Lecture 9: More on Transformers and Multimodal Models

Announcements

- Upcoming deadlines:
  - A2 due next Tue Nov 1
  - Midterm: In class, Mon Nov 7
    - 80 minutes
    - 1 page 8.5” x 11” of notes allowed (back and front)
    - No calculators allowed or needed
    - Covers material through “Genomics: Introduction”
    - Practice midterm will be released about a week before the midterm

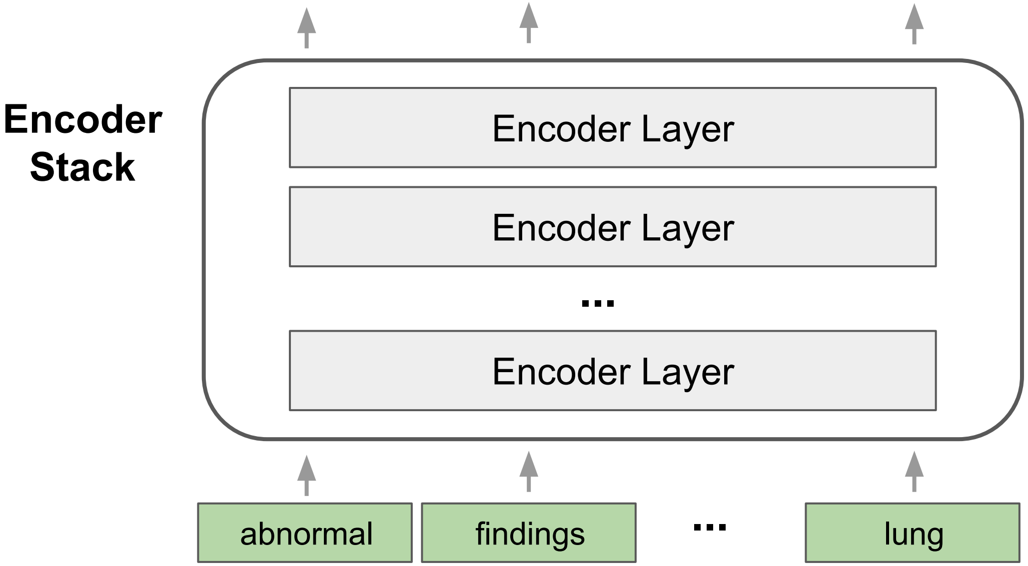
Previously, saw BERT: Highly successful transfer learning through learning bidirectional representations with a “Transformer” architecture

- BERT: **B**idirectional **E**ncoder **R**epresentations from **T**ransformers
- Builds on ELMo idea of bidirectional context embeddings, but introduces advancements with “Transformer” architecture and new training objectives
- Showed that learned model could be a successful “pre-trained” model that could be fine-tuned to achieve state-of-the-art performance on 11 different NLP tasks: an “ImageNet” moment for NLP

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

BERT architecture framework

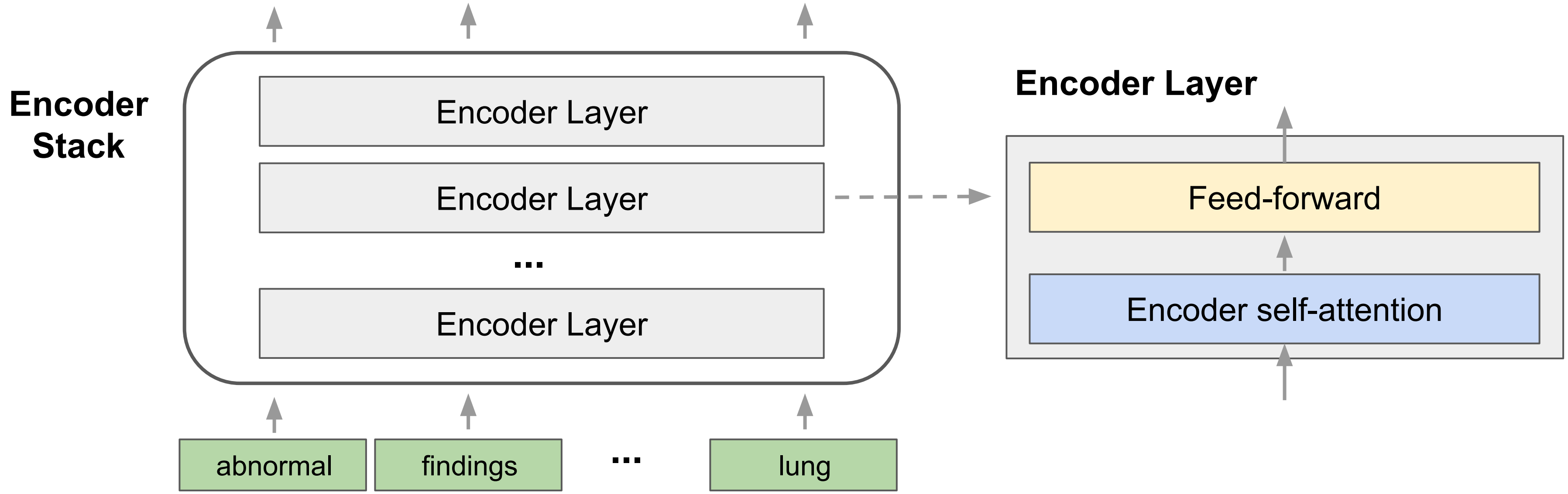
- Recent approach for sequence processing based on “self-attention” (Vaswani et al. 2017). BERT uses a stack of “encoder layers” each with self-attention (original Transformer also had decoder layers).



Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.
Vaswani et al. Attention is All You Need, 2017.

BERT architecture framework

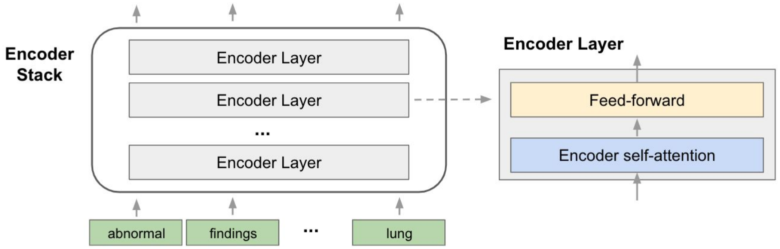
- Recent approach for sequence processing based on “self-attention” (Vaswani et al. 2017). BERT uses a stack of “encoder layers” each with self-attention (original Transformer also had decoder layers).



Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.
Vaswani et al. Attention is All You Need, 2017.

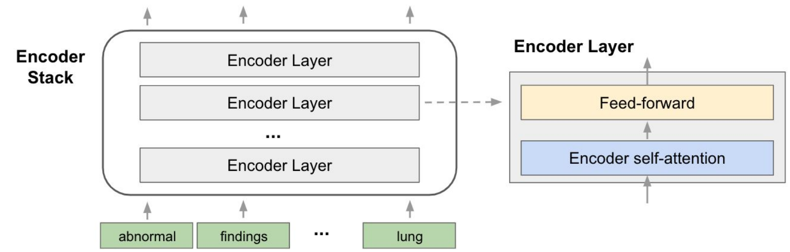
Today, bigger picture: Transformer-based architectures can be comprised of encoder and/or decoder layers

Encoders only

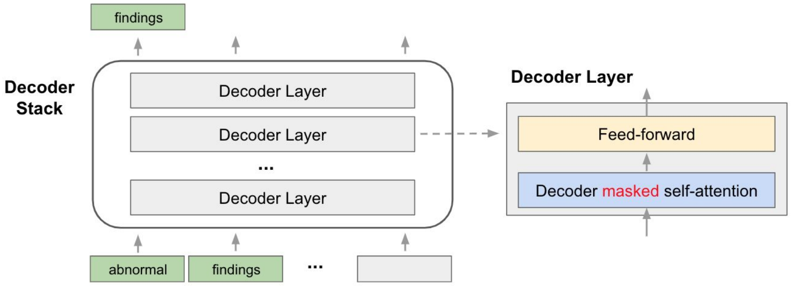


Today, bigger picture: Transformer-based architectures can be comprised of encoder and/or decoder layers

Encoders only

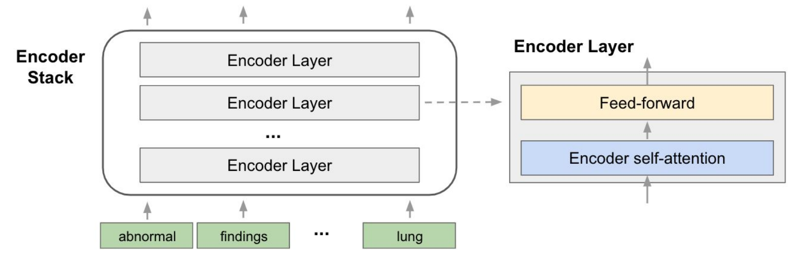


Decoders only

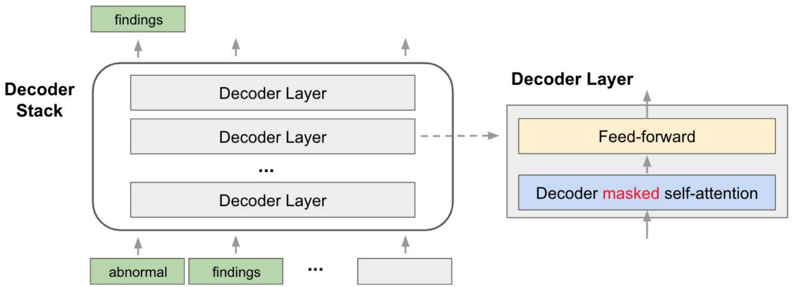


Today, bigger picture: Transformer-based architectures can be comprised of encoder and/or decoder layers

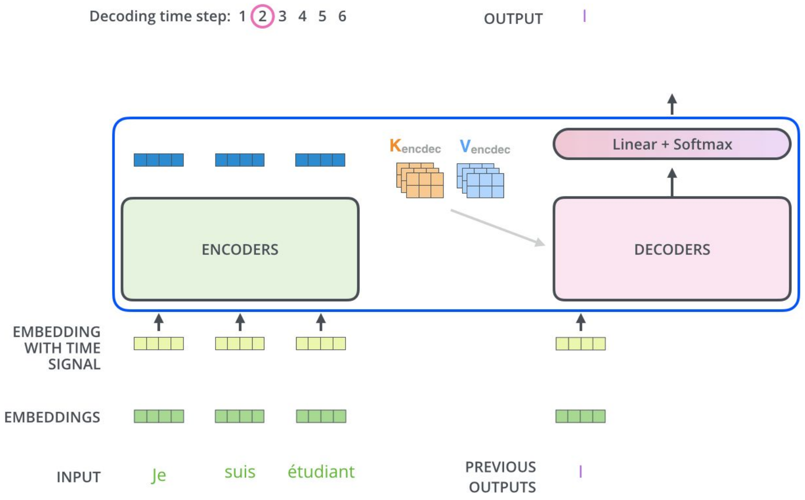
Encoders only



Decoders only



Encoder-decoder



Encoder-decoder figure credit: https://jalammar.github.io/illustrated-transformer/

Review: Transformer "attention" mechanism

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

$$a_j = \text{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)V_j(y)$$

Vaswani et al. Attention is All You Need, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Transformer “attention” mechanism

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

“Query” embedding: [num_x, d_c] where
d_c is embedding dimension

$$a_j = \mathrm{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)V_j(y)$$

Vaswani et al. Attention is All You Need, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Transformer "attention" mechanism

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

"Query" embedding: [num_x, d_c] where
d_c is embedding dimension

"Key" embedding: [num_y, d_c]

$$a_j = \mathrm{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)V_j(y)$$

Vaswani et al. Attention is All You Need, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Transformer "attention" mechanism

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

"Query" embedding: [num_x, d_c] where
d_c is embedding dimension

"Key" embedding: [num_y, d_c]

$$a_j = \text{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)V_j(y)$$

Embedding dimension

Vaswani et al. Attention is All You Need, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Transformer “attention” mechanism

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

“Query” embedding: [num_x, d_c] where d_c is embedding dimension

“Key” embedding: [num_y, d_c]

$$a_j = \underbrace{\text{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)}_{\text{Attention weights}} V_j(y)$$

Embedding dimension

Vaswani et al. Attention is All You Need, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Transformer "attention" mechanism

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

"Query" embedding: [num_x, d_c] where
d_c is embedding dimension

"Key" embedding: [num_y, d_c]

$$a_j = \underbrace{\text{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)}_{\text{Attention weights}} V_j(y)$$

Embedding dimension

"Value" embedding:
[num_y, d_c]

Attention weights

Vaswani et al. Attention is All You Need, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Transformer “attention” mechanism

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

“Query” embedding: [num_x, d_c] where d_c is embedding dimension

“Key” embedding: [num_y, d_c]

$$a_j = \underbrace{\text{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)}_{\text{Attention weights}} V_j(y)$$

Attention-weighted outputs of jth attention head: [num_x, d_c]

Embedding dimension

“Value” embedding: [num_y, d_c]

Vaswani et al. Attention is All You Need, 2017.

Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Transformer "attention" mechanism

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

"Query" embedding: [num_x, d_c] where d_c is embedding dimension

"Key" embedding: [num_y, d_c]

$$a_j = \underbrace{\text{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)}_{\text{Attention weights}} V_j(y)$$

Attention-weighted outputs of jth attention head: [num_x, d_c]

Embedding dimension

"Value" embedding: [num_y, d_c]

If using multiple attention heads, combine outputs with another matrix multiply

Attention weights

Vaswani et al. Attention is All You Need, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

Transformer "self-attention" mechanism

"Self-attention" is just this attention mechanism with x = y!

Consider first attention between a sequence x (of length num_x), and a sequence y (of length num_y):

"Query" embedding: [num_x, d_c] where d_c is embedding dimension

"Key" embedding: [num_y, d_c]

$$a_j = \underbrace{\mathrm{softmax}\left(\frac{Q_j(x)K_j(y)^T}{\sqrt{d_c}}\right)}_{\text{Attention weights}} V_j(y)$$

Attention-weighted outputs of jth attention head: [num_x, d_c]

Embedding dimension

"Value" embedding: [num_y, d_c]

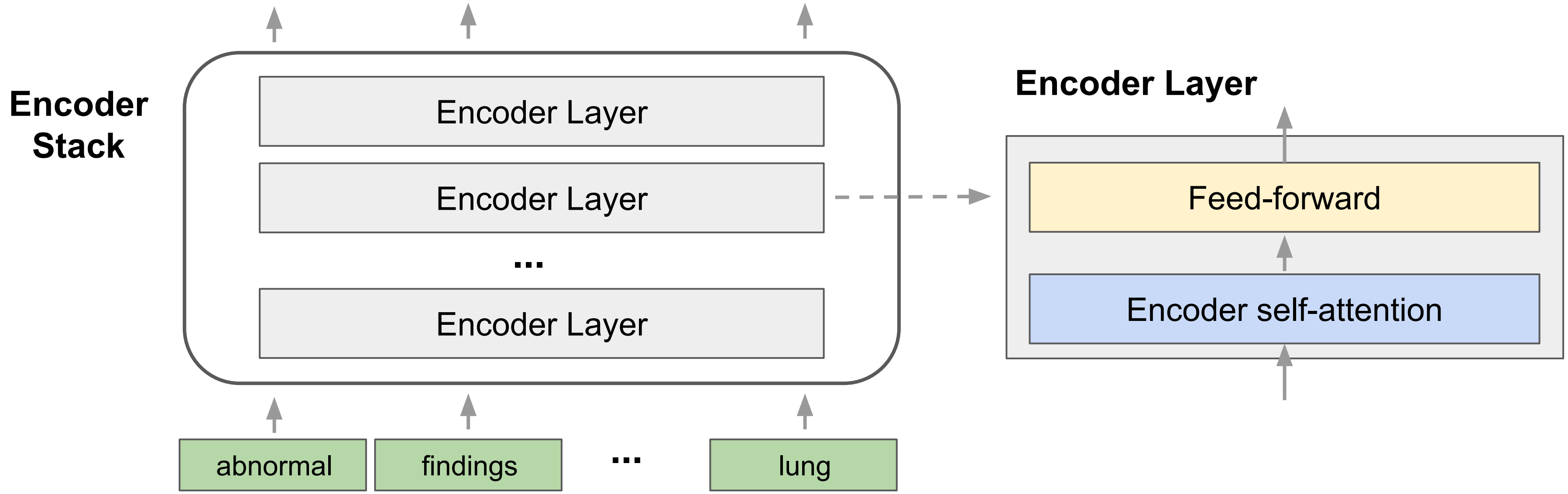
Attention weights

If using multiple attention heads, combine outputs with another matrix multiply

Vaswani et al. Attention is All You Need, 2017.
Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.

BERT architecture framework

- Recent approach for sequence processing based on "self-attention" (Vaswani et al. 2017). BERT uses a stack of "encoder layers" each with self-attention (original Transformer also had decoder layers).



Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.
Vaswani et al. Attention is All You Need, 2017.

Another aspect of the architecture to be aware of (in this class, at a high level): positional encoding

Vector added to each input embedding that gives information about the relative location of that input within the sequence. Often a fixed function, sometimes can also be learned.

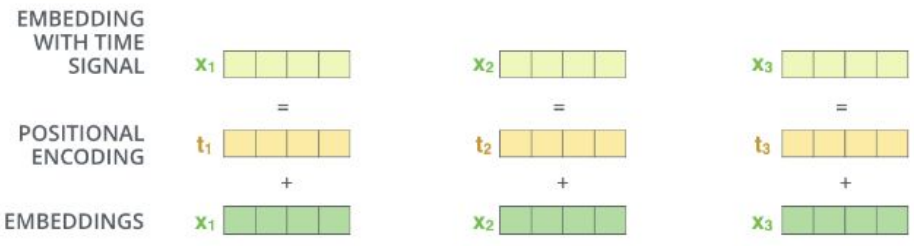


Figure credit: https://jalammar.github.io/illustrated-transformer/

Another aspect of the architecture to be aware of (in this class, at a high level): positional encoding

Vector added to each input embedding that gives information about the relative location of that input within the sequence. Often a fixed function, sometimes can also be learned.

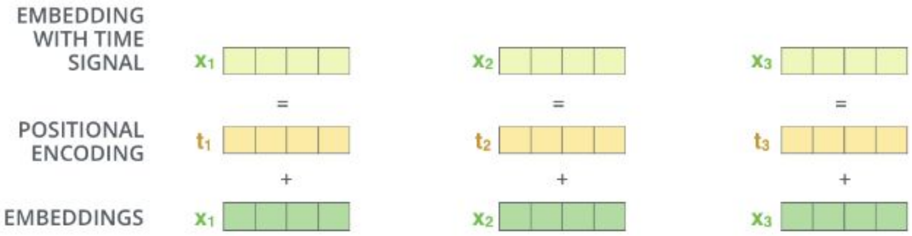
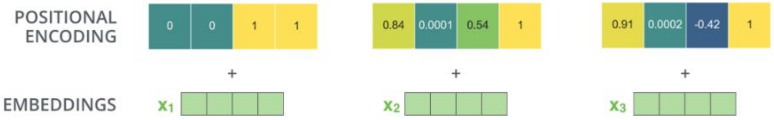
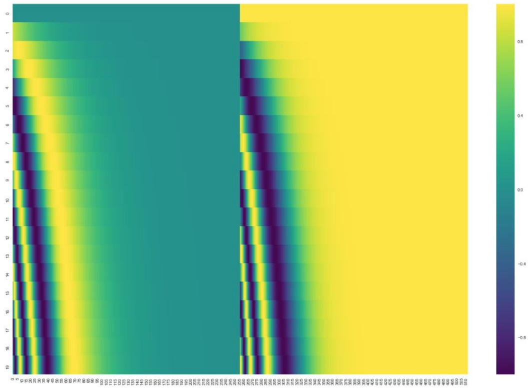


Figure credit: https://jalammar.github.io/illustrated-transformer/

Example of positional encoding based on sine/cosine functions:

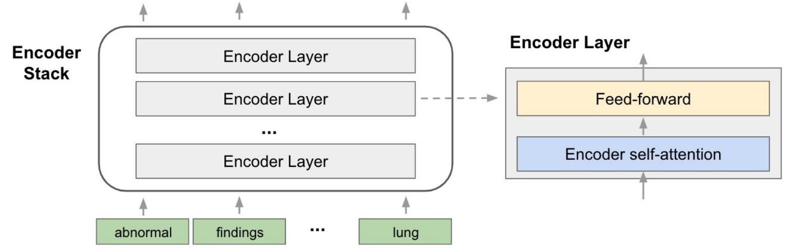




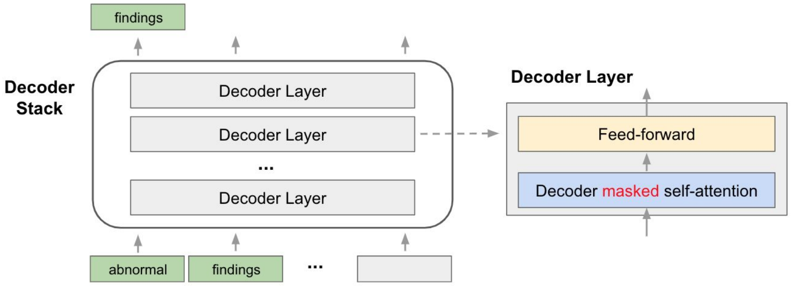
Example of positional encoding vectors corresponding to 512-dim embeddings (x-axis), for 20 positions (y-axis)

Bigger picture: Transformer-based architectures can be comprised of encoder and/or decoder layers

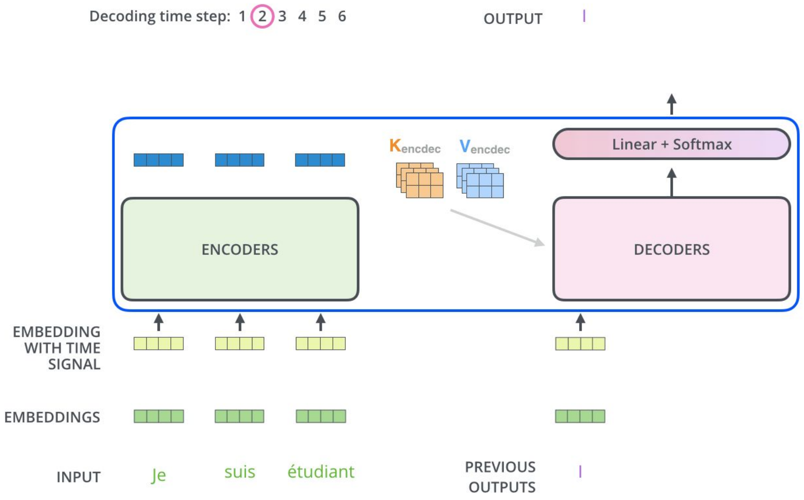
Encoders only



Decoders only



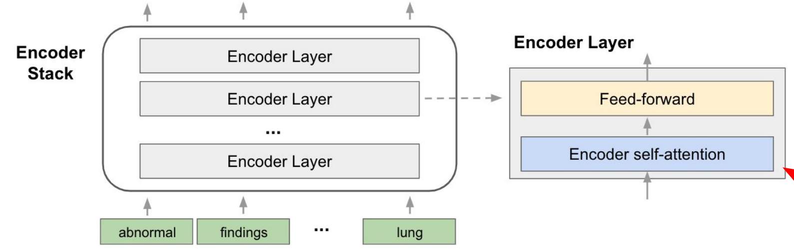
Encoder-decoder



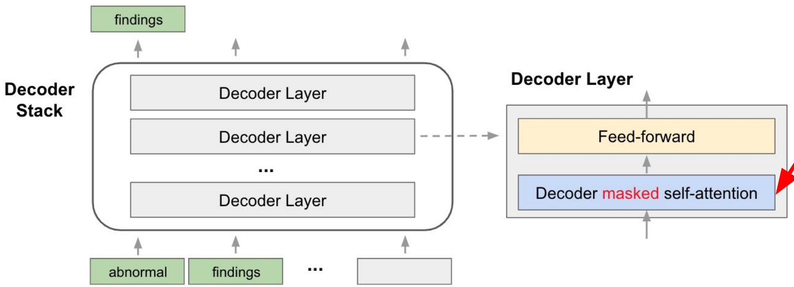
Encoder-decoder figure credit: https://jalammar.github.io/illustrated-transformer/

Bigger picture: Transformer-based architectures can be comprised of encoder and/or decoder layers

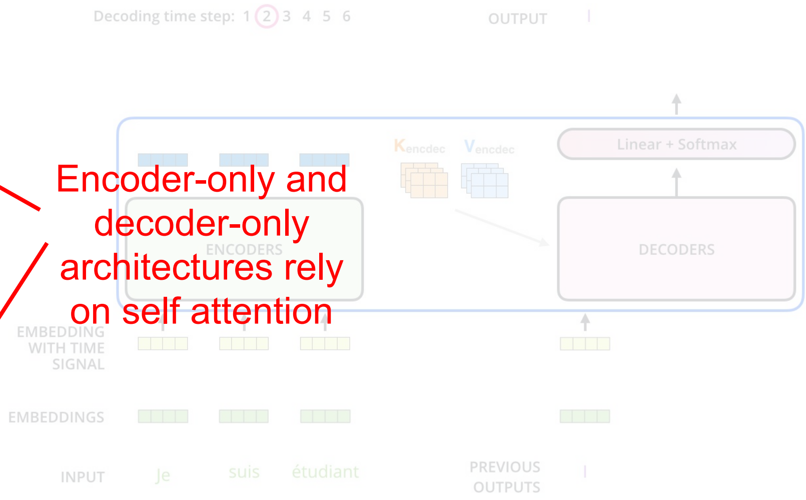
Encoders only



Decoders only



Encoder-decoder

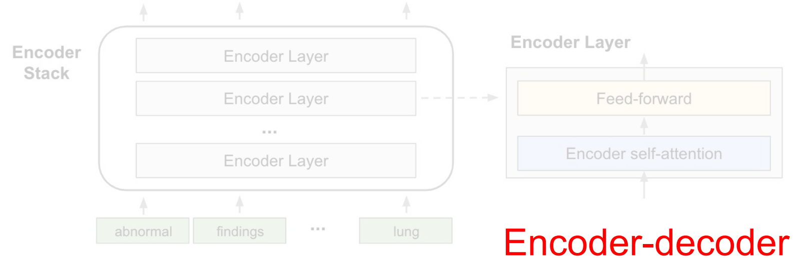


Encoder-only and decoder-only architectures rely on self attention

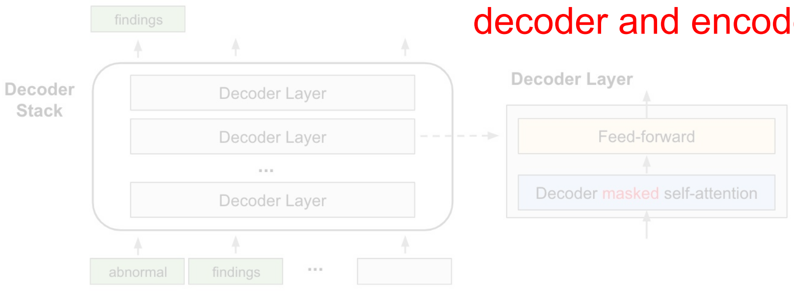
Encoder-decoder figure credit: https://jalammar.github.io/illustrated-transformer/

Bigger picture: Transformer-based architectures can be comprised of encoder and/or decoder layers

Encoders only

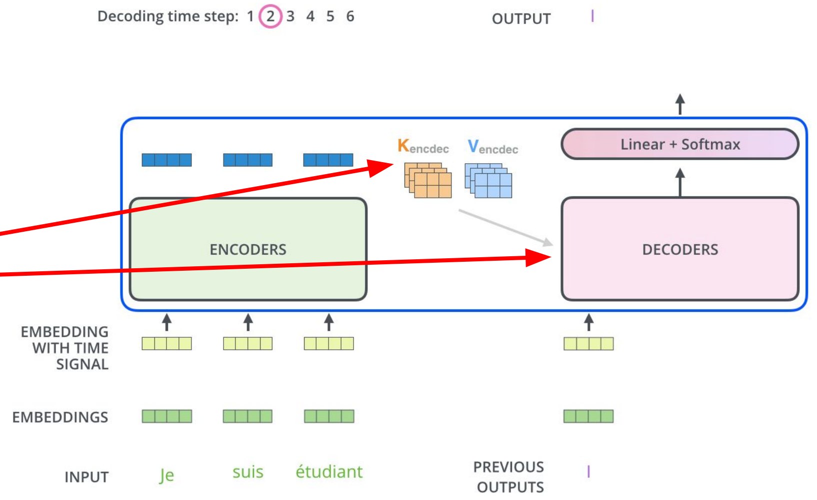


Decoders only



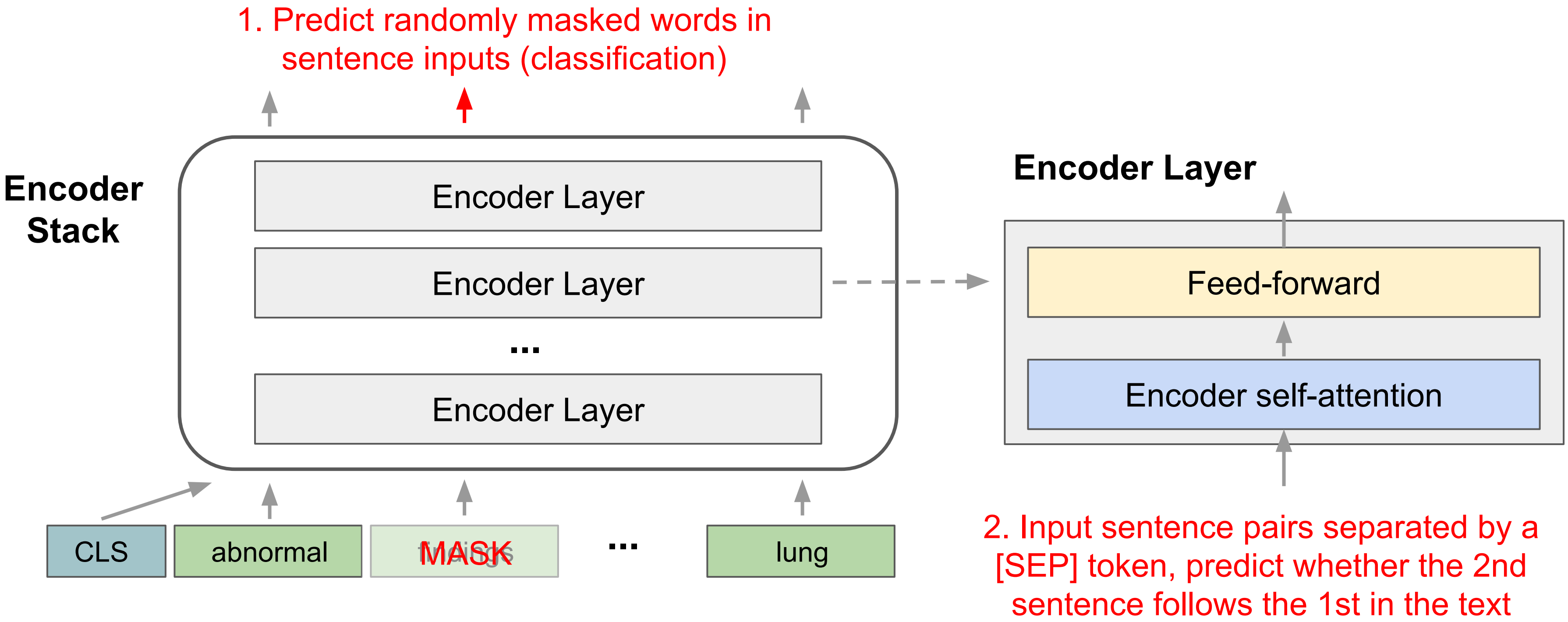
Encoder-decoder architectures use attention between the decoder and encoder

Encoder-decoder



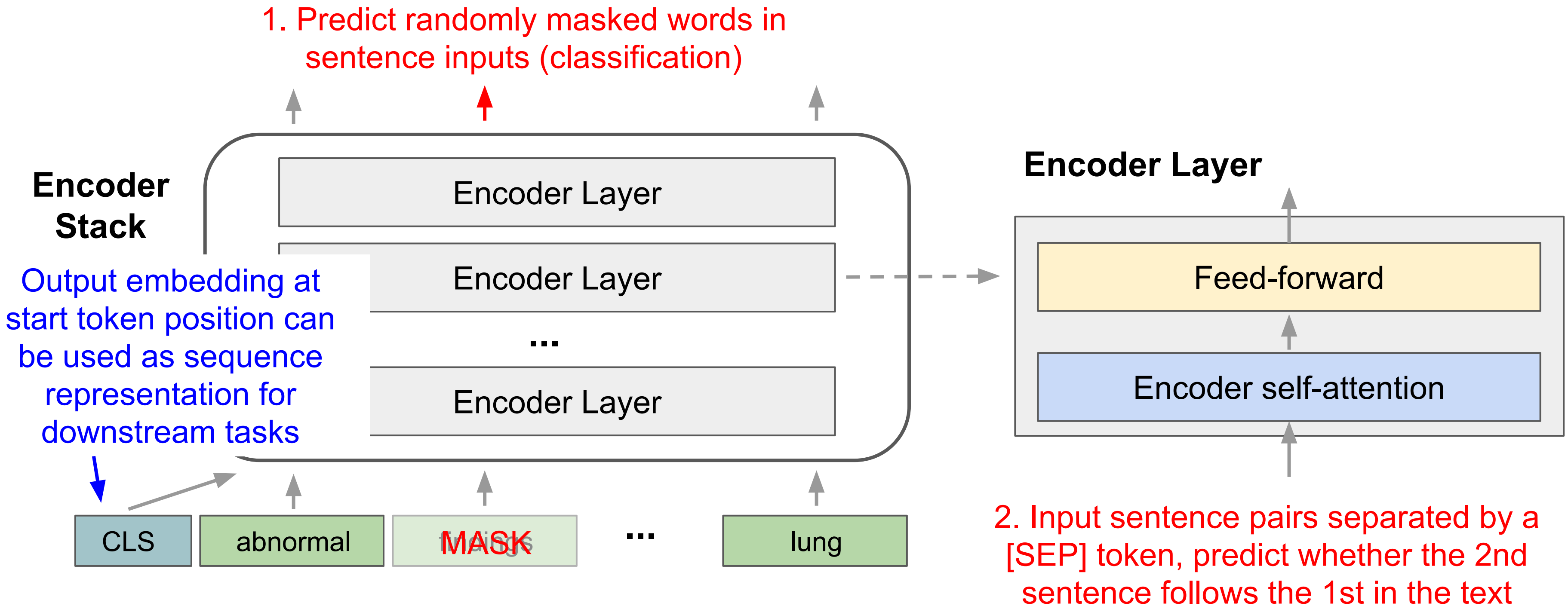
Encoder-decoder figure credit: https://jalammar.github.io/illustrated-transformer/

Remember: BERT was trained with masked-word and sentence pair self-supervised objectives



Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.
Vaswani et al. Attention is All You Need, 2017.

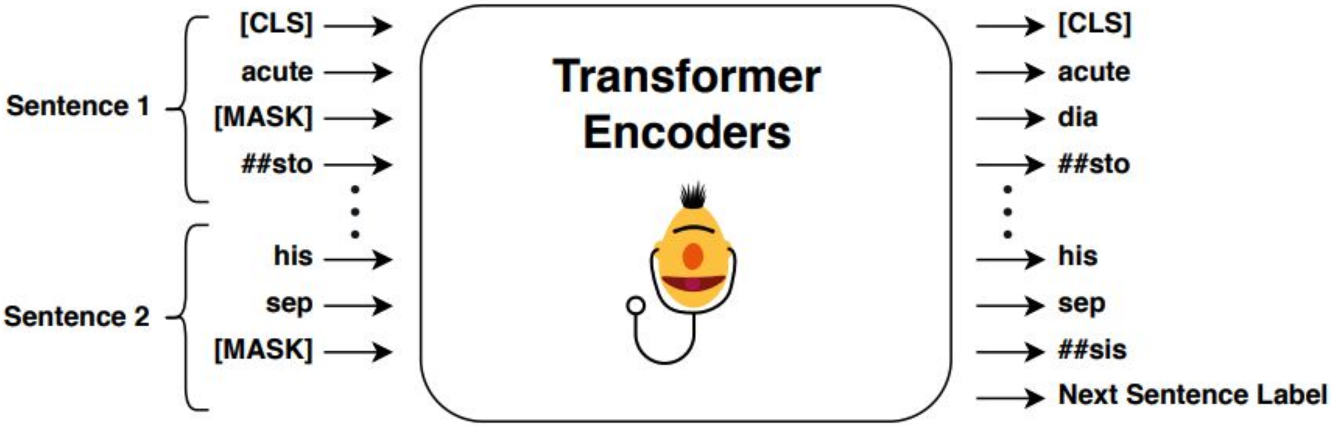
Remember: BERT was trained with masked-word and sentence pair self-supervised objectives



Devlin et al. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, 2018.
Vaswani et al. Attention is All You Need, 2017.

Example of ClinicalBERT: training on clinical notes (from MIMIC)

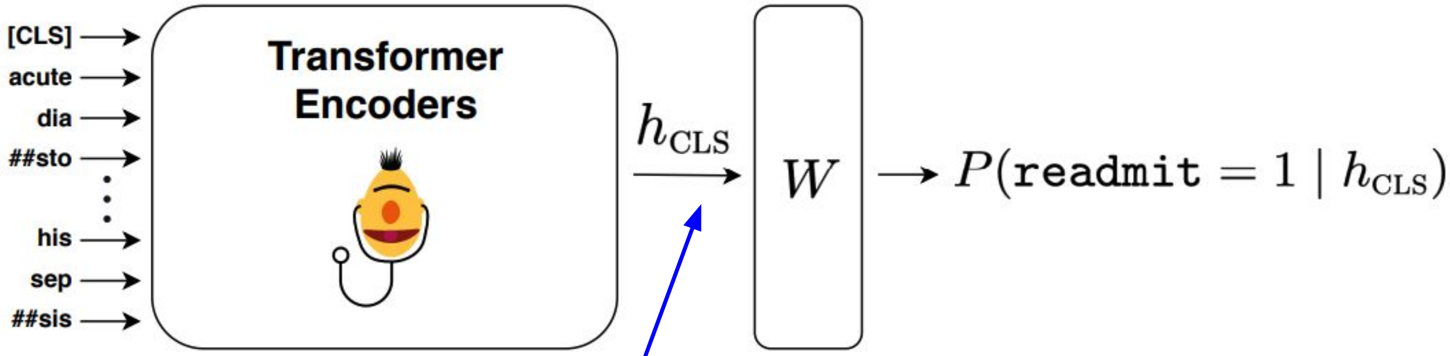
Training ClinicalBERT with the masked prediction and next sentence objectives:



Huang et al. ClinicalBert: Modeling Clinical Notes and Predicting Hospital Readmission, 2019.

Example of ClinicalBERT: training on clinical notes (from MIMIC)

Fine-tuning ClinicalBERT for prediction of 30-day hospital readmission:

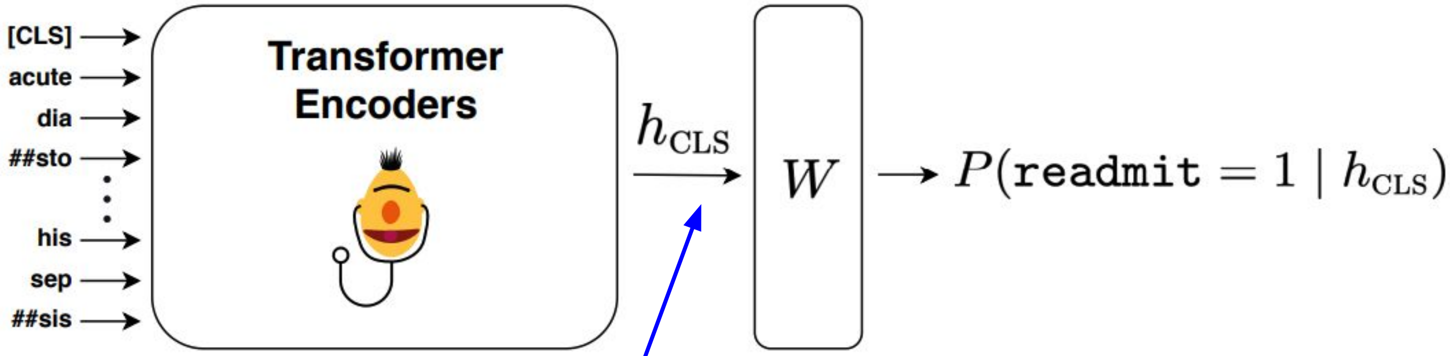


Use hidden state corresponding to [CLS] token

Huang et al. ClinicalBert: Modeling Clinical Notes and Predicting Hospital Readmission, 2019.

Example of ClinicalBERT: training on clinical notes (from MIMIC)

Fine-tuning ClinicalBERT for prediction of 30-day hospital readmission:



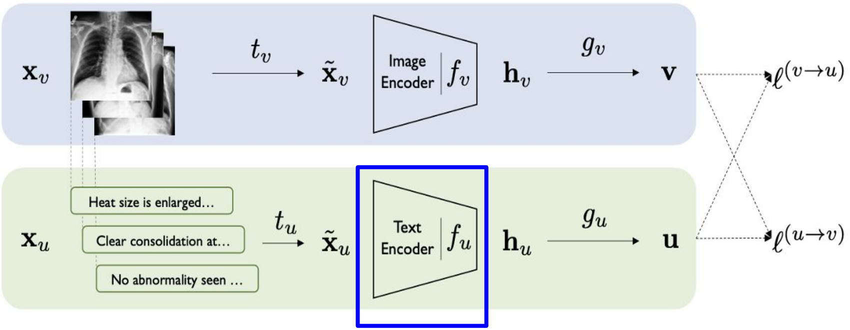
Use hidden state corresponding to [CLS] token

When performing prediction from long sequences, obtain predictions for each sentence separately and then combine

Huang et al. ClinicalBert: Modeling Clinical Notes and Predicting Hospital Readmission, 2019.

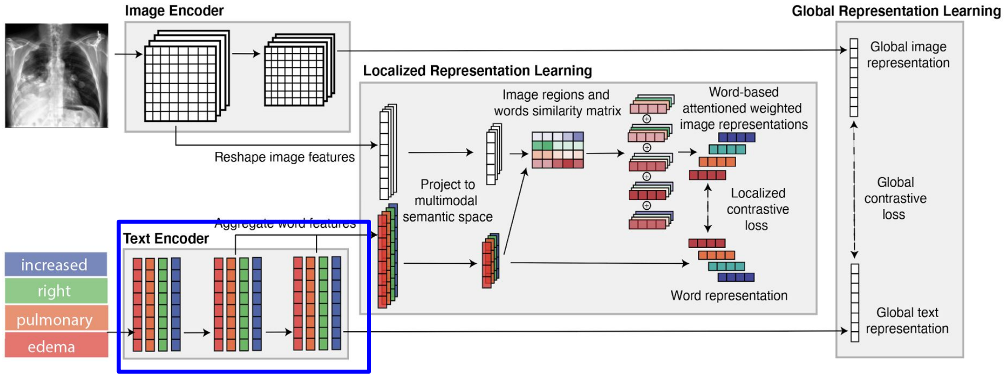
In previous lecture: BERT-based text representations also used in ConVIRT and GLoRIA models

ConVIRT



Zhang et al. 2020.

GLoRIA



Huang et al. 2021.

Next, GPT: Based on Transformer decoder layers

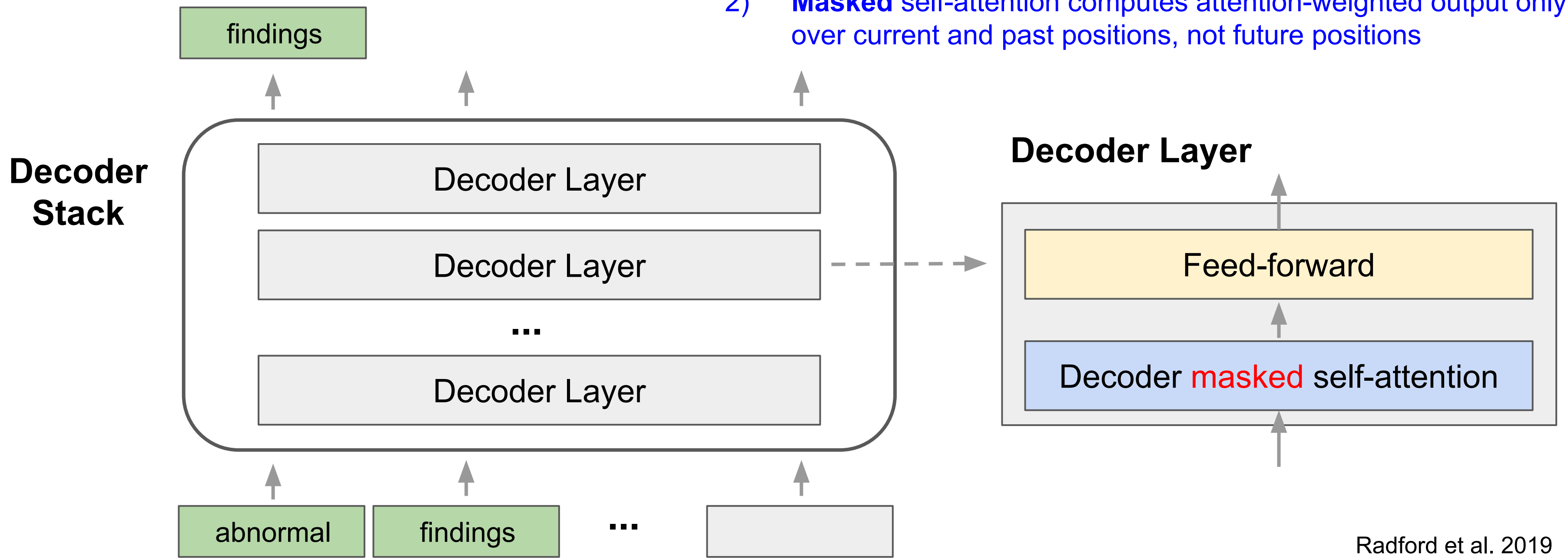
Generative Pre-Training (GPT): First introduced by Radford et al. 2018 (OpenAI), with subsequent GPT-2 and GPT-3 of increasingly larger scale

Radford et al. 2019
Brown et al. 2020

GPT: Based on Transformer decoder layers

Key architectural differences with encoder stack:

1) Output elements are produced sequentially; at inference time, output y_t at position t is fed as input x_(t+1) at the next position
2) **Masked** self-attention computes attention-weighted output only over current and past positions, not future positions



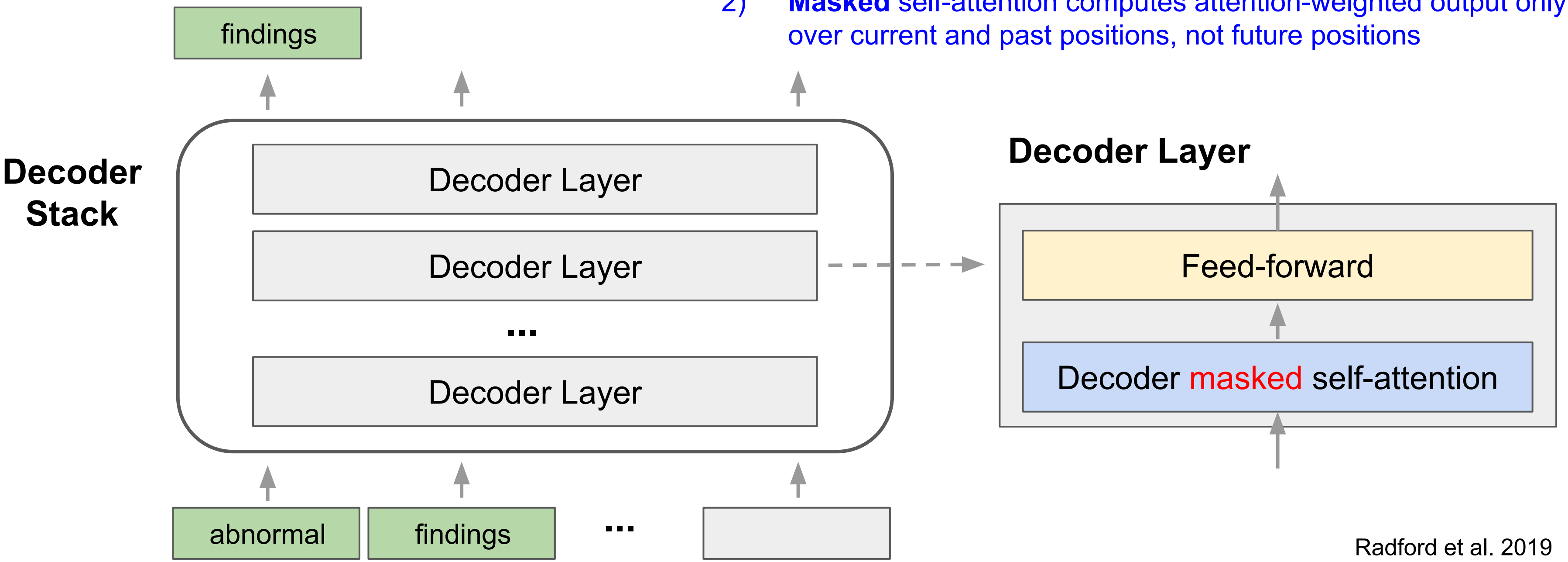
Radford et al. 2019
Brown et al. 2020

GPT: Based on Transformer decoder layers

Training objective: next-token prediction, a standard language-modeling objective (contrast with bidirectional BERT objective)

Key architectural differences with encoder stack:

1) Output elements are produced sequentially; at inference time, output y_t at position t is fed as input x_(t+1) at the next position
2) **Masked** self-attention computes attention-weighted output only over current and past positions, not future positions



Radford et al. 2019
Brown et al. 2020

GPT: Natural text generation capabilities

```
Title:  United Methodists Agree to Historic Split
Subtitle:  Those who oppose gay marriage will form their own denomination
Article:  After two days of intense debate, the United Methodist Church
has agreed to a historic split - one that is expected to end in the
creation of a new denomination, one that will be "theologically and
socially conservative," according to The Washington Post.  The majority of
delegates attending the church's annual General Conference in May voted to
strengthen a ban on the ordination of LGBTQ clergy and to write new rules
that will "discipline" clergy who officiate at same-sex weddings.  But
those who opposed these measures have a new plan:  They say they will form a
separate denomination by 2020, calling their church the Christian Methodist
denomination.
The Post notes that the denomination, which claims 12.5 million members, was
in the early 20th century the "largest Protestant denomination in the U.S.,"
but that it has been shrinking in recent decades.  The new split will be the
second in the church's history.  The first occurred in 1968, when roughly
10 percent of the denomination left to form the Evangelical United Brethren
Church.  The Post notes that the proposed split "comes at a critical time
for the church, which has been losing members for years," which has been
"pushed toward the brink of a schism over the role of LGBTQ people in the
church." Gay marriage is not the only issue that has divided the church.  In
2016, the denomination was split over ordination of transgender clergy, with
the North Pacific regional conference voting to ban them from serving as
clergy, and the South Pacific regional conference voting to allow them.
```

Example from GPT-3

Radford et al. 2019
Brown et al. 2020

GPT: Natural text generation capabilities

Context →	The City BY C. P. CAVAFY TRANSLATED BY EDMUND KEELEY [Poem text omitted] SOME TREES John Ashbery [Poem text omitted] Shadows on the Way Wallace Stevens

```
The sun was all we had.  Now, in the shade
All is changed.  The mind must dwell on those
White fields, that to its eyes were always old;
Those ancient gleams, convoluting
The way they lay among the huge roots,
The great dark tomes of reverie,
The plumed labyrinth of the sea.

When our own breath gonne faint and fade
Up the long stairway of the dead
We shall remember well
The blindness of the light, the whiteness
Of that white land.  The footsteps, and
The grasses in that light, and in the shade
The menacing life.

We played, too, once, in the warmest rooms.
To one content, there is one content
For all mankind.  The forest glades
Are all the more for being in fable,
And wood and town in legend shadowed-
Once more to return.  The one content
Has in it every happiness, more brave
For being purposeless; and so
Those island gardens, deep in the mind,
Make of remoteness familiarity
And of the unspoken name a friend.
```

Example from GPT-3

Radford et al. 2019
Brown et al. 2020

GPT: Utilizing for other downstream tasks

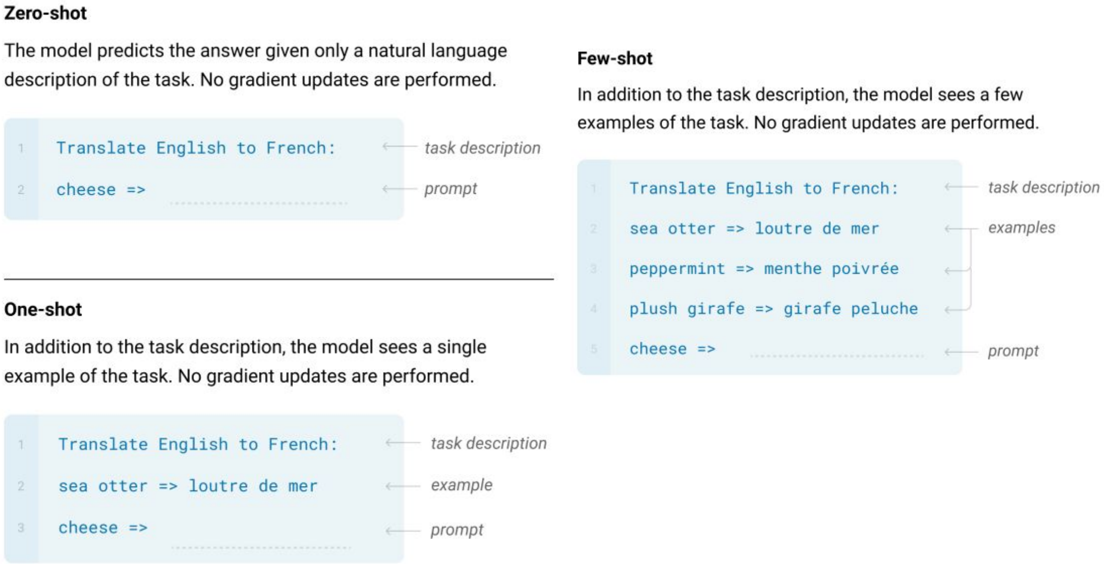
Fine-tune on downstream tasks by re-formatting data for task into sequences for completion

Radford et al. 2019
Brown et al. 2020

GPT: Utilizing for other downstream tasks

Fine-tune on downstream tasks by re-formatting data for task into sequences for completion

Alternatively, GPT-3 paper focuses on showing that the trained model is effective in **zero-shot, one-shot, and few-shot** task settings



Radford et al. 2019
Brown et al. 2020

GPT: Utilizing for other downstream tasks

Example of GPT-3 performing a **one-shot** task of "using a new word in a sentence" (grey text is user-provided, black text is GPT-3 output)

```
A "whatpu" is a small, furry animal native to Tanzania.  An example of a sentence that uses
the word whatpu is:
We were traveling in Africa and we saw these very cute whatpus.
```

```
To do a "farduddle" means to jump up and down really fast.  An example of a sentence that uses
the word farduddle is:
One day when I was playing tag with my little sister, she got really excited and she
started doing these crazy farduddles.
A "yalubalu" is a type of vegetable that looks like a big pumpkin.  An example of a sentence
that uses the word yalubalu is:
I was on a trip to Africa and I tried this yalubalu vegetable that was grown in a garden
there.  It was delicious.
```

```
A "Burringo" is a car with very fast acceleration.  An example of a sentence that uses the
word Burringo is:
In our garage we have a Burringo that my father drives to work every day.
```

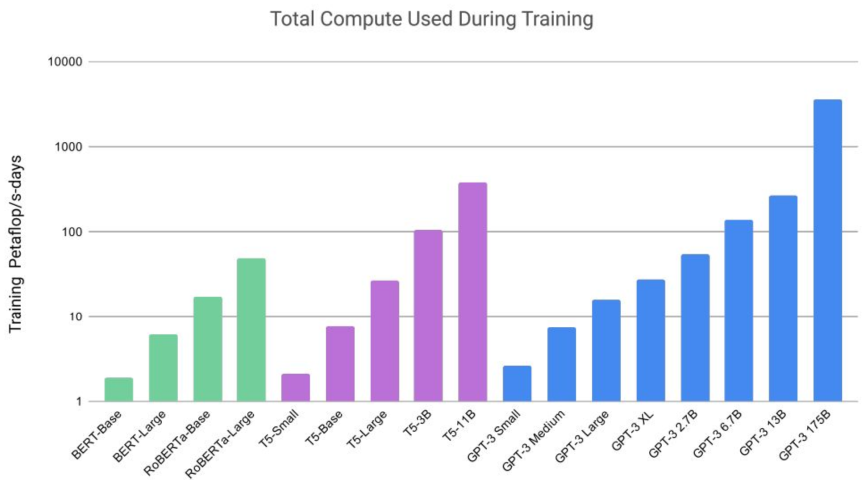
```
A "Gigamuru" is a type of Japanese musical instrument.  An example of a sentence that uses the
word Gigamuru is:
I have a Gigamuru that my uncle gave me as a gift.  I love to play it at home.
```

```
To "screeg" something is to swing a sword at it.  An example of a sentence that uses the word
screeg is:
We screeghed at each other for several minutes and then we went outside and ate ice cream.
```

Radford et al. 2019
Brown et al. 2020

GPT

- GPT-2 has 1.5B parameters, GPT-3 has 175B parameters (100x increase in model size)!
- GPT-3 trained on 500 billion tokens from 5 datasets
- GPT-3 API accessible at https://beta.openai.com/playground



Dataset	Quantity (tokens)	Weight in training mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

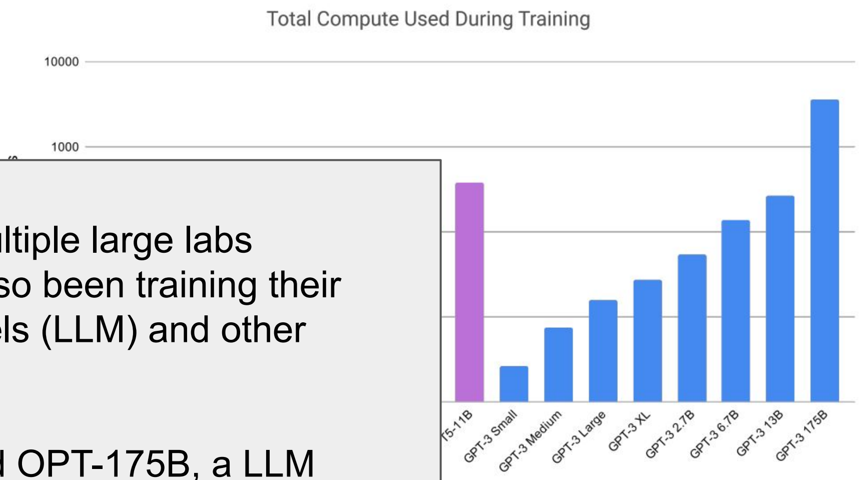
Radford et al. 2019
Brown et al. 2020

GPT

- GPT-2 has 1.5B parameters, GPT-3 has 175B pa model size)!
- GnPT-3 trai from 5 datas
- GPT-3 API a https://beta.o

While GPT is from OpenAI, multiple large labs (especially in industry) have also been training their own very large language models (LLM) and other Transformer-based models.

Recently, Meta AI has released OPT-175B, a LLM similar to GPT-3 (same # of parameters) but open-source (Zhang et al. 2022).



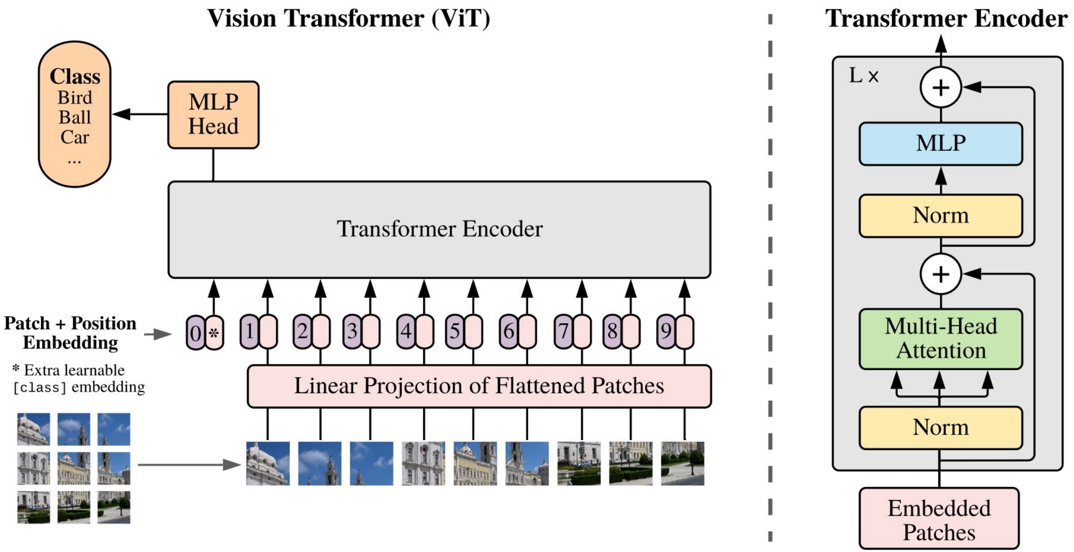
		ht in g mix	Epochs elapsed when training for 300B tokens
Common Crawl (filtered)	410 billion	60%	0.44
WebText2	19 billion	22%	2.9
Books1	12 billion	8%	1.9
Books2	55 billion	8%	0.43
Wikipedia	3 billion	3%	3.4

Radford et al. 2019
Brown et al. 2020

Vision Transformers: ViT

Transformer architecture can be applied to images as well!

Key idea: Convert image into sequence of patches. Can then benefit from Transformer architecture and self-attention, which jointly attends over all patches



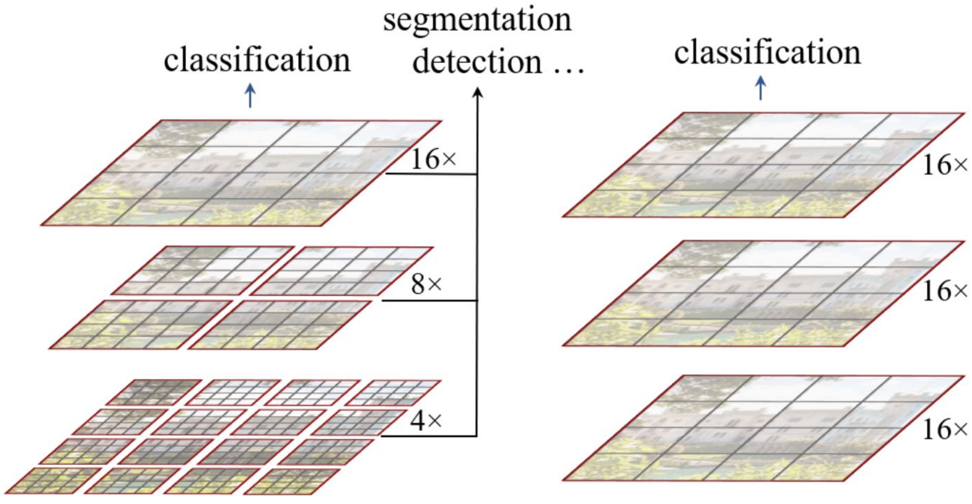
Dosovitsky et al. 2021

Do vision transformers work well?

- ViT first Transformer-based vision model to achieve comparable results to state-of-the-art CNNs, while being more computationally efficient to train
- Transformer architecture has less inductive biases than CNNs (i.e., assumes less about the spatial structure than convolutional filter design does)
  - Consequence: Transformers works well when trained on very large amounts of data, less so when there are smaller / medium amounts of data (in this case, leveraging CNN’s assumptions about data structure is helpful)
- Weakness: Transformer architectures such as ViT also memory-intensive
- Weakness: ViT cannot scale to high-resolution images (due to computational complexity of self-attention), and does not work for denser prediction tasks like object detection or segmentation

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

- Constructs a hierarchical feature representation by starting from small-size patches and gradually merging in deeper layers
  -> suitable for denser vision tasks
- Maintains low computational complexity by computing self-attention only within non-overlapping windows that partition an image



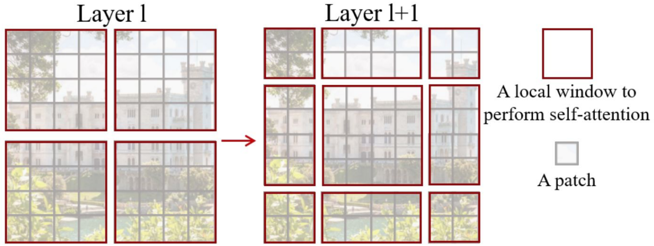
(a) Swin Transformer (ours)

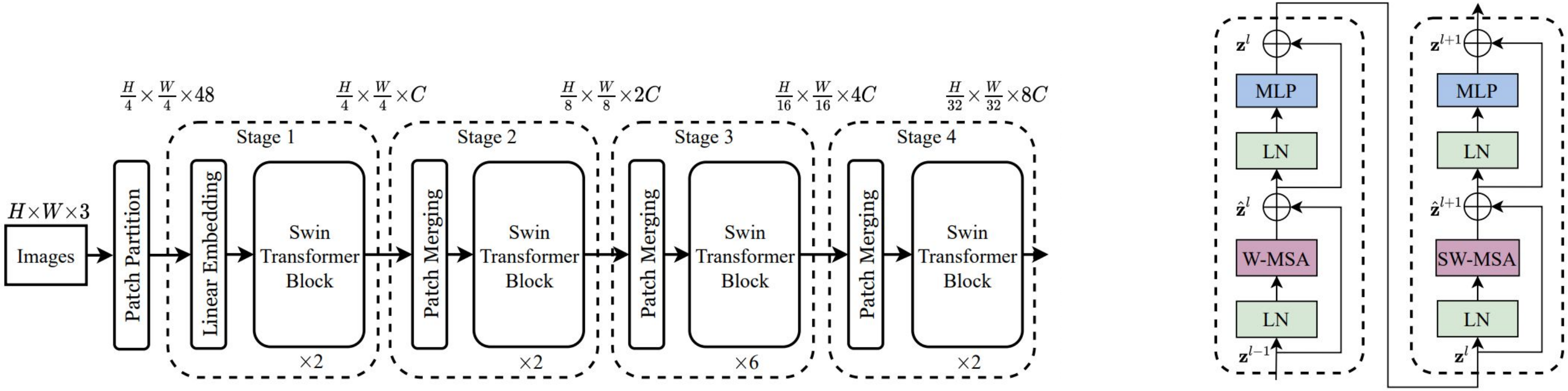
(b) ViT

Liu et al. 2021

Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

An important element: shifted windows for local self-attention at different layers, to provide connections across local regions

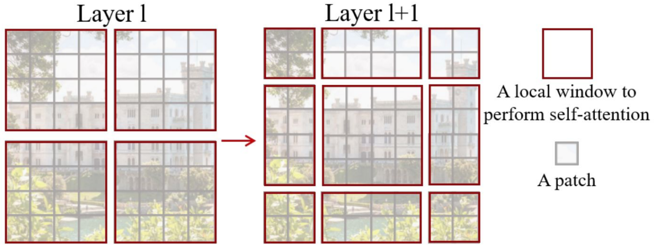




Liu et al. 2021

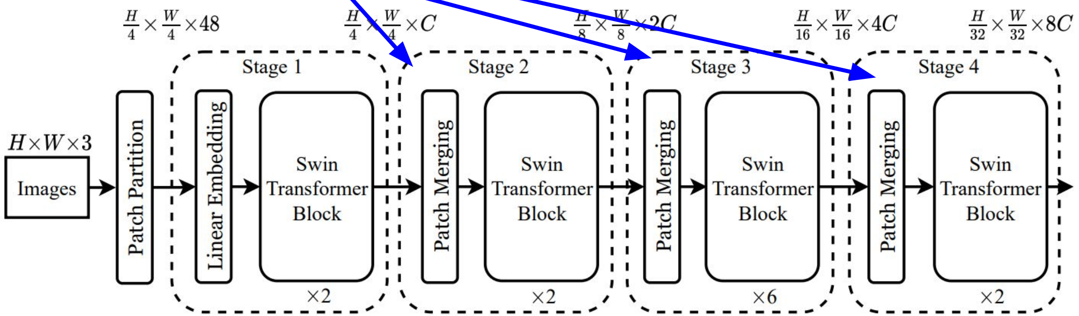
(a) Architecture (b) Two Successive Swin Transformer Blocks

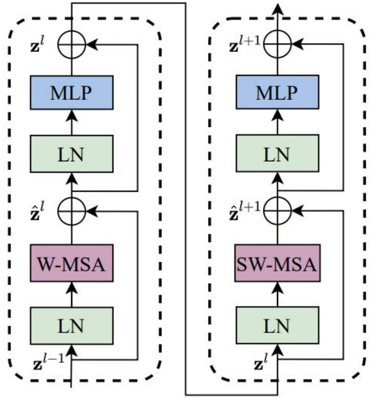
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows



An important element: shifted windows for local self-attention at different layers, to provide connections across local regions

Increasing patch size through merging (earlier layers allow more localized features that can be useful for dense prediction tasks)



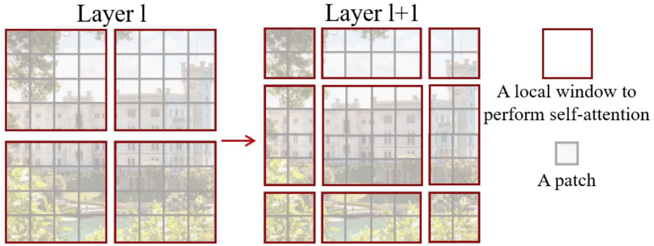


Liu et al. 2021

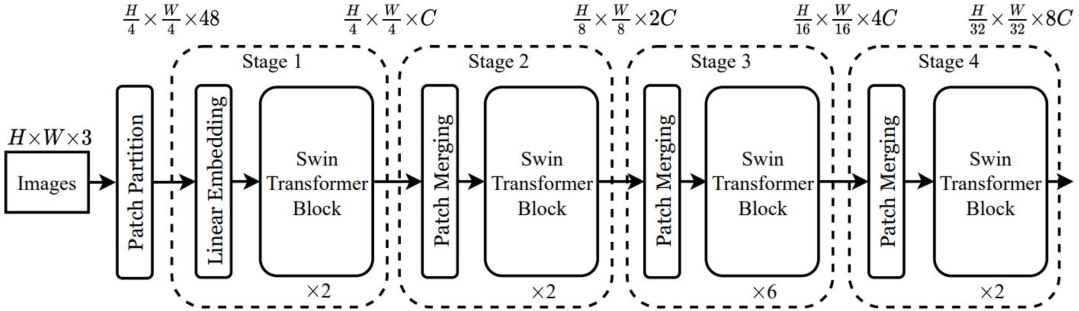
(a) Architecture (b) Two Successive Swin Transformer Blocks

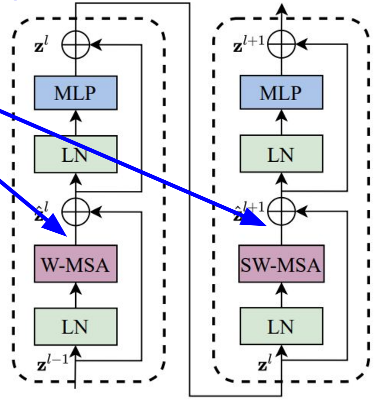
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

An important element: shifted windows for local self-attention at different layers, to provide connections across local regions



Consecutive multi-head self attention modules with regular windows (W-MSA) and shifted windows (SW-MSA)



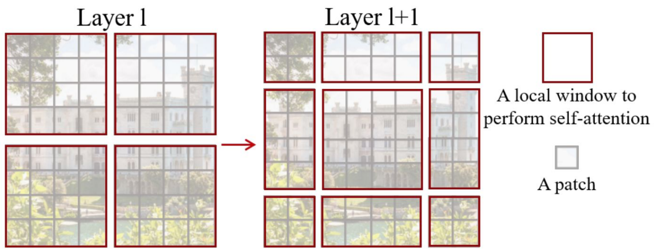


Liu et al. 2021

(a) Architecture (b) Two Successive Swin Transformer Blocks

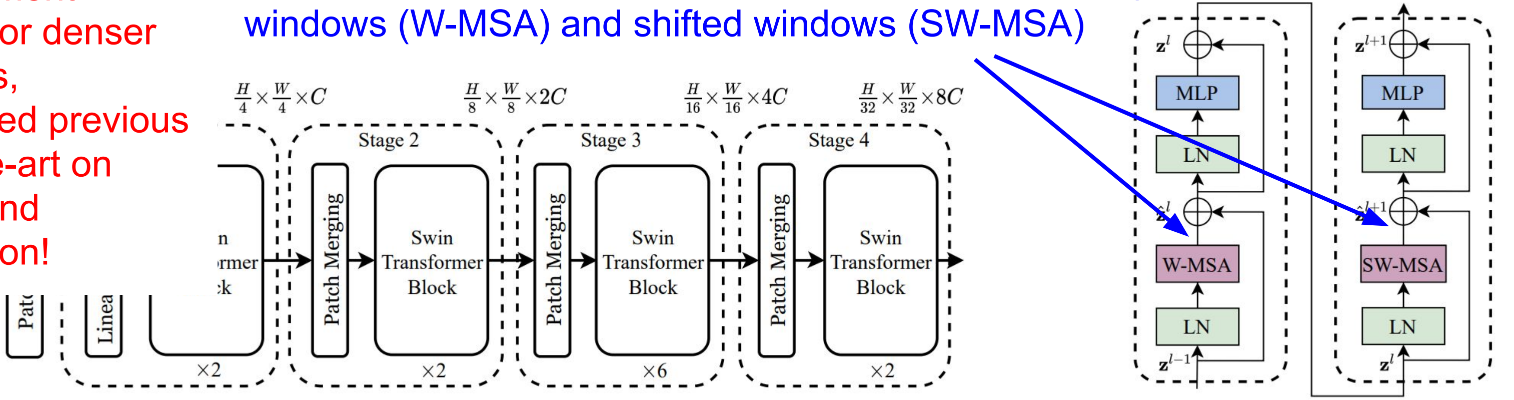
Swin Transformer: Hierarchical Vision Transformer using Shifted Windows

An important element: shifted windows for local self-attention at different layers, to provide connections across local regions



As replacement backbone for denser vision tasks, outperformed previous state-of-the-art on detection and segmentation!

Consecutive multi-head self attention modules with regular windows (W-MSA) and shifted windows (SW-MSA)

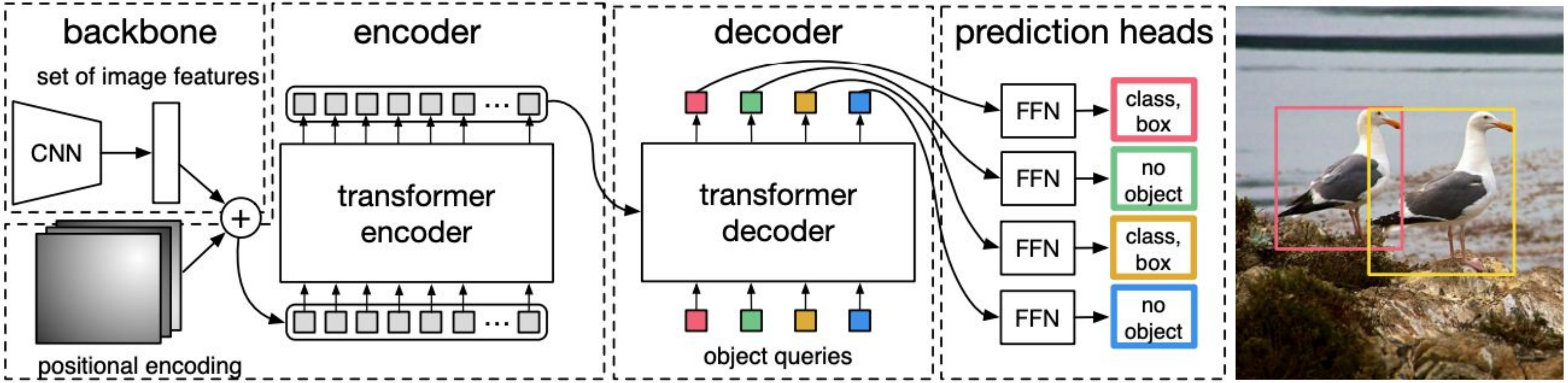


Liu et al. 2021

(a) Architecture (b) Two Successive Swin Transformer Blocks

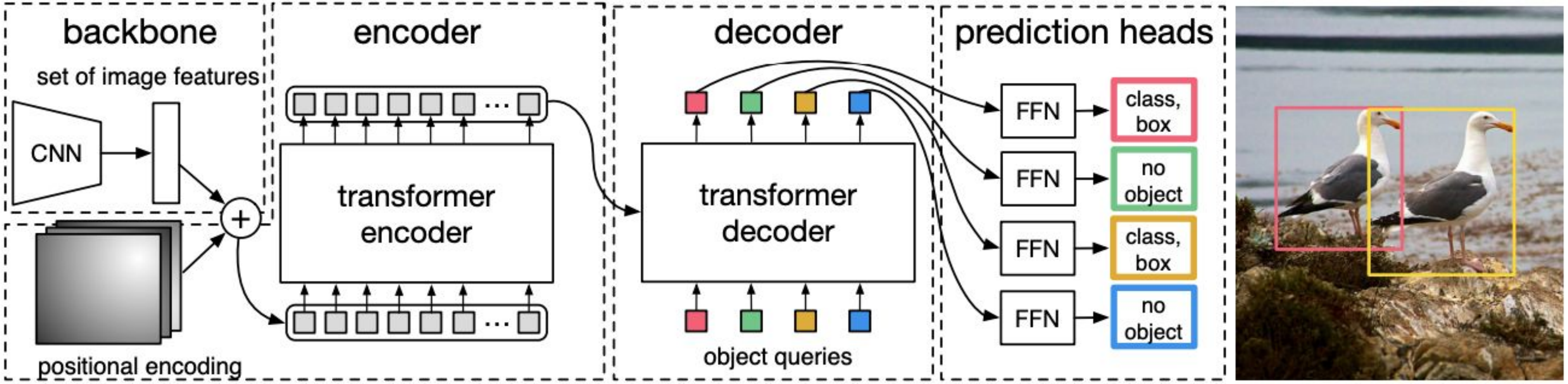
DETR: Transformer-based models for more complex vision tasks

- Uses a Transformer encoder-decoder model to perform detection and segmentation
- Trained directly end-to-end to predict all objects at once, with a set loss function that performs bipartite matching
- Allows avoiding previous hand-designed components of object detection models, like spatial anchors and non-maximal suppression!



Carion et al. 2020

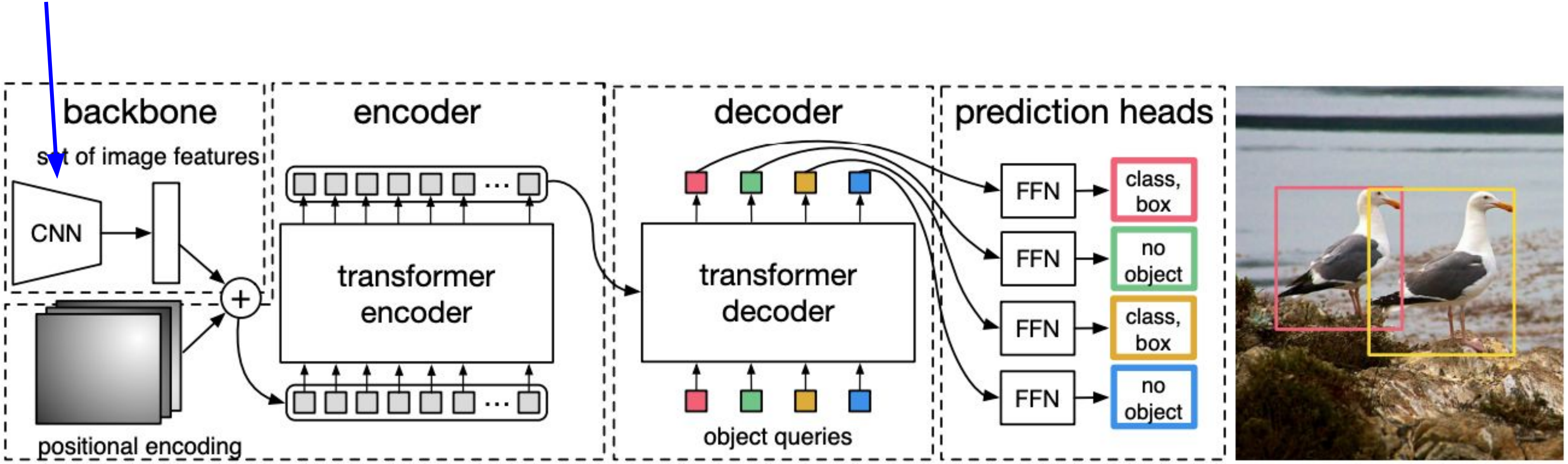
DETR: Transformer-based models for more complex vision tasks



Carion et al. 2020

DETR: Transformer-based models for more complex vision tasks

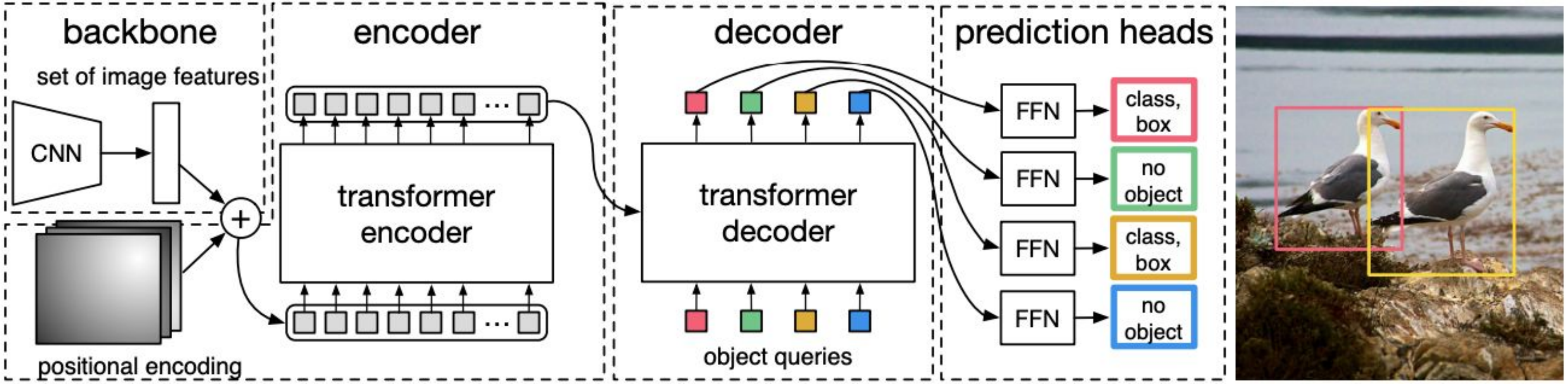
First extract image features with a CNN



Carion et al. 2020

DETR: Transformer-based models for more complex vision tasks

Transformer encoder-decoder part of the model is used for predicting a set of objects from the image features



Carion et al. 2020

DETR: Transformer-based models for more complex vision tasks

Transformer encoder-decoder part of the model is used for predicting a set of objects from the image features

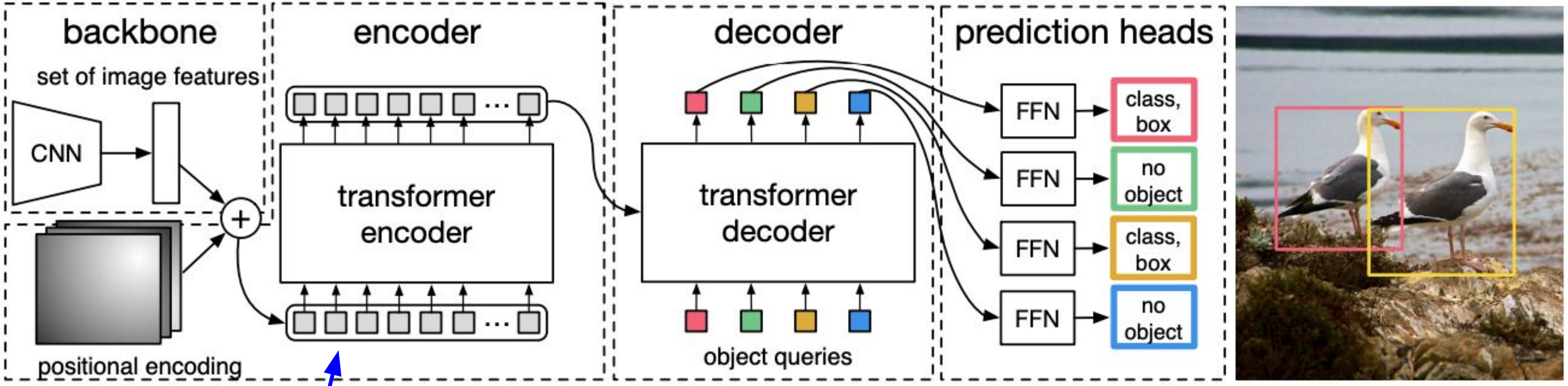
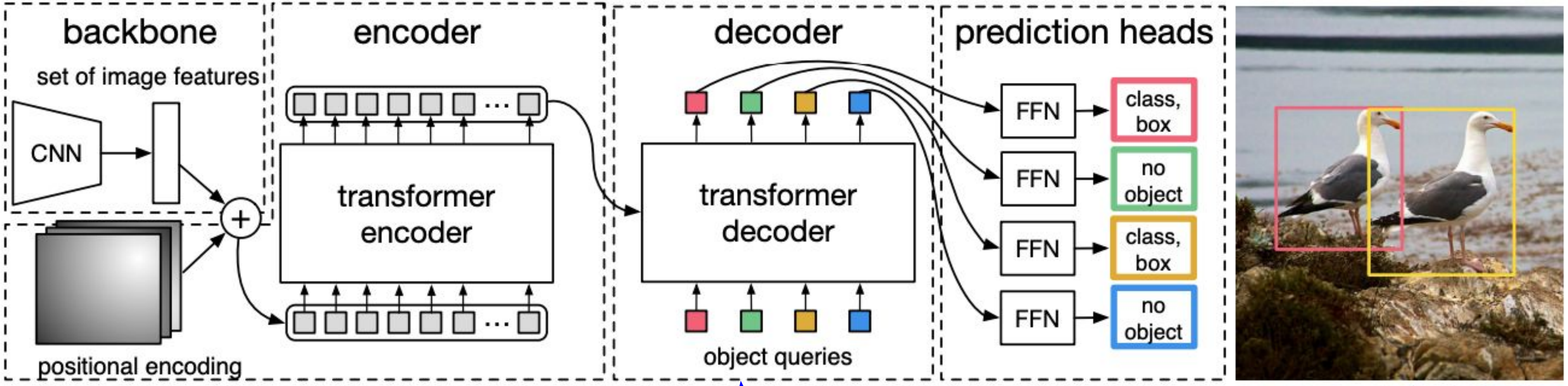


Image features are converted to sequence and input to encoder

Carion et al. 2020

DETR: Transformer-based models for more complex vision tasks

Transformer encoder-decoder part of the model is used for predicting a set of objects from the image features



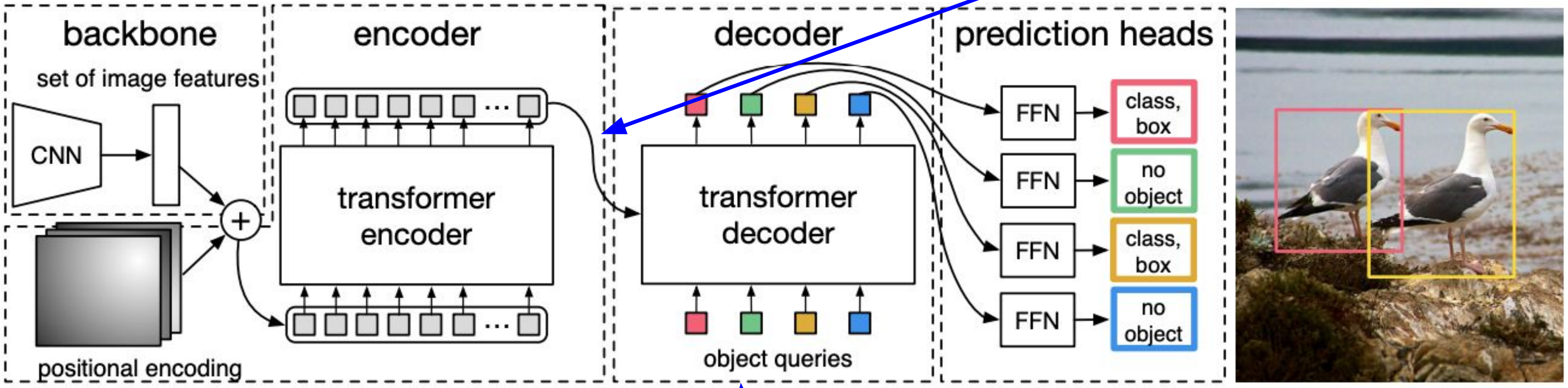
Decoder produces outputs that are fed into simple feedforward network for predicting bounding box locations and classes. Input to decoder is a set of learned positional encodings (can be understood as "object queries") => each one will correspond to a predicted box at output. Use N object queries > total expected number of boxes in the image, class output can also be "no object" to predict variable # of objects.

Carion et al. 2020

DETR: Transformer-based models for more complex vision tasks

Transformer encoder-decoder part of the model is used for predicting a set of objects from the image features

Encoder-decoder attention (remember how Transformer attention mechanism can be defined between any two sequences)



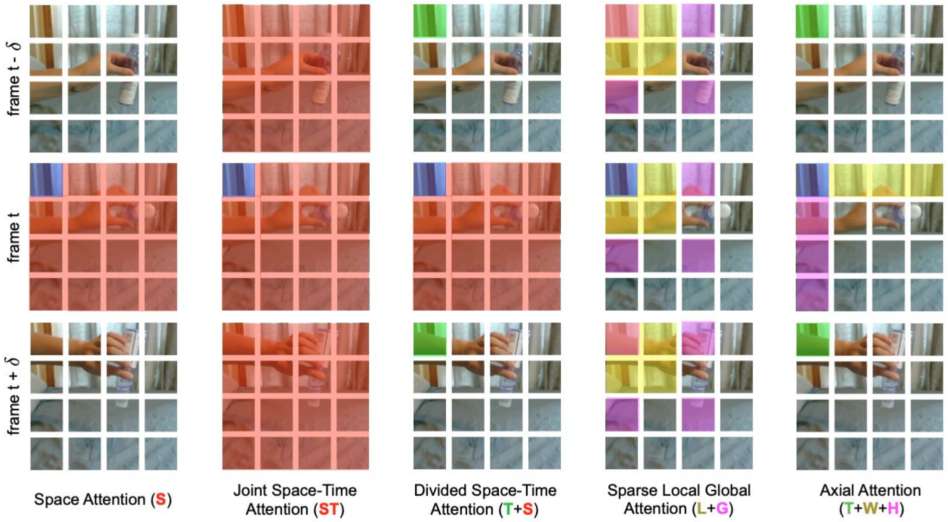
Decoder produces outputs that are fed into simple feedforward network for predicting bounding box locations and classes. Input to decoder is a set of learned positional encodings (can be understood as "object queries") => each one will correspond to a predicted box at output. Use N object queries > total expected number of boxes in the image, class output can also be "no object" to predict variable # of objects.

Carion et al. 2020

Transformers for video: attention across space-time

Example: TimeSformer model (Bertasius 2021)

Extension of what we have already seen, but can compute attention of query position (blue patch) over sequences corresponding to different neighborhoods (other colored patches)

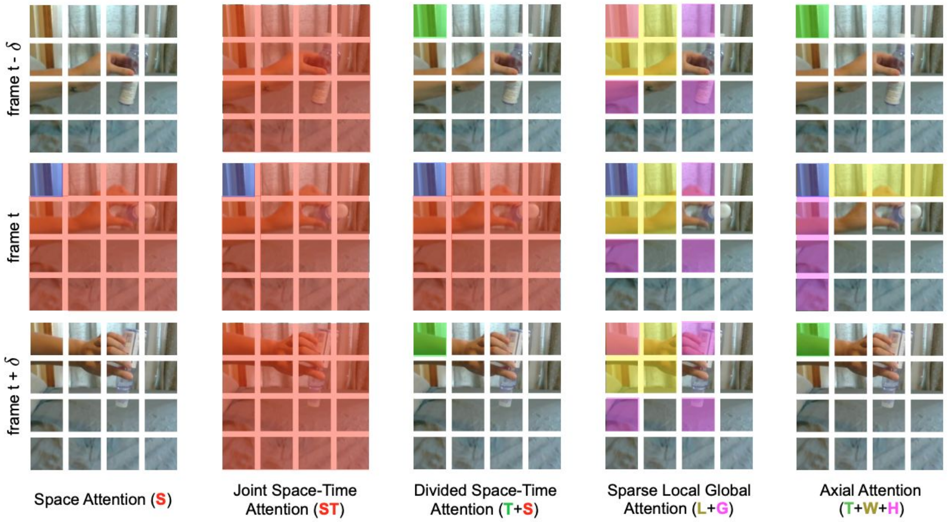


Bertasius et al. 2020

Transformers for video: attention across space-time

Example: TimeSformer model (Bertasius 2021)

Extension of what we have already seen, but can compute attention of query position (blue patch) over sequences corresponding to different neighborhoods (other colored patches)

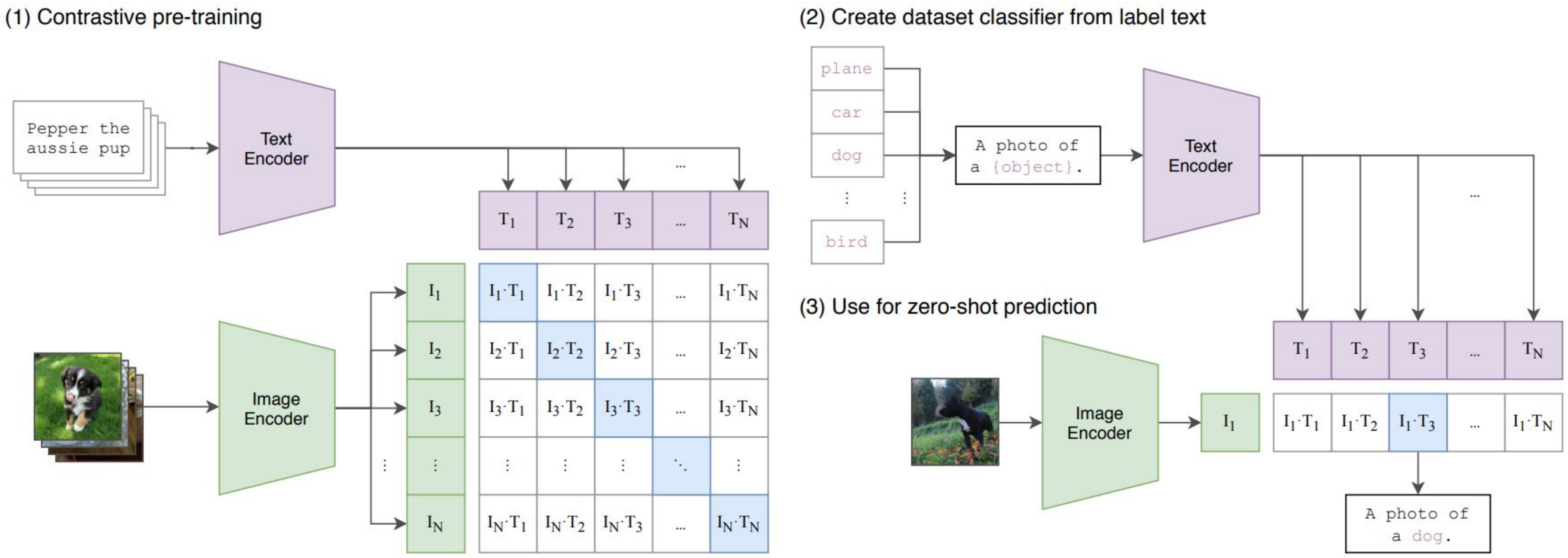


Standard image-level attention

Attention over different spatiotemporal neighborhoods

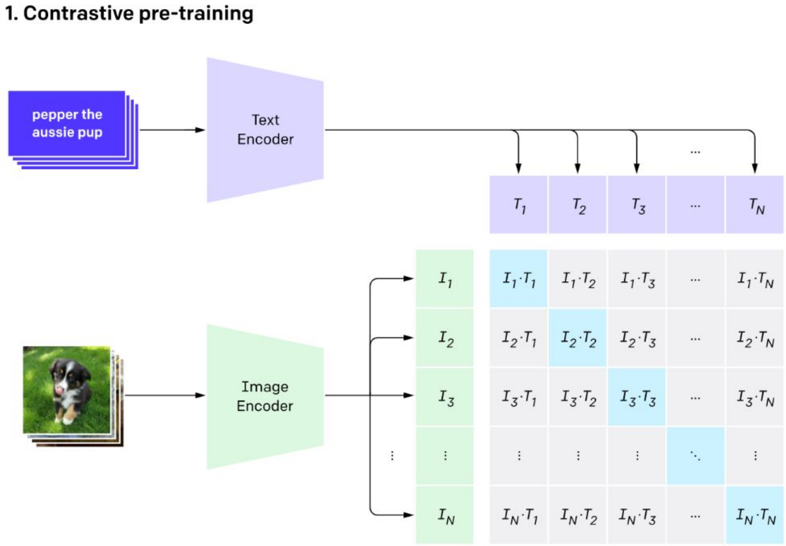
Bertasius et al. 2020

Let’s revisit multimodal models: CLIP



CLIP

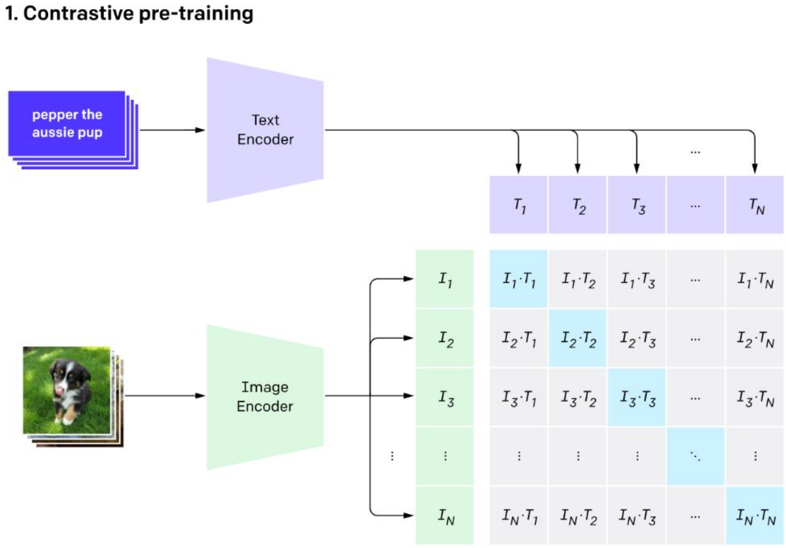
Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs



Radford et al. 2021.

CLIP

Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs



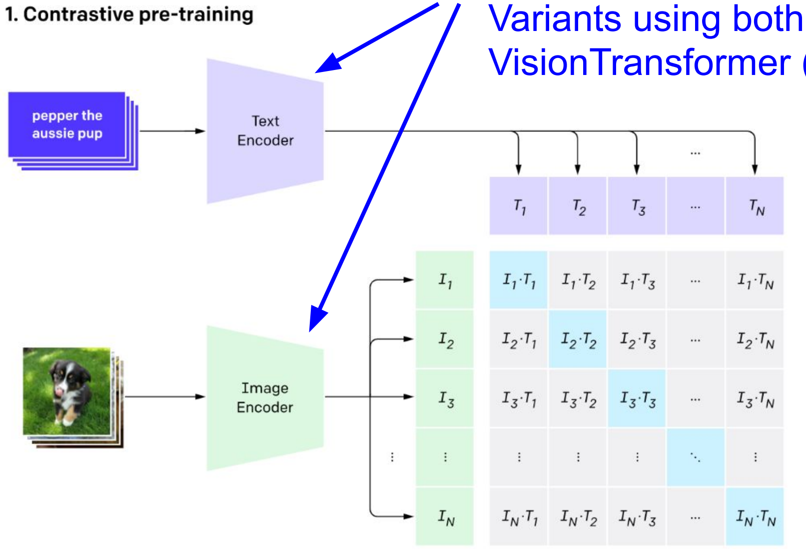
Dataset generated by searching for image-text pairs on the web, where text comes from a base query list of 500,000 queries comprising all words occurring at least 100 times in the English version of Wikipedia. This is augmented and processed in various ways, see paper for details.

Radford et al. 2021.

CLIP

Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Transformer-based, trained from scratch.
Variants using both ResNet and VisionTransformer (ViT) for image encoder.

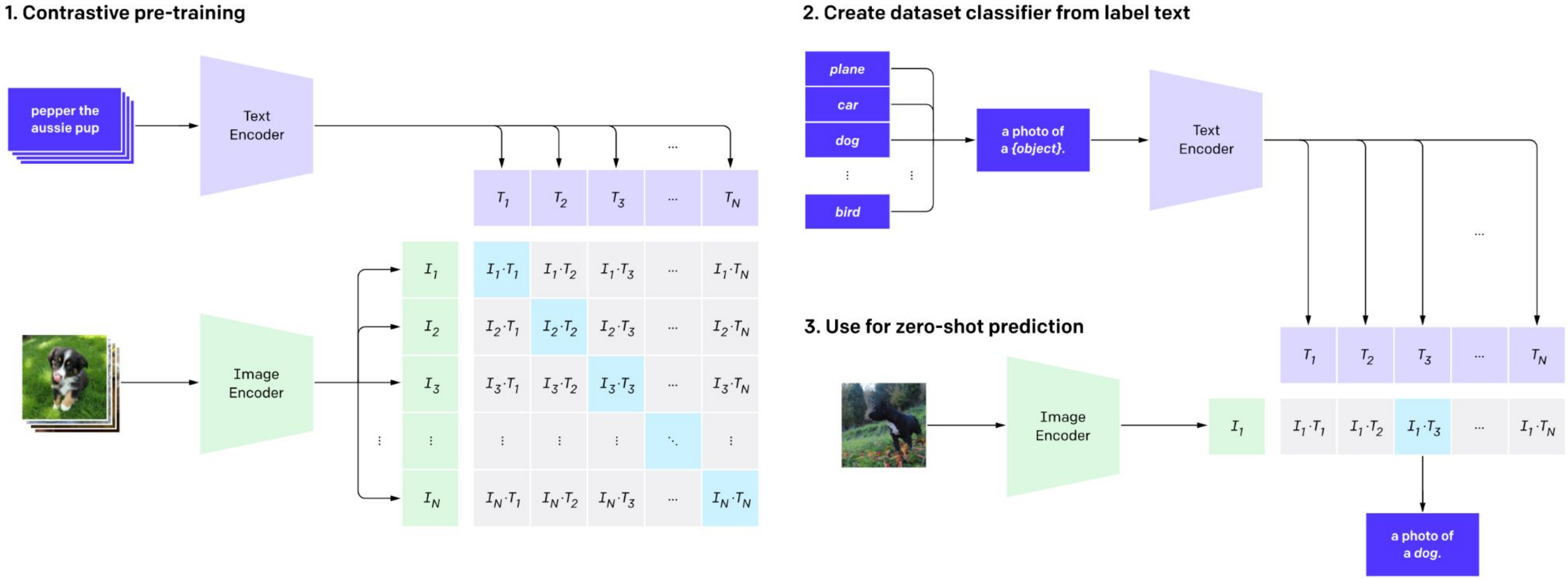


Radford et al. 2021.

CLIP

Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Can be used for **zero-shot** prediction tasks



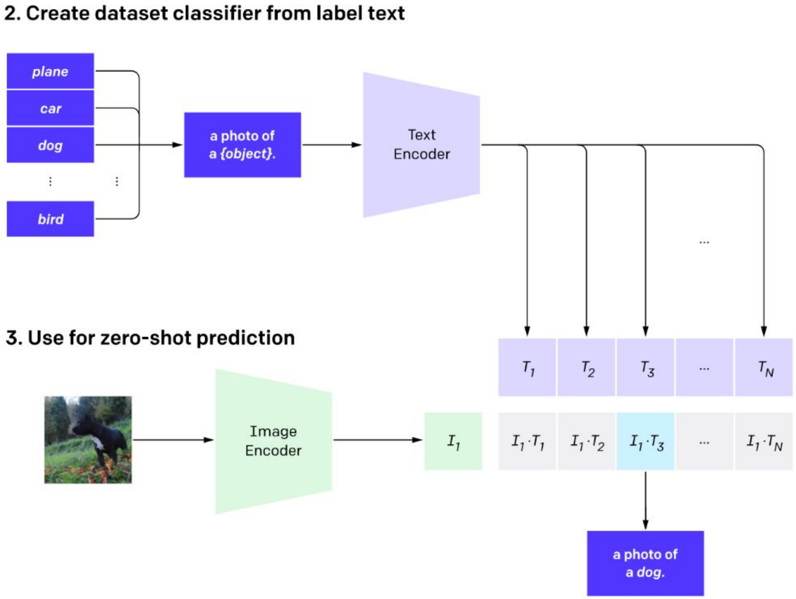
Radford et al. 2021.

CLIP

Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Zero-shot classification: Perform N-way classification without showing the model any paired examples of (input, class) for any of the N classes

Can be used for **zero-shot** prediction tasks



Radford et al. 2021.

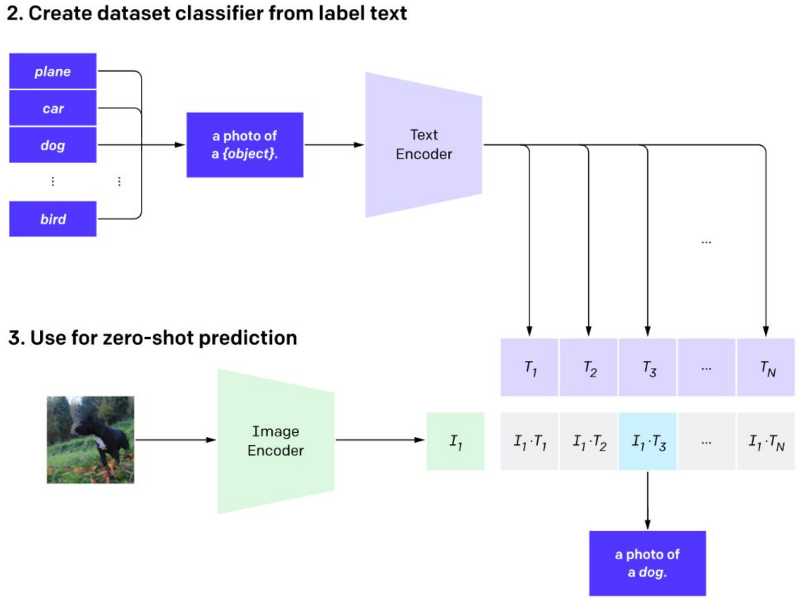
CLIP

Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Zero-shot classification: Perform N-way classification without showing the model any paired examples of (input, class) for any of the N classes

Steps to perform zero-shot image classification, given a trained CLIP model:

Can be used for **zero-shot** prediction tasks



Radford et al. 2021.

CLIP

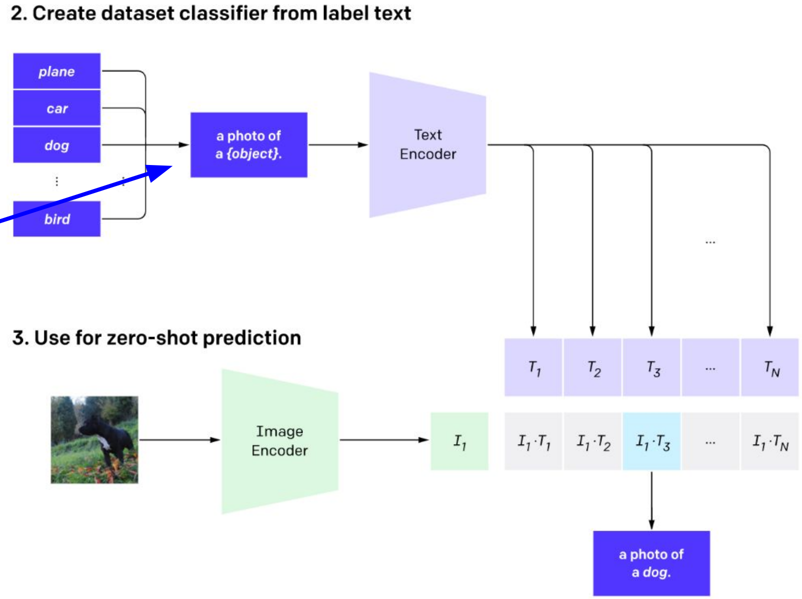
Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Can be used for **zero-shot** prediction tasks

Zero-shot classification: Perform N-way classification without showing the model any paired examples of (input, class) for any of the N classes

Steps to perform zero-shot image classification, given a trained CLIP model:

1. Generate text prompts corresponding to each of the N classes



Radford et al. 2021.

CLIP

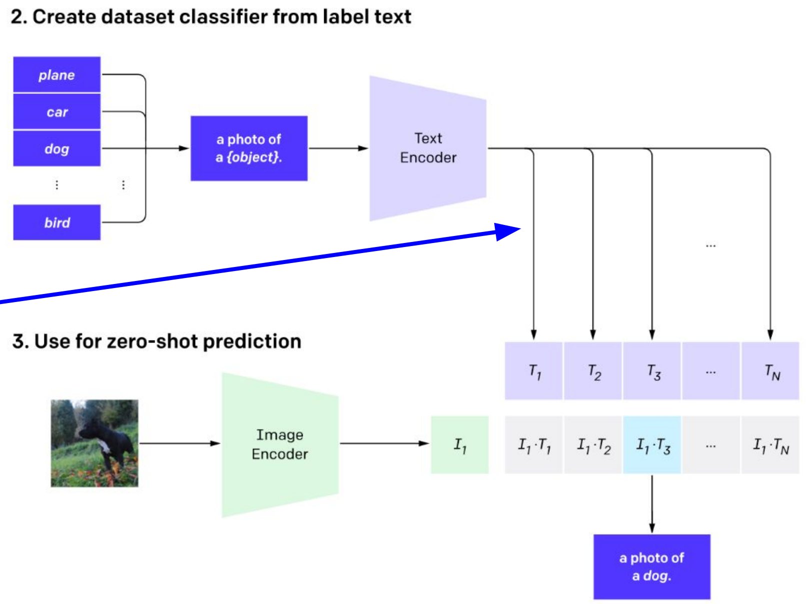
Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Can be used for **zero-shot** prediction tasks

Zero-shot classification: Perform N-way classification without showing the model any paired examples of (input, class) for any of the N classes

Steps to perform zero-shot image classification, given a trained CLIP model:

1. Generate text prompts corresponding to each of the N classes
2. Using the CLIP text encoder to obtain embedding vectors for each text prompt



Radford et al. 2021.

CLIP

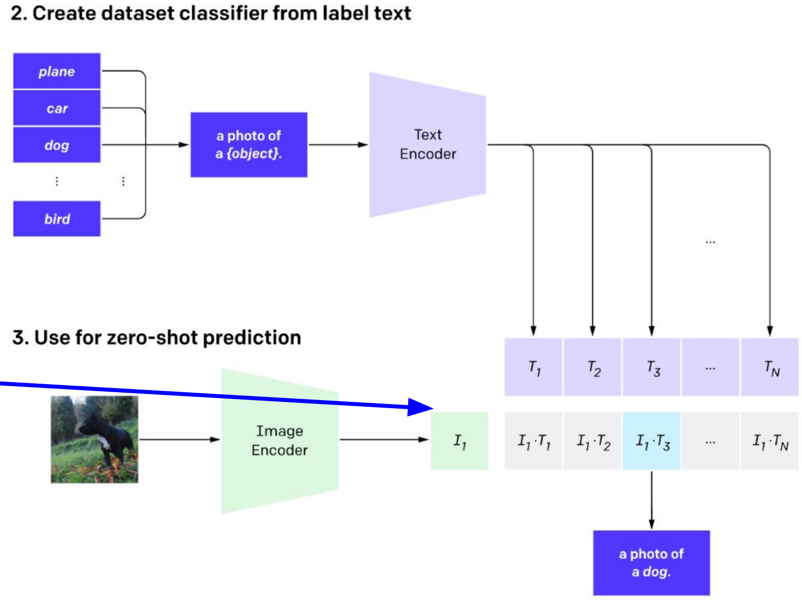
Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Zero-shot classification: Perform N-way classification without showing the model any paired examples of (input, class) for any of the N classes

Steps to perform zero-shot image classification, given a trained CLIP model:

1. Generate text prompts corresponding to each of the N classes
2. Using the CLIP text encoder to obtain embedding vectors for each text prompt
3. Using the CLIP image encoder to obtain an embedding vector for the image to classify

Can be used for **zero-shot** prediction tasks



Radford et al. 2021.

CLIP

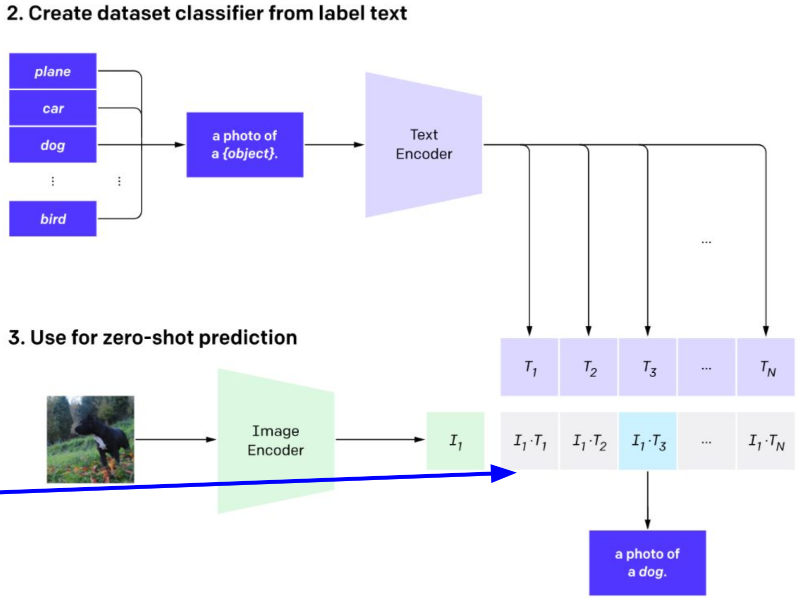
Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Zero-shot classification: Perform N-way classification without showing the model any paired examples of (input, class) for any of the N classes

Steps to perform zero-shot image classification, given a trained CLIP model:

1. Generate text prompts corresponding to each of the N classes
2. Using the CLIP text encoder to obtain embedding vectors for each text prompt
3. Using the CLIP image encoder to obtain an embedding vector for the image to classify
4. Compare similarity of the image embedding with each of the text prompt embeddings

Can be used for **zero-shot** prediction tasks



Radford et al. 2021.

CLIP

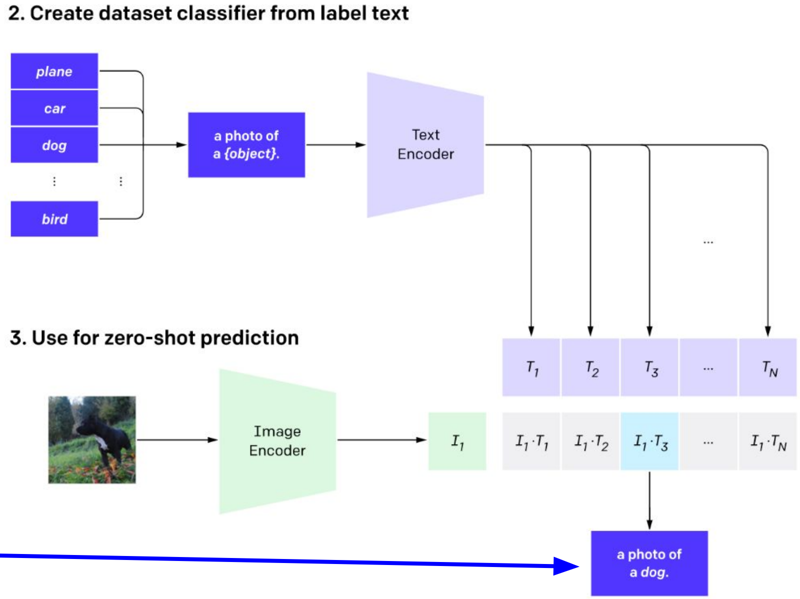
Multimodal contrastive learning similar to ConVIRT, but now on very large dataset of 400 million image-text pairs

Can be used for **zero-shot** prediction tasks

Zero-shot classification: Perform N-way classification without showing the model any paired examples of (input, class) for any of the N classes

Steps to perform zero-shot image classification, given a trained CLIP model:

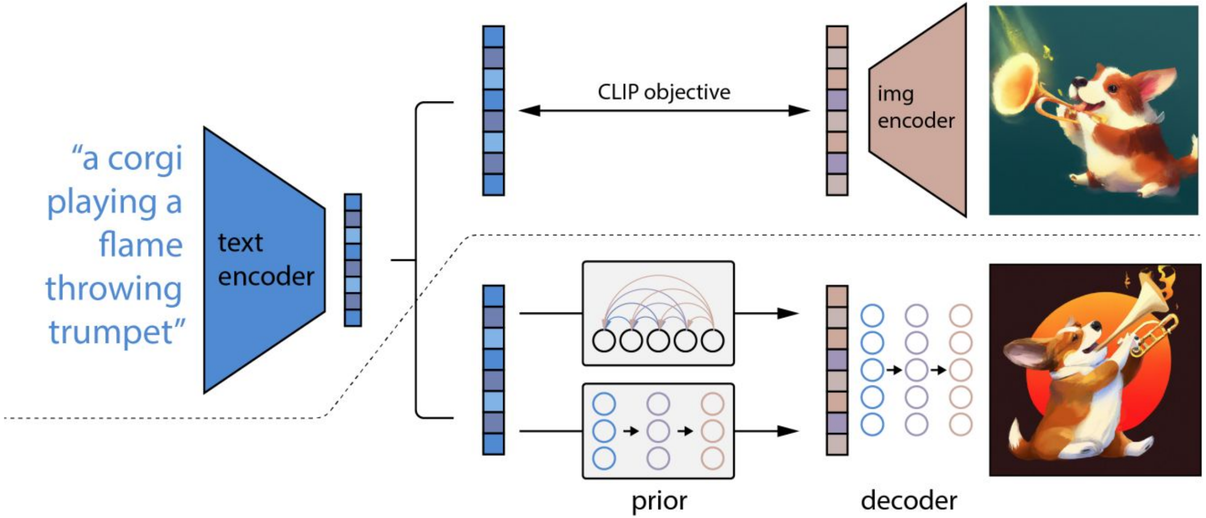
1. Generate text prompts corresponding to each of the N classes
2. Using the CLIP text encoder to obtain embedding vectors for each text prompt
3. Using the CLIP image encoder to obtain an embedding vector for the image to classify
4. Compare similarity of the image embedding with each of the text prompt embeddings
5. Assign the image class label to the one associated with the most similar text prompt



Radford et al. 2021.

Building off CLIP to perform text-to-image generation: DALL-E, DALL-E 2

Given CLIP model, train text-to-image generation model (bottom pathway) that goes from text input -> CLIP text embedding -> CLIP image embedding (through a "prior" network that learns this mapping -> generative model that decodes from image embedding to generated image



Ramesh et al. 2022.

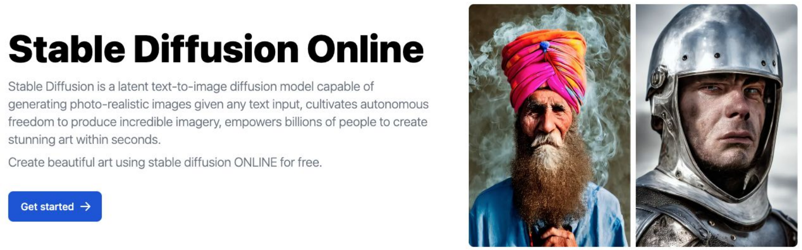
Aside: Transformer-based text-to-image generation models are an active area of ongoing work

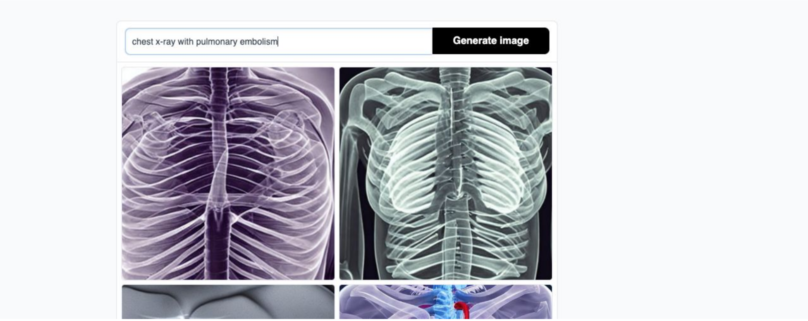
Google Imagen.
Saharia et al. 2022.

Teddy bears swimming at the Olympics 400m Butterfly event.

A transparent sculpture of a duck made out of glass. The sculpture is in front of a painting of a landscape.

A Pomeranian is sitting on the Kings throne wearing a crown. Two tiger soldiers are standing next to the throne.

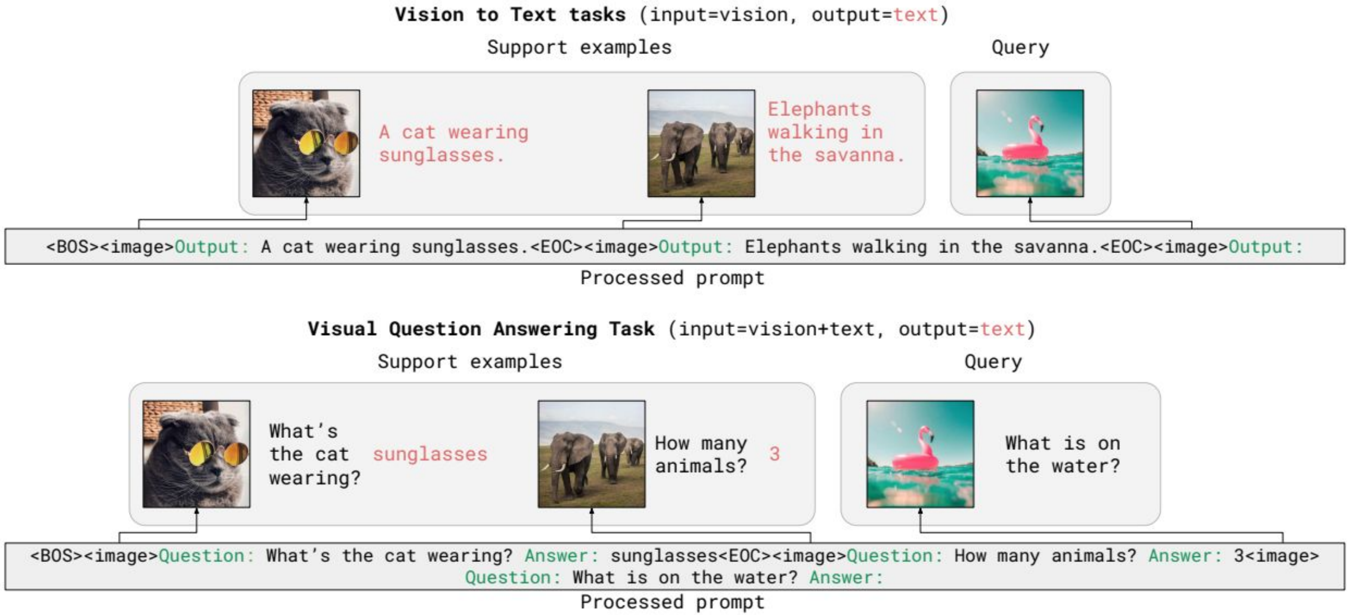




Meta Make-a-Video.
Singer et al. 2022

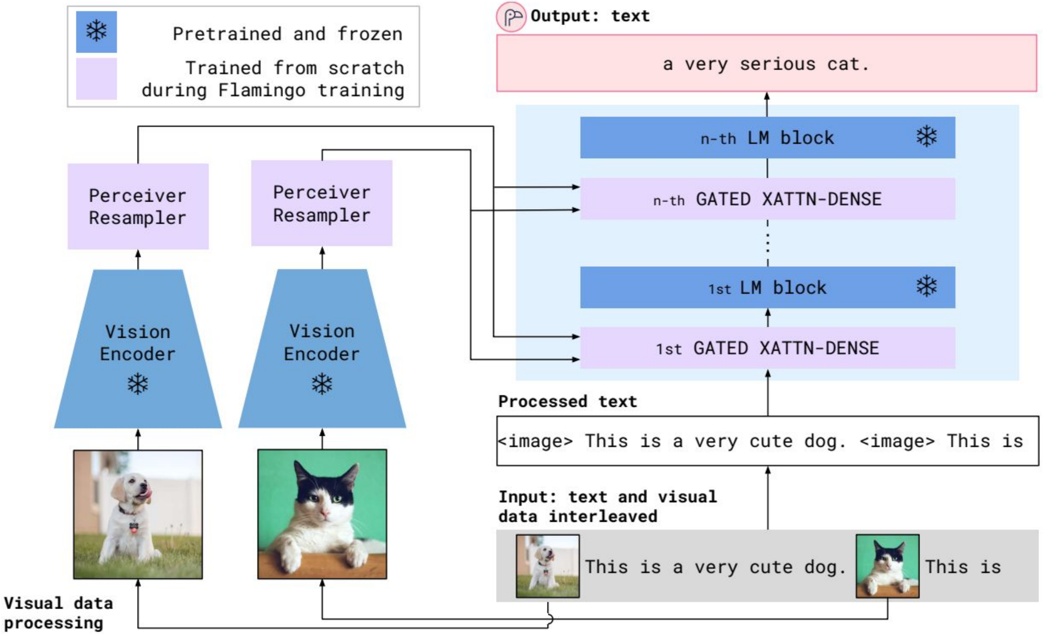
Stable Diffusion.
https://stablediffusionweb.com/

Next-gen text generation models that take multimodal interleaved data as input: Flamingo



Alayrac et al. 2022.

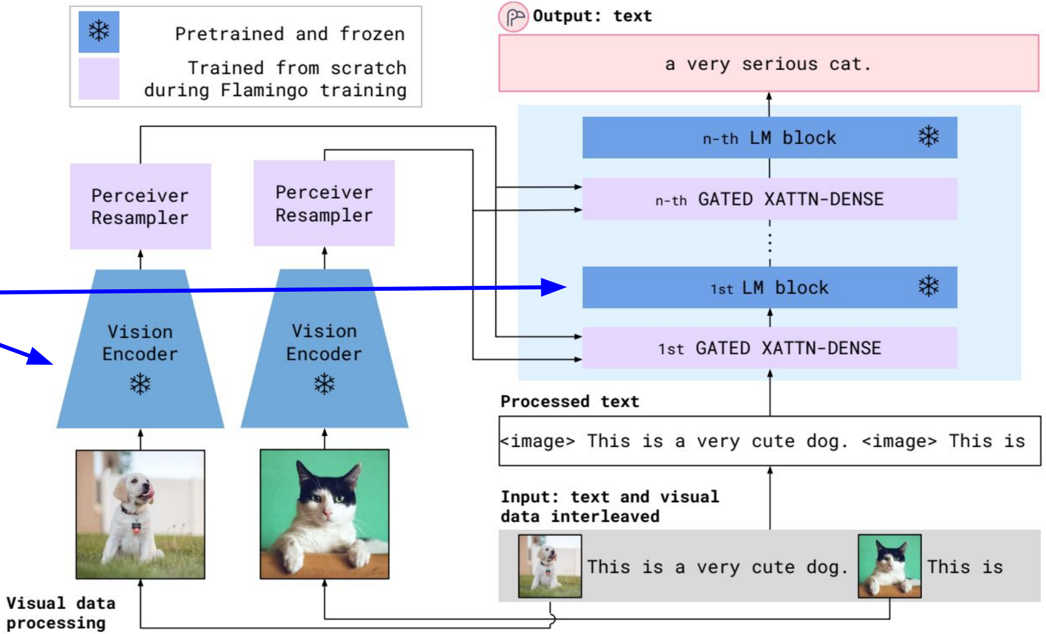
Next-gen text generation models that take multimodal interleaved data as input: Flamingo



Alayrac et al. 2022.

Next-gen text generation models that take multimodal interleaved data as input: Flamingo

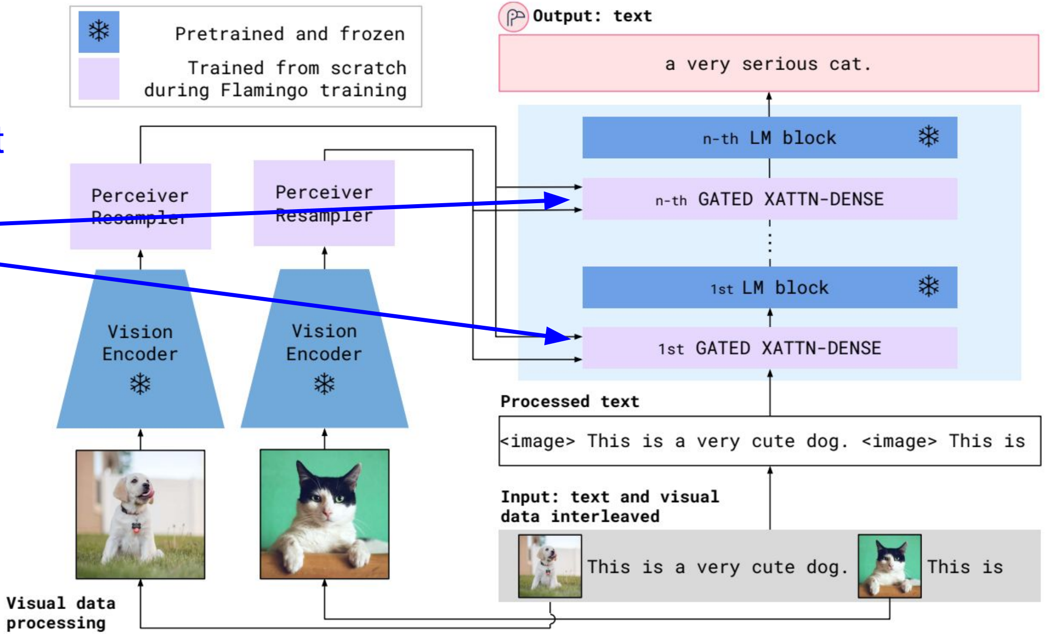
Pre-trained vision and language encoders



Alayrac et al. 2022.

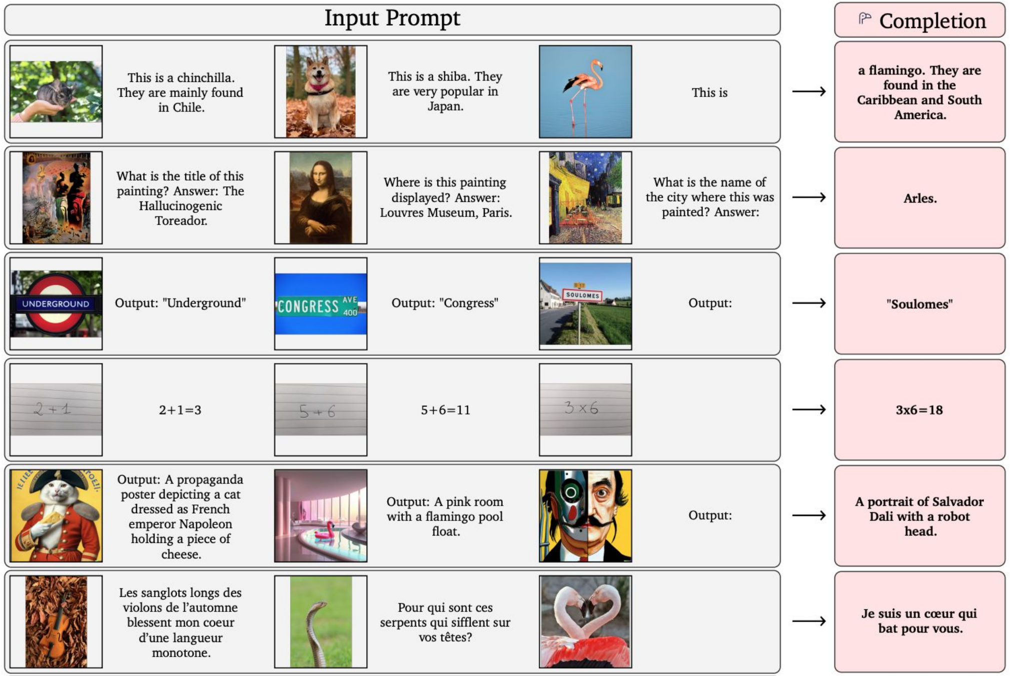
Next-gen text generation models that take multimodal interleaved data as input: Flamingo

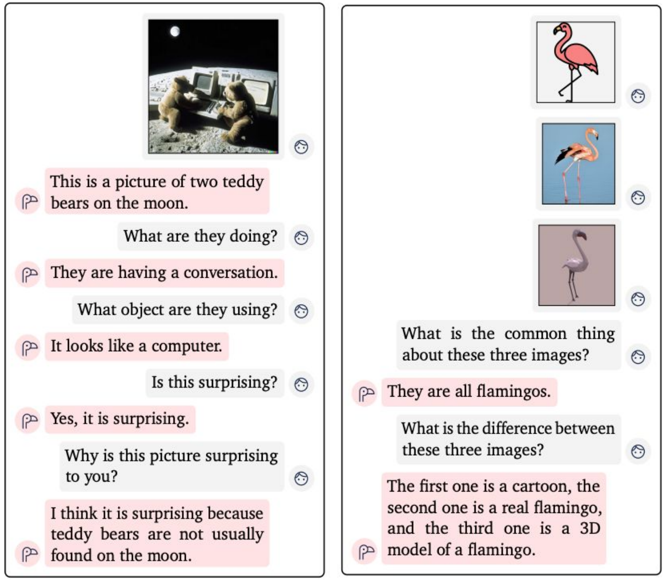
Learn new attention layers that merge visual and language information



Alayrac et al. 2022.

Next-gen text generation models that take multimodal interleaved data as input: Flamingo





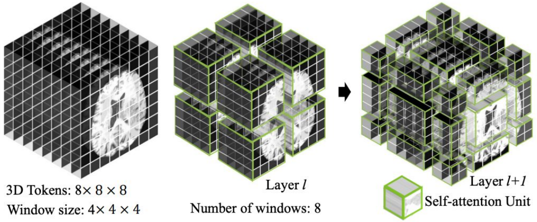
Alayrac et al. 2022.

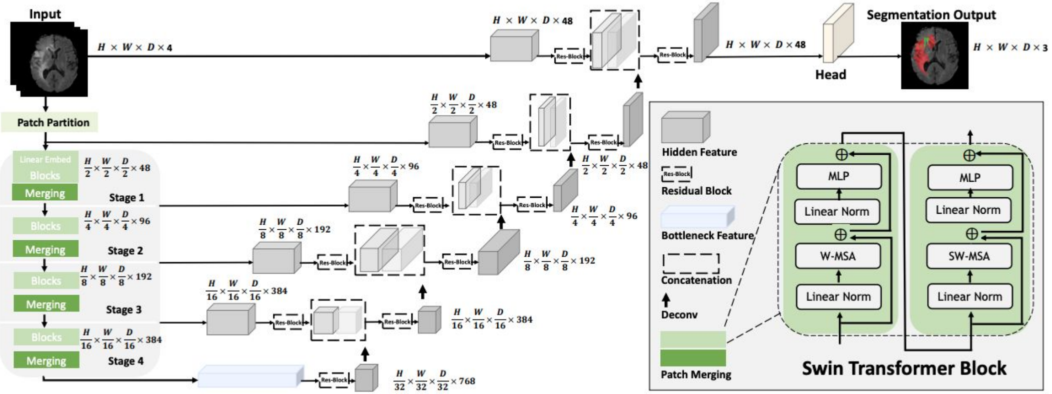
Transformer-based models in biomedical applications

In medical NLP, Transformer-based models like BERT already widespread (saw in previous lecture)

In medical computer vision, Transformer-based models seeing increasing usage and success across various tasks. Particularly segmentation, e.g. Swin UNETR (Hatamizadeh et al. 2022)

Swin UNETR Shifted Windows mechanism (refer to Swin Transformer discussion)





Swin UNETR Architecture

Hatamizadeh et al. 2022.

Transformer-based models in biomedical applications

Many open questions remain, e.g.:

- Can GPT-3 be used to perform zero-shot / few-shot medical reasoning tasks?
  - Related question: Unclear how much biomedical information is represented in the training data of these models
- Can we effectively fine-tune CLIP for biomedical domains?
- Do current text-to-image generation models work for biomedical prompts or can we effectively adapt them to do so?

Transformer-based models in biomedical applications

Many open questions remain, e.g.:

- Can GPT-3 be used to perform zero-shot / few-shot medical reasoning tasks?
  - Related question: Unclear how much biomedical information is represented in the training data of these models
- Can we effectively fine-tune CLIP for biomedical domains?
- Do current text-to-image generation models work for biomedical prompts or can we effectively adapt them to do so?

Another major open question / challenge around large language and vision models more generally: what biases are captured in the data / model, how this affects downstream ethical use, etc. Will talk more about bias and fairness in a later lecture.

Summary

Today we covered:

- More on Transformers: Encoder-based, decoder-based, and encoder-decoder models
- Transformers for computer vision tasks
- More discussion of Transformers used in different types of multimodal models
- Very large models like GPT-3 and CLIP are also being explored for zero-shot / few-shot prediction tasks
- Use in biomedical applications still very early, but expect to see much more in future. Also many open questions that remain

Next lecture: Genomics: Introduction