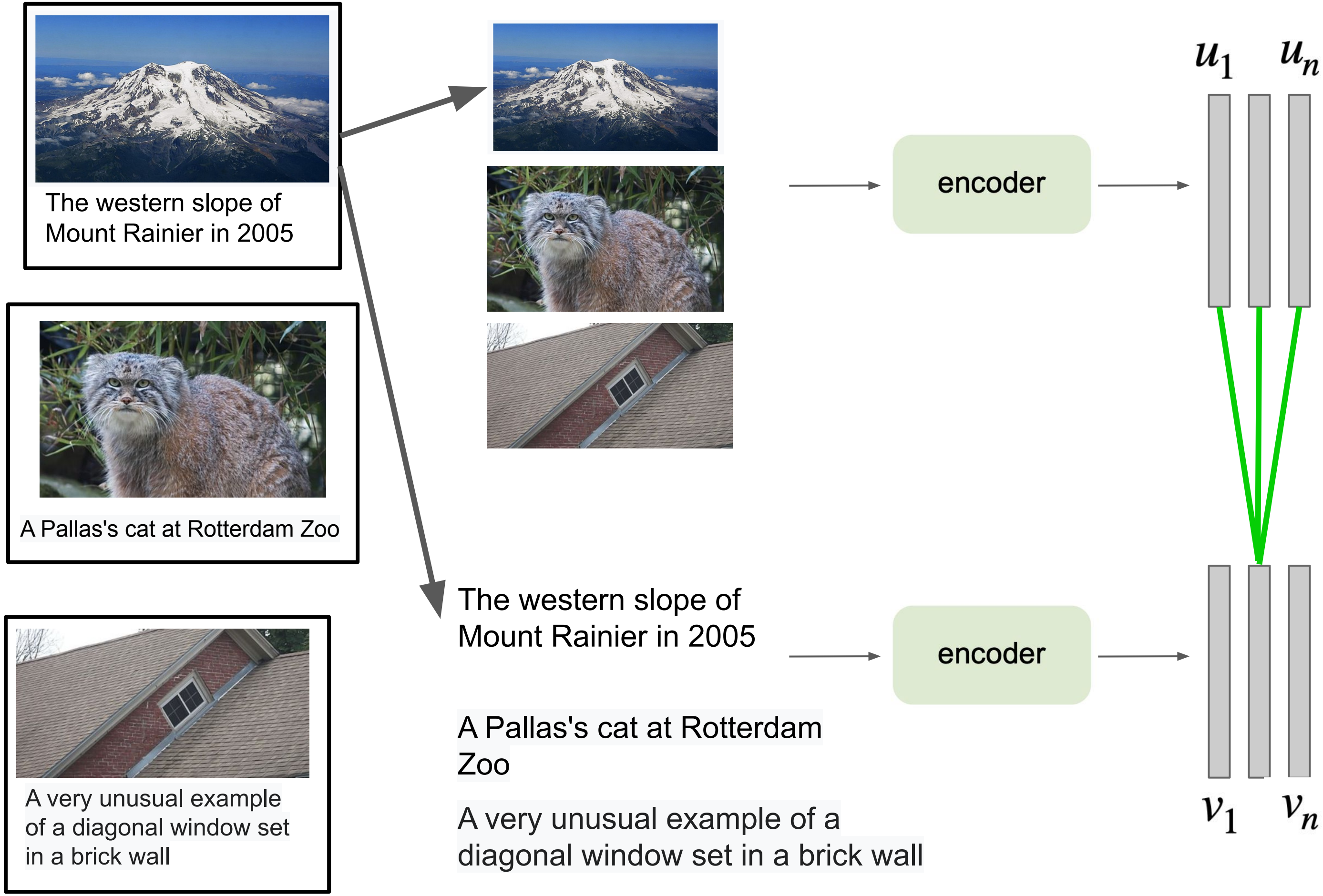
The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

$$\sum_{i=1}^{n} -\log\left(\frac{e^{\langle u_i, v_i\rangle}}{\sum_{j=1}^{n} e^{\langle u_i, v_j\rangle}}\right)$$

CLIP Training



The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

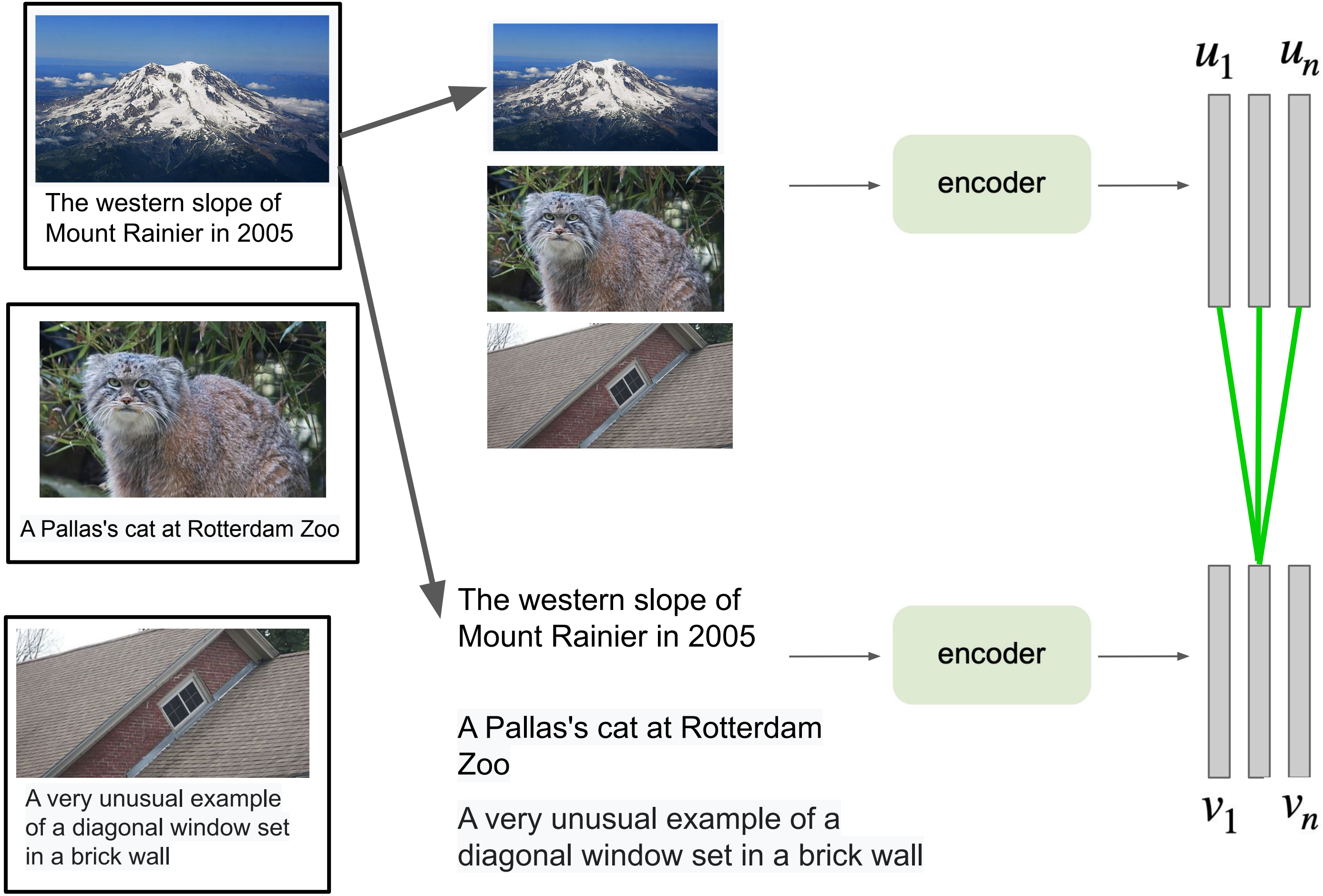
The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

$$\sum_{i=1}^{n} -\log\left(\frac{e^{\langle u_i, v_i\rangle}}{\sum_{j=1}^{n} e^{\langle u_i, v_j\rangle}}\right)$$

CLIP Training



The western slope of Mount Rainier in 2005

A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

The western slope of Mount Rainier in 2005

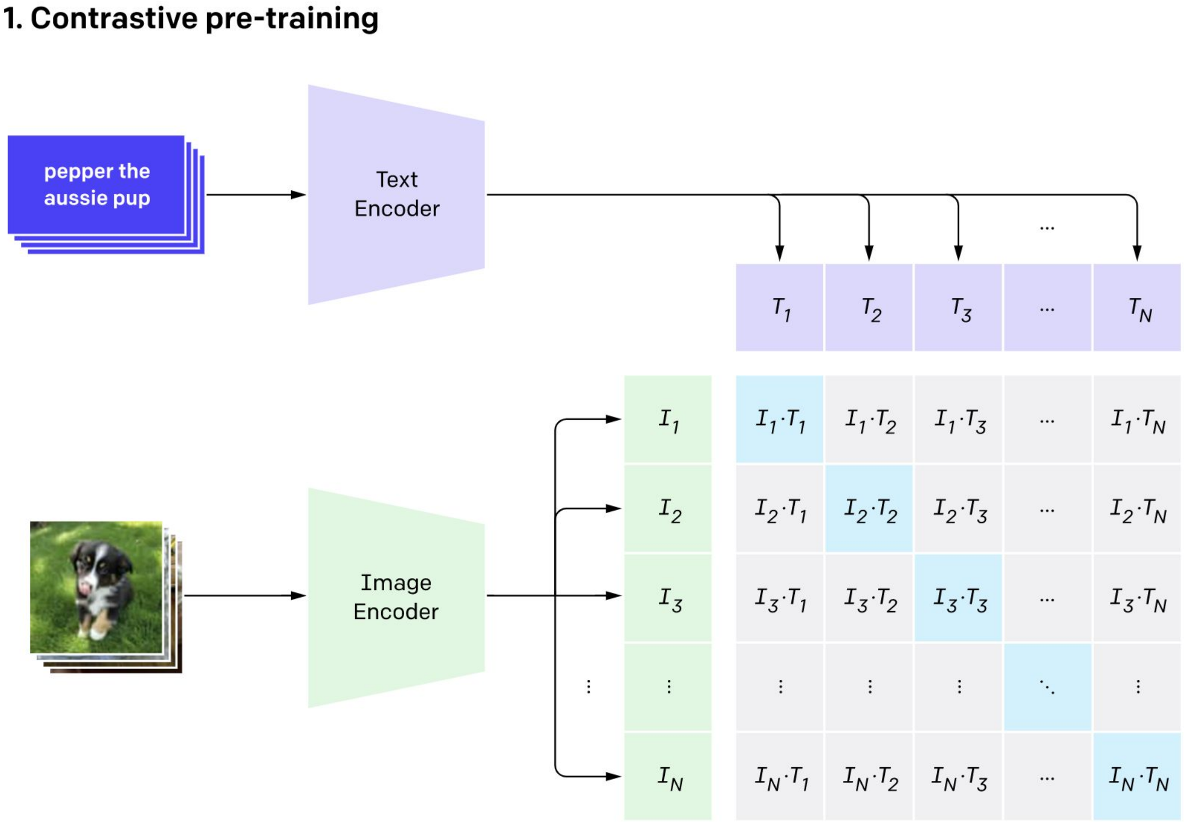
A Pallas's cat at Rotterdam Zoo

A very unusual example of a diagonal window set in a brick wall

$$\sum_{i=1}^{n} -\log\left(\frac{e^{\langle u_i,v_i\rangle}}{\sum_{j=1}^{n} e^{\langle u_i,v_j\rangle}}\right)$$

$$+\sum_{i=1}^{n} -\log\left(\frac{e^{\langle u_i,v_i\rangle}}{\sum_{j=1}^{n} e^{\langle u_j,v_i\rangle}}\right)$$

CLIP Training (from the CLIP paper)



1. Contrastive pre-training

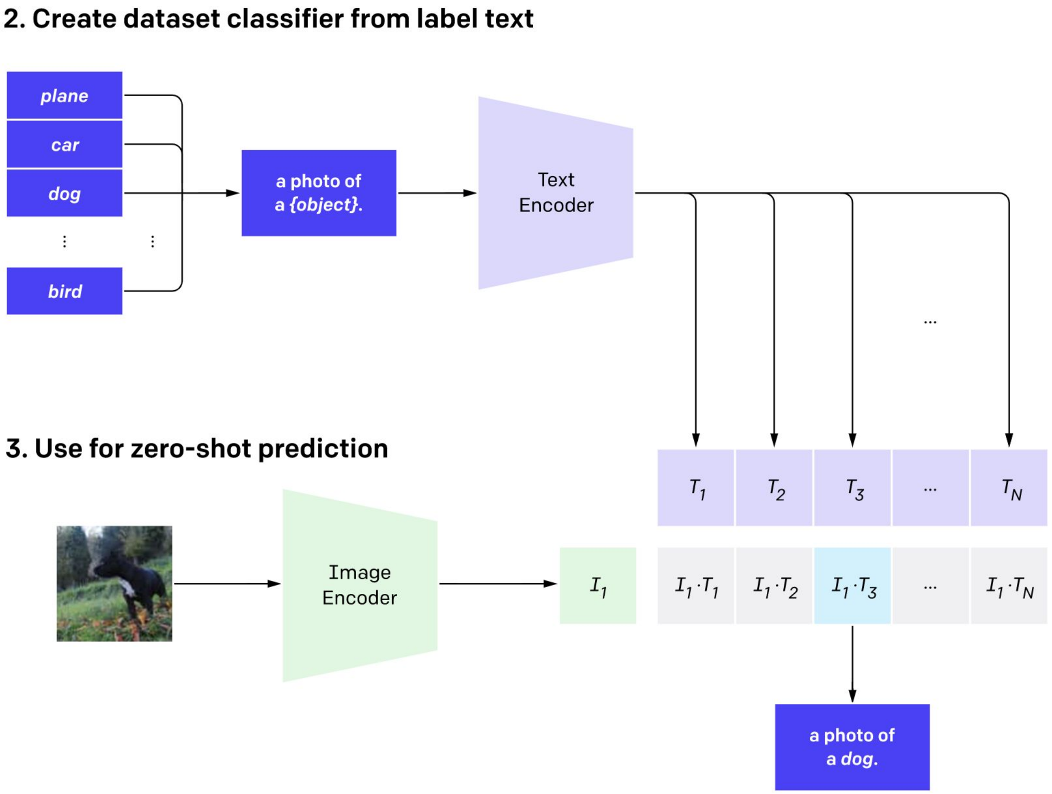
	T_1	T_2	T_3	...	T_N
I_1	$I_1 \cdot T_1$	$I_1 \cdot T_2$	$I_1 \cdot T_3$	...	$I_1 \cdot T_N$
I_2	$I_2 \cdot T_1$	$I_2 \cdot T_2$	$I_2 \cdot T_3$	...	$I_2 \cdot T_N$
I_3	$I_3 \cdot T_1$	$I_3 \cdot T_2$	$I_3 \cdot T_3$	...	$I_3 \cdot T_N$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
I_N	$I_N \cdot T_1$	$I_N \cdot T_2$	$I_N \cdot T_3$	...	$I_N \cdot T_N$

Radford et al "Learning Transferable Visual Models From Natural Language Supervision"

CLIP Inference (from the CLIP paper)

Radford et al “Learning Transferable Visual Models From Natural Language Supervision”

CLIP Inference (from the CLIP paper)



Radford et al “Learning Transferable Visual Models From Natural Language Supervision”

Image Classification on ImageNet

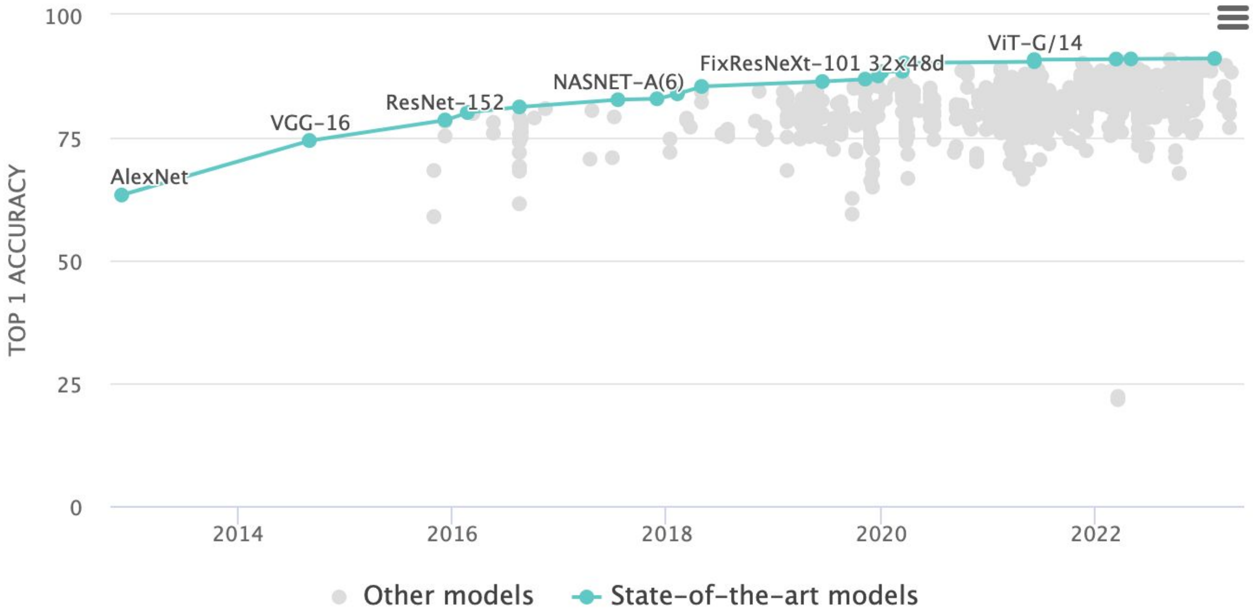
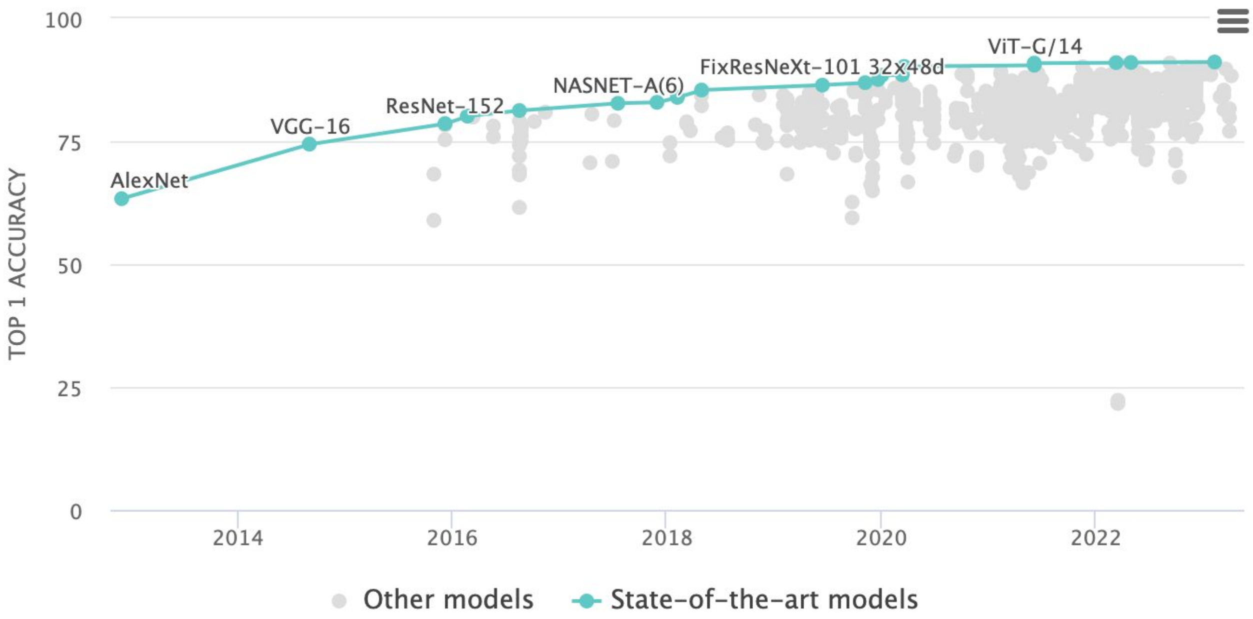


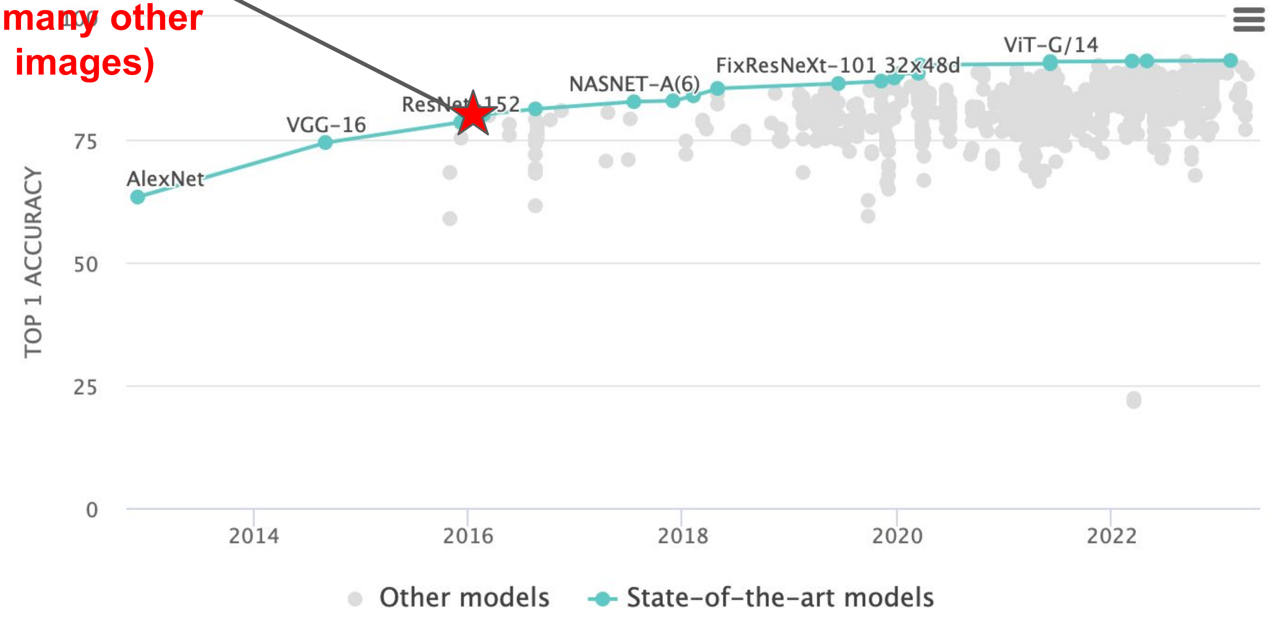
Image Classification on ImageNet



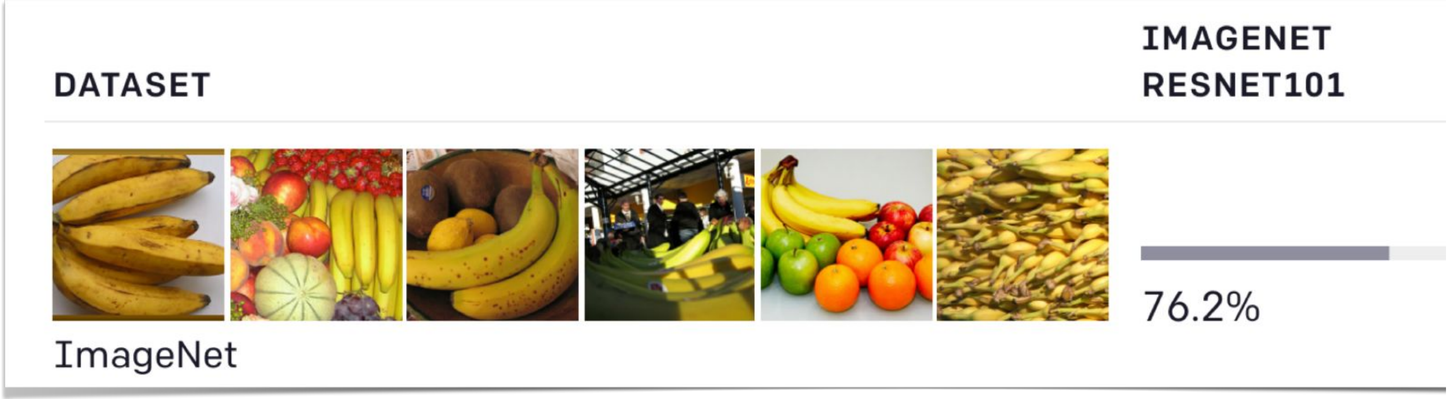
After training on ~1,000,000 labeled ImageNet train images

Image Classification on ImageNet

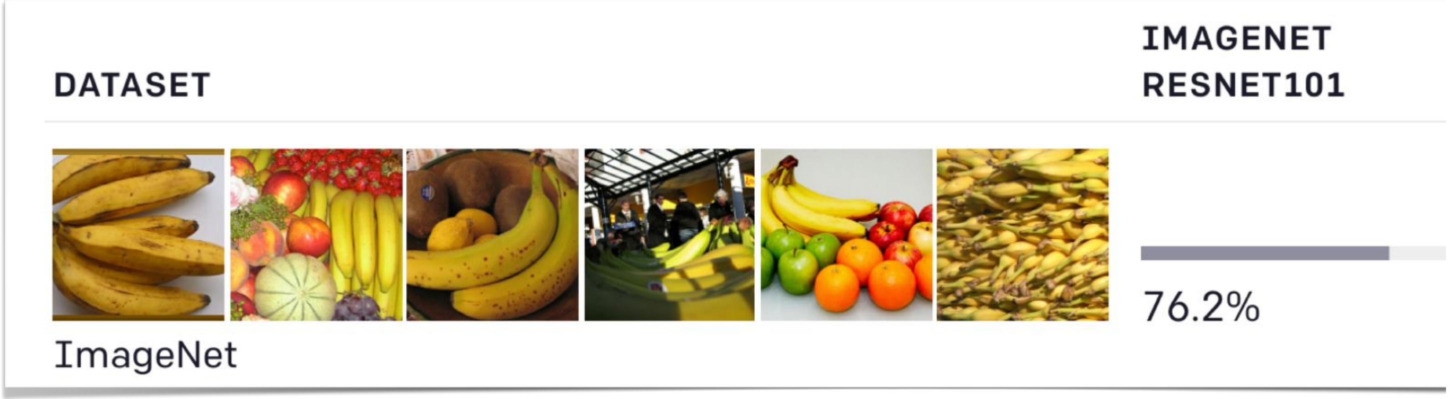
After training on 0 labeled ImageNet train images (but many other "unlabeled" images)



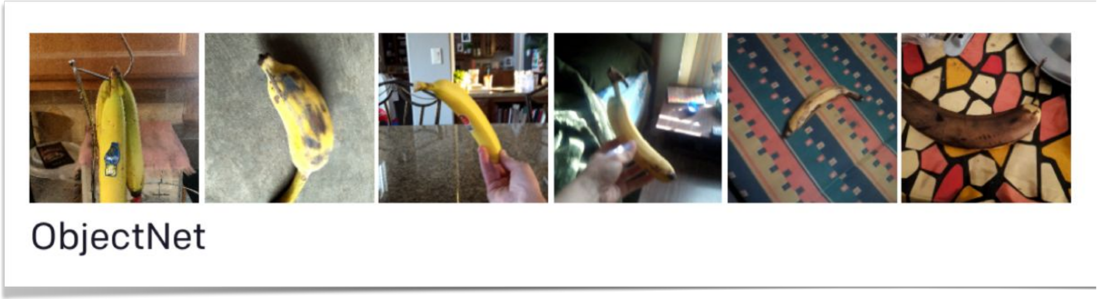
After training on ~1,000,000 labeled ImageNet train images



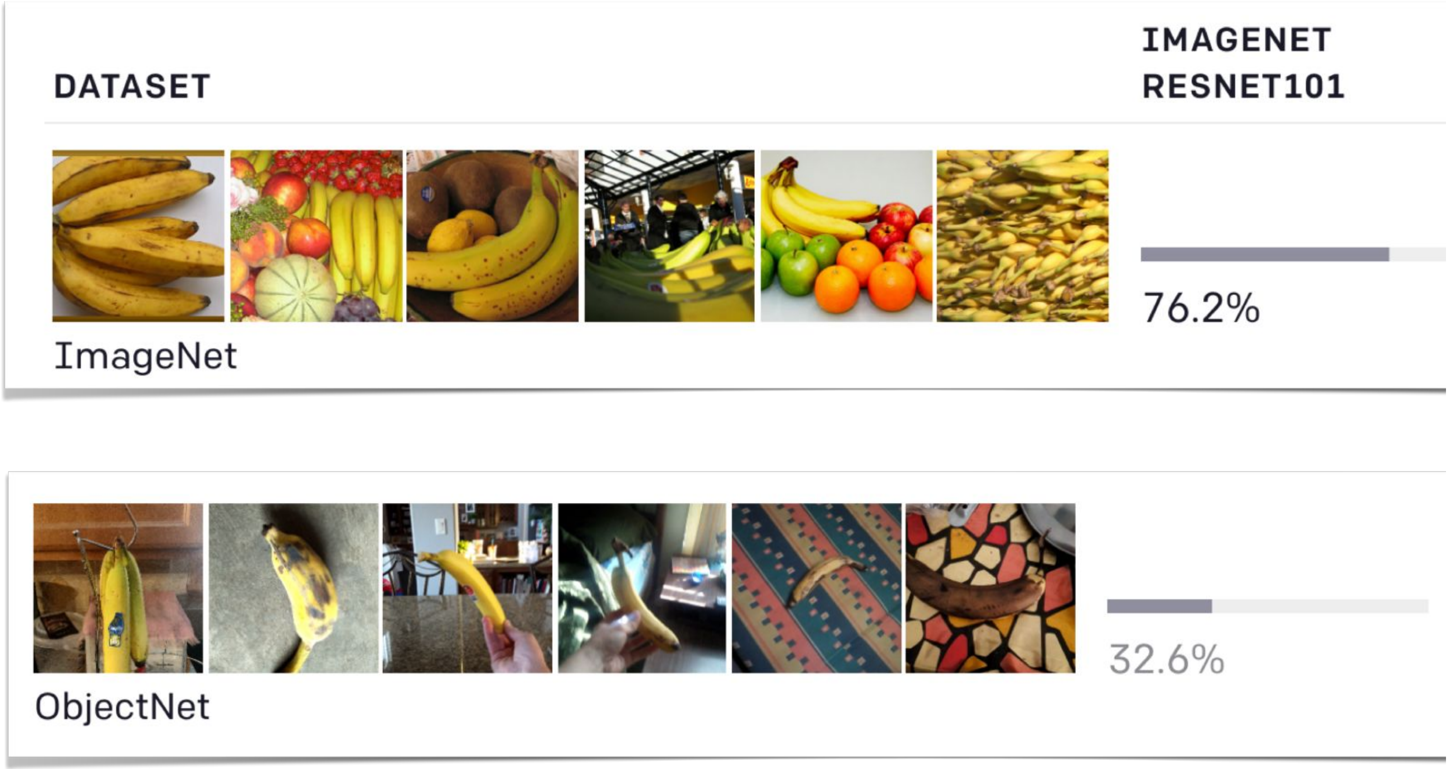
DATASET	IMAGENET RESNET101
ImageNet	76.2%



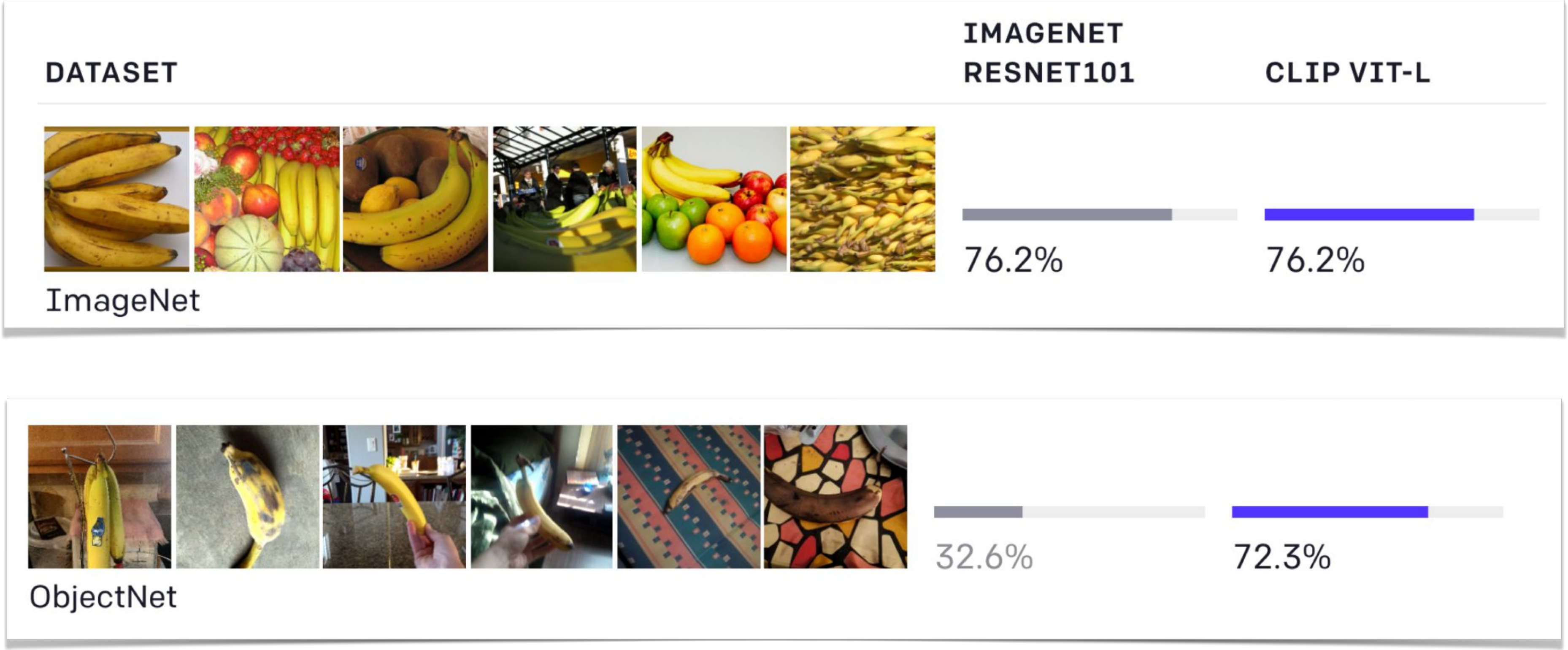
DATASET	IMAGENET RESNET101
ImageNet	76.2%

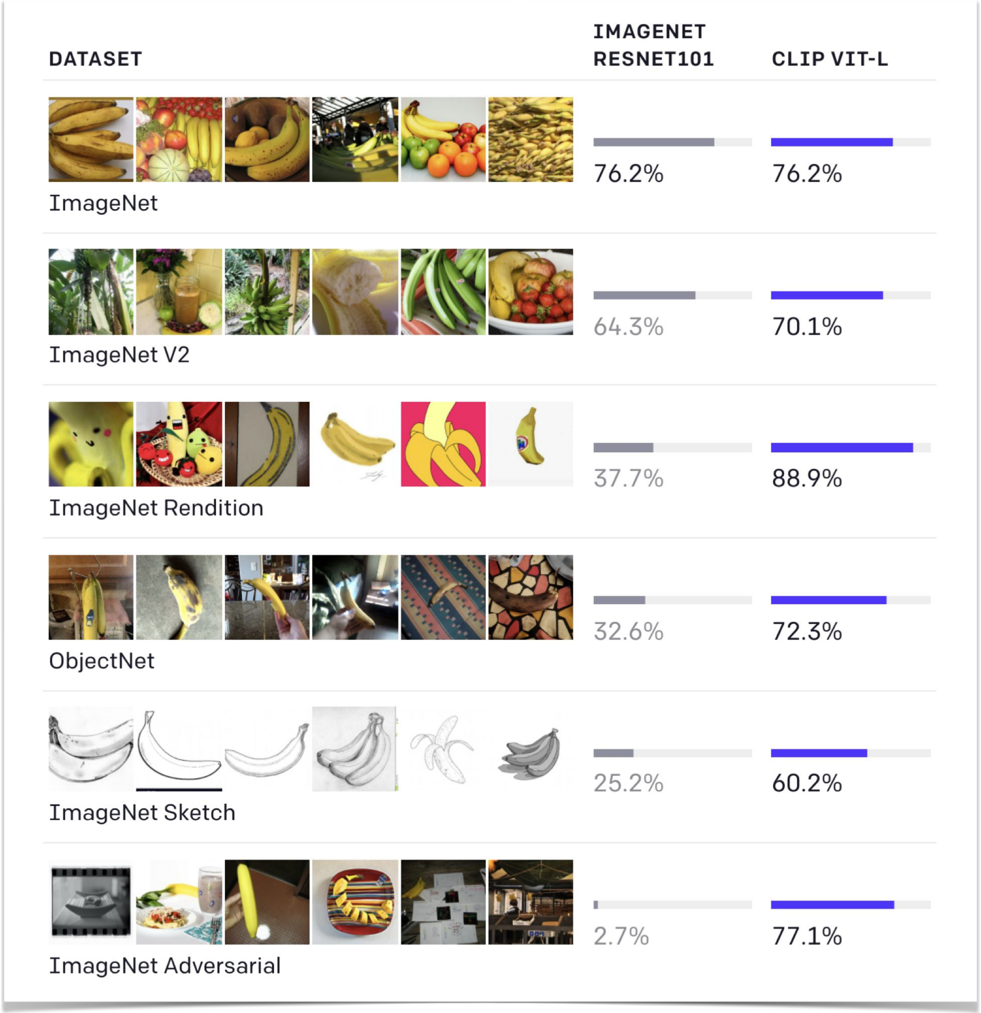


ObjectNet



DATASET	IMAGENET RESNET101
ImageNet	76.2%
ObjectNet	32.6%





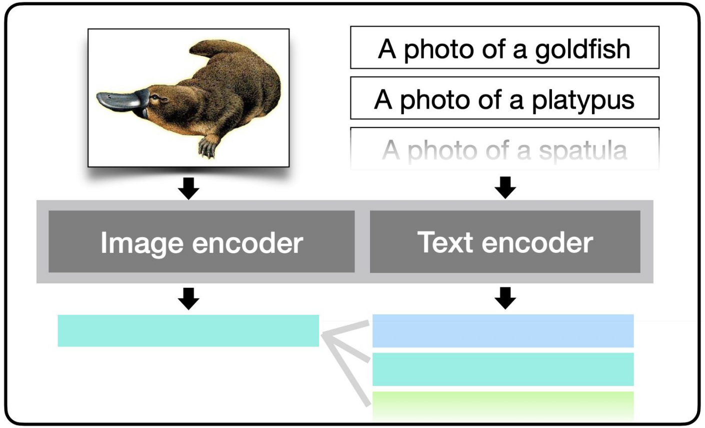
DATASET	IMAGENET RESNET101	CLIP VIT-L
ImageNet	76.2%	76.2%
ImageNet V2	64.3%	70.1%
ImageNet Rendition	37.7%	88.9%
ObjectNet	32.6%	72.3%
ImageNet Sketch	25.2%	60.2%
ImageNet Adversarial	2.7%	77.1%

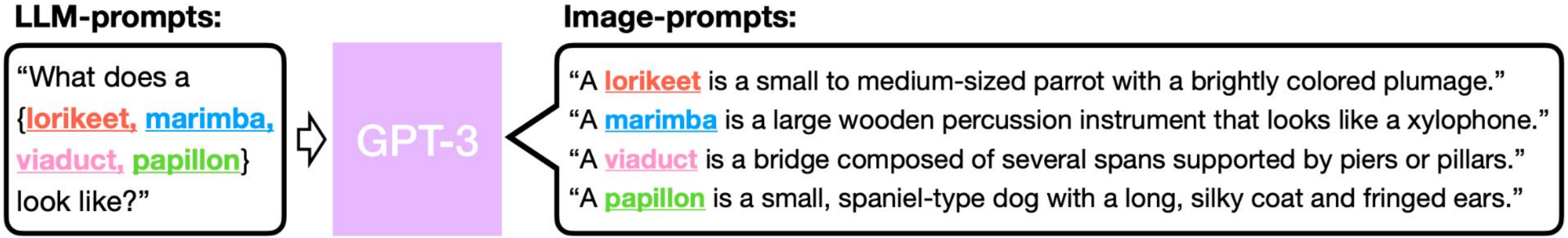
AI-Chaining

(aka a plug for my own past work)

AI-Chaining

(aka a plug for my own past work)





LLM-prompts:

“What does a {lorikeet, marimba, viaduct, papillon} look like?”

Image-prompts:

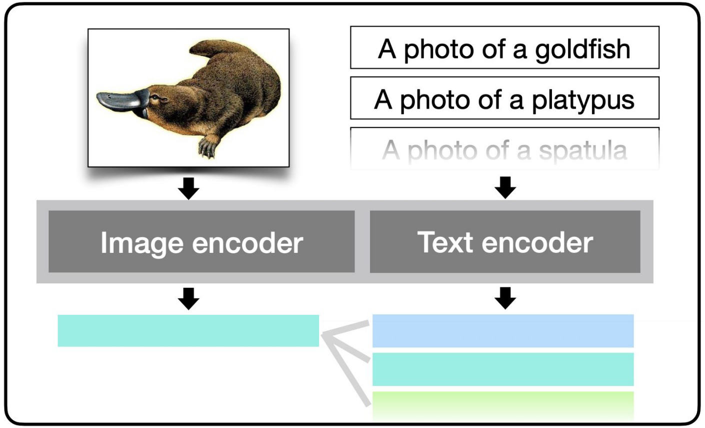
“A lorikeet is a small to medium-sized parrot with a brightly colored plumage.”
“A marimba is a large wooden percussion instrument that looks like a xylophone.”
“A viaduct is a bridge composed of several spans supported by piers or pillars.”
“A papillon is a small, spaniel-type dog with a long, silky coat and fringed ears.”

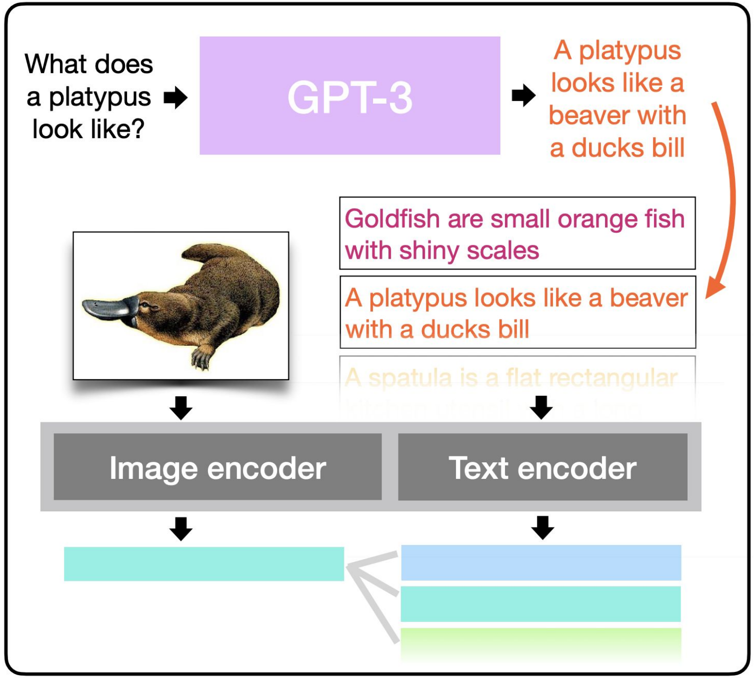
Lorikeet

Marimba

Viaduct

Papillon


A photo of a goldfish
A photo of a platypus
A photo of a spatula
Image encoder
Text encoder


What does a platypus look like?
GPT-3
A platypus looks like a beaver with a ducks bill
Goldfish are small orange fish with shiny scales
A platypus looks like a beaver with a ducks bill
A spatula is a flat rectangular
Image encoder
Text encoder

	ImageNet	DTD	Stanford Cars	SUN397	Food101	FGVC Aircraft	Oxford Pets	Caltech101	Flowers 102	UCF101	Kinetics-700	RESISC45	CIFAR-10	CIFAR-100	Birdsnap
std	75.54	55.20	77.53	69.31	93.08	32.88	93.33	93.24	78.53	77.45	60.07	71.10	95.59	78.26	50.43
# hw	80	8	8	2	1	2	1	34	1	48	28	18	18	18	1
CuPL (base)	76.19	58.90	76.49	72.74	93.33	36.69	93.37	93.45	78.83	77.74	60.24	68.96	95.81	78.47	51.11
Δ std	+0.65	+3.70	-1.04	+3.43	+0.25	+3.81	+0.04	+0.21	+0.30	+0.29	+0.17	-2.14	+0.22	+0.21	+0.63
# hw	3	3	3	3	3	3	3	3	3	3	3	3	3	3	3

Thank you!