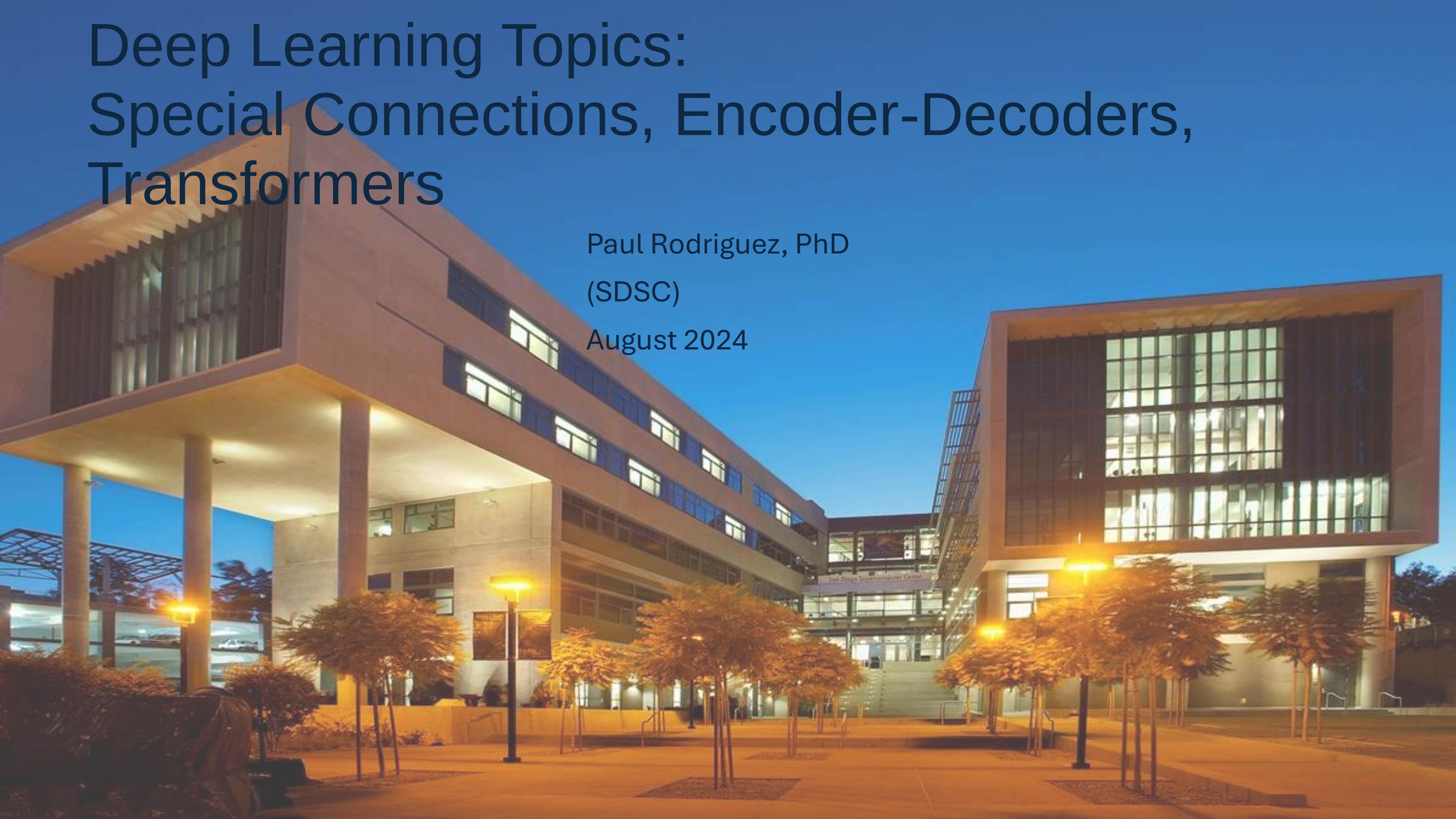


Deep Learning Topics: Special Connections, Encoder-Decoders, Transformers

Paul Rodriguez, PhD

(SDSC)

August 2024

Outline

- Basic word prediction task and motivating the attention strategy
- From Embeddings and Attention Head to Transformers
- BERT and GPT strategies
- Transformers in Science Applications

Dependences of Language

Consider this sequence:

The Law will never be perfect, but it's application should be just - this is what we are missing, in my opinion <End of Sequence>

What does 'it' refer to that can have an 'application'?

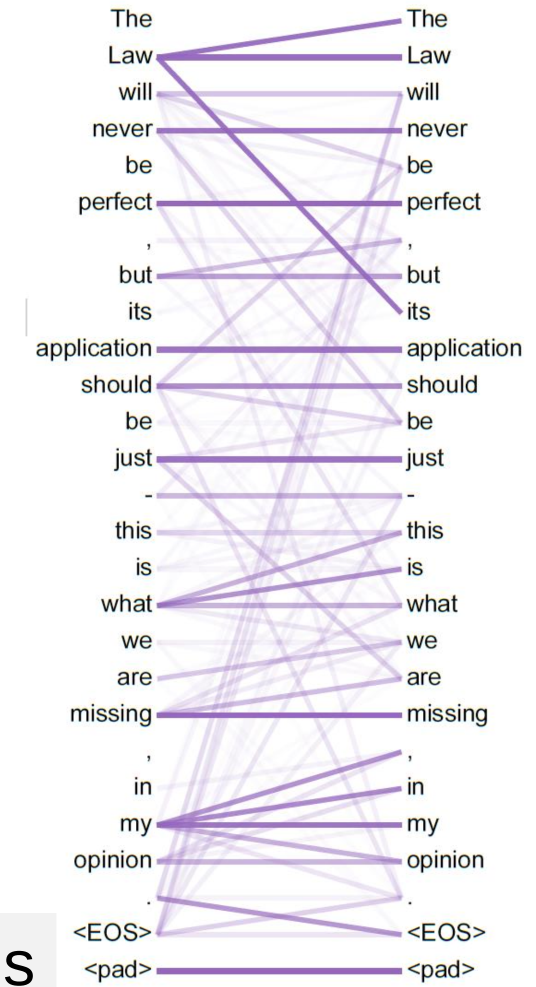
Dependences of Language

Consider this sequence:

The Law will never be perfect, but it's application should be just - this is what we are missing, in my opinion <End of Sequence>

What does 'it' refer to that can have an 'application'?

e.g 'it' refers back to 'Law', which is part of 'the Law' noun phrase, which is the entity that will 'never be perfect', and so on …



many dependencies and interactions

A toy problem to get some intuition

- Let's use the following list of 5 tokens (ie words):
  <**s**tart>, the, man, chicken, ordered

- Let's use this sequence of 6 tokens as our only sentence:
  <start> the man ordered the chicken

- If we use **token** ids 1 to 5, the sequence is [1,2,3,5,2,4]

- **Now let's try to predict the next word by 'attention' idea**

The toy task: predict next word

The data: 5 tokens (V=5),
1 sequence (length=T=6):   <Start> the man ordered the chicken

A basic solution is a bigram matrix:
X=Sequence-to-Word, size is TxV

Next Token (ie word) Prediction

Token Sequence

Pos	Word	<strt>	The	Man	Chikn	Order.
0	<start>		1.0			
1	The			0.5	0.5	
2	Man					1.0
3	Orde.r		1.0			
4	The			0.5	0.5	
5	Chick.	1.0				

The toy task: predict next word

The data: 5 tokens (V=5),

1 sequence (length=T=6):   <Start> the man ordered the chicken

A basic solution is a bigram matrix:

X=Sequence-to-Word, size is TxV

Challenge, can we learn predictions (→) that depend on context of other tokens and/or position

After   <Start>   the→man = 1.0

After 'Ordered'   the →chicken = 1.0

Next Token (ie word) Prediction

Token Sequence

Pos	Word	<strt>	The	Man	Chikn	Order .
0	<start>		1.0			
1	The			0.5	0.5	
2	Man					1.0
3	Orde.r		1.0			
4	The			0.5	0.5	
5	Chick.	1.0				

The attention idea

Let’s get all tokens to ‘pass information’ about dependencies

E.G. for *X a TxV* matrix of bigrams, we want to transform *X*

X= Sequence-to-Word is TxV

$$
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0.5 & 0.5 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0.5 & 0.5 & 0 \\
1.0 & 0 & 0 & 0 & 0
\end{pmatrix}
\rightarrow\rightarrow
\begin{pmatrix}
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 \\
0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 \\
1 & 0 & 0 & 0 & 0
\end{pmatrix}
$$

The attention idea

Let's get all tokens to 'pass information' about dependencies

E.G. for *X a TxV* matrix of bigrams, we want to transform *X*

we want *W a TxT* matrix – aka 'attention' weights

X= Sequence-to-Word is TxV

W dependencies is TxT

$$\begin{pmatrix} w_{11} & \cdots & w_{1T} \\ \vdots & \vdots & \vdots \\ w_{T1} & \cdots & w_{TT} \end{pmatrix} * \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 \\ 1.0 & 0 & 0 & 0 & 0 \end{pmatrix} \rightarrow\rightarrow \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

The attention idea

Let's get all tokens to 'pass information' about dependencies

E.G. for *X a TxV* matrix of bigrams, we want to transform *X*

we want *W a TxT* matrix – aka 'attention' weights

X= Sequence-to-Word is TxV

W dependencies is TxT

$$\begin{pmatrix} w_{11} & \cdots & w_{1T} \\ \vdots & \vdots & \vdots \\ w_{T1} & \cdots & w_{TT} \end{pmatrix} * \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0.5 & 0.5 & 0 \\ 1.0 & 0 & 0 & 0 & 0 \end{pmatrix} \rightarrow\rightarrow \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

W should reflect the interdependencies of the sequence.

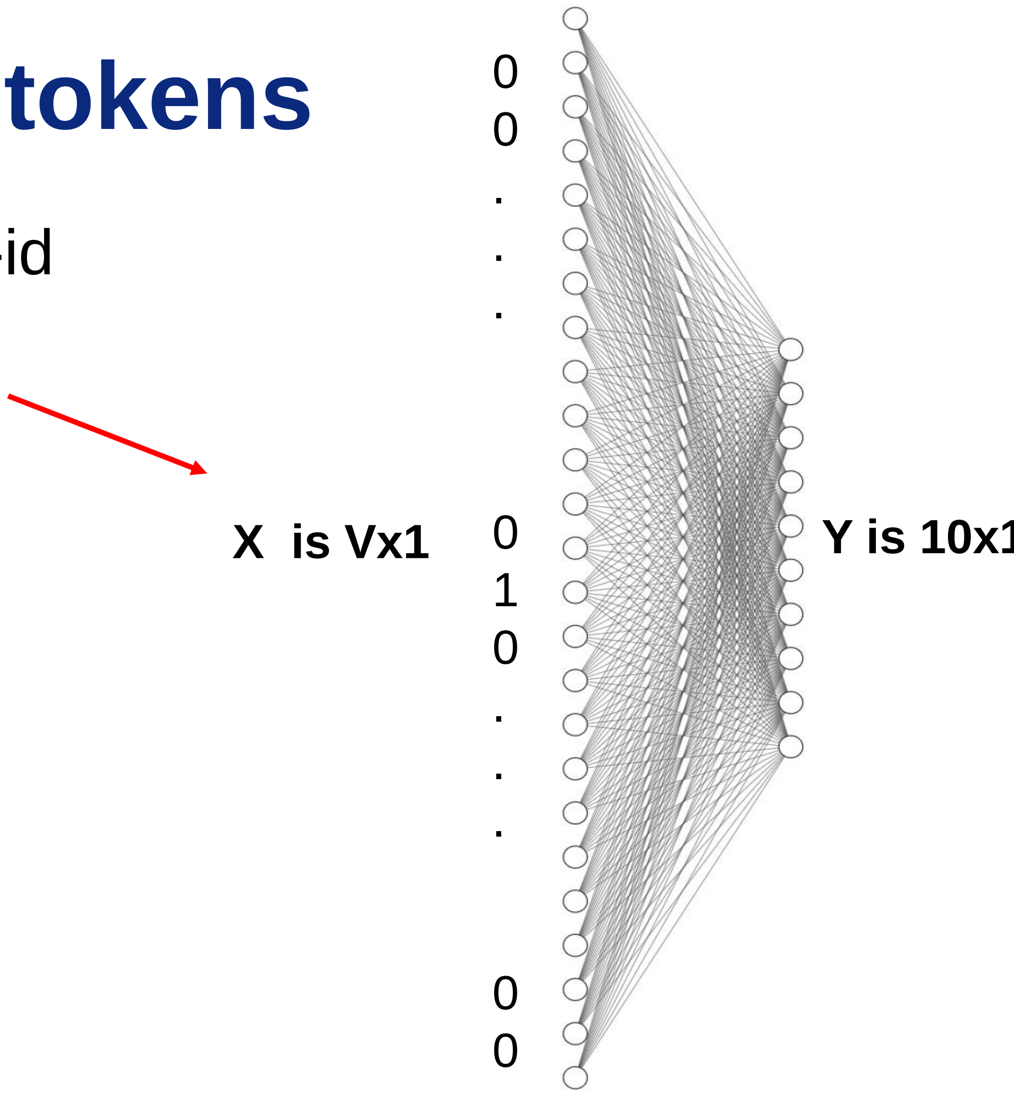
Q: Where should W come from?

- **Let’s build up the attention architecture**

An embedding layer for tokens

1. For an input vector X of V elements let the token-id determine which element is 1, the rest are 0.

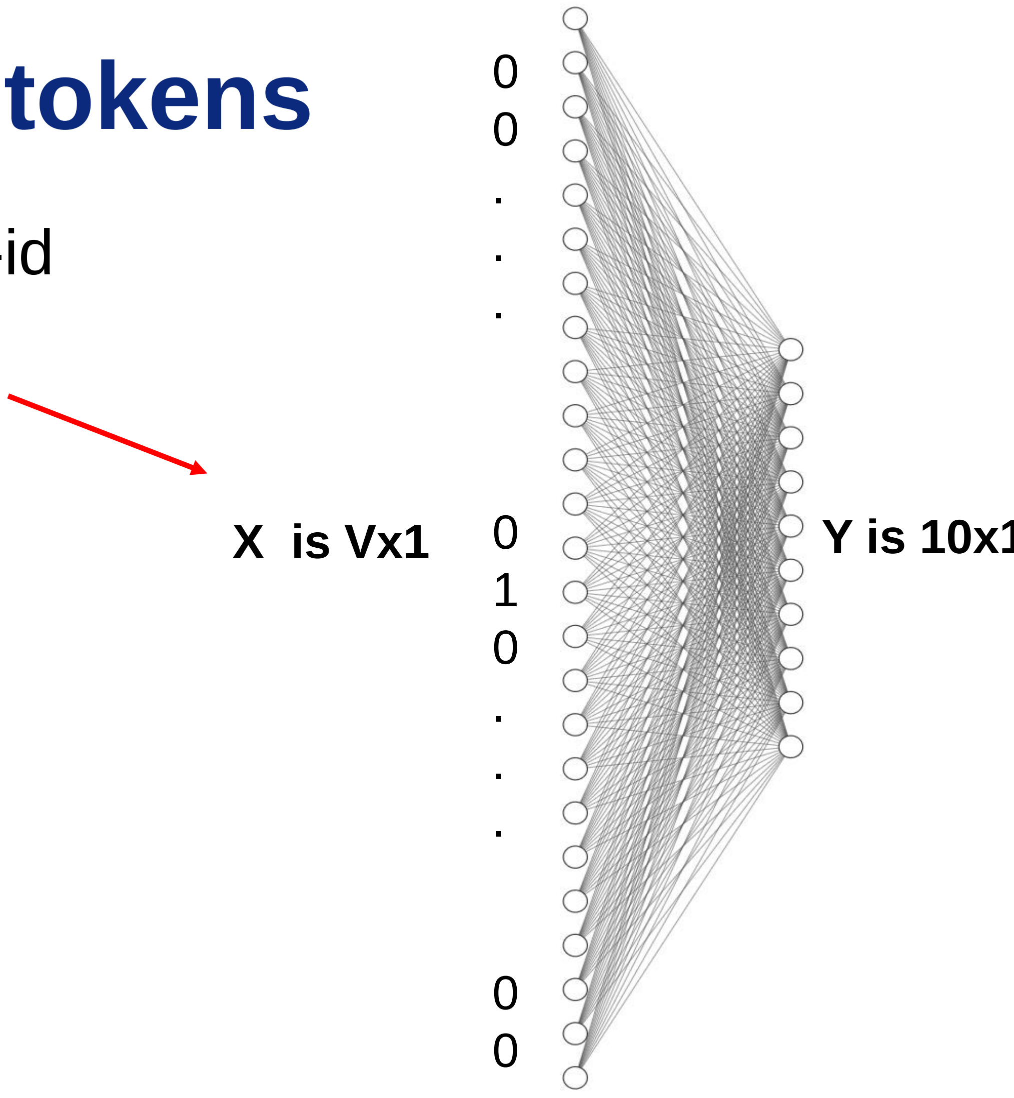
2. Use a single, smaller, hidden layer with linear activation, i.e: $Y = W * X$



An embedding layer for tokens

1. For an input vector X of V elements let the token-id determine which element is 1, the rest are 0.

2. Use a single, smaller, hidden layer with linear activation, i.e: $Y = W * X$



Thus, each token id is converted to a lower dimensional vector with size 1xE

Each token sequence of vectors is TxE

Get input embeddings

First, get sequence of token embeddings (call it X)

[1,2,3,5,2,4] → X_{TxE}

Get input and add in position info

First, get sequence of token embeddings (call it X)

[1,2,3,5,2,4] → X_{TxE}

Then do the same for positions 1…T

[1,2,3,4,5,6] → P_{TxE}

X + P is final TxE matrix of input embeddings

Embeddings for attention weights

Take Input Embeddings and build a ‘Query’ (Q) and ‘Key’ (K) embedding matrix *of size TxE*

$$X + P \rightarrow Q_{TxE}$$

$$X + P \rightarrow K_{TxE}$$

Recall that embedding layers helps transform inputs into lower dimensional representations that capture information

Embeddings for attention weights

Take Input Embeddings and build a 'Query' (Q) and 'Key' (K) embedding matrix *of size TxE*

$X + P \rightarrow Q_{TxE}$

$X + P \rightarrow K_{TxE}$

Recall that embedding layers helps transform inputs into lower dimensional representations that capture information

Now let $W = Q*K'$

Notice that every token's embedding gets to 'interact' with every other token's embedding to make up the *TxT* elements of ***W***

Embeddings for attention weights

Take Input Embeddings and build a 'Query' (Q) and 'Key' (K) embedding matrix *of size TxE*

$X + P \rightarrow Q_{TxE}$

$X + P \rightarrow K_{TxE}$

Recall that embedding layers helps transform inputs into lower dimensional representations that capture information

Now let $W = Q^*K'$

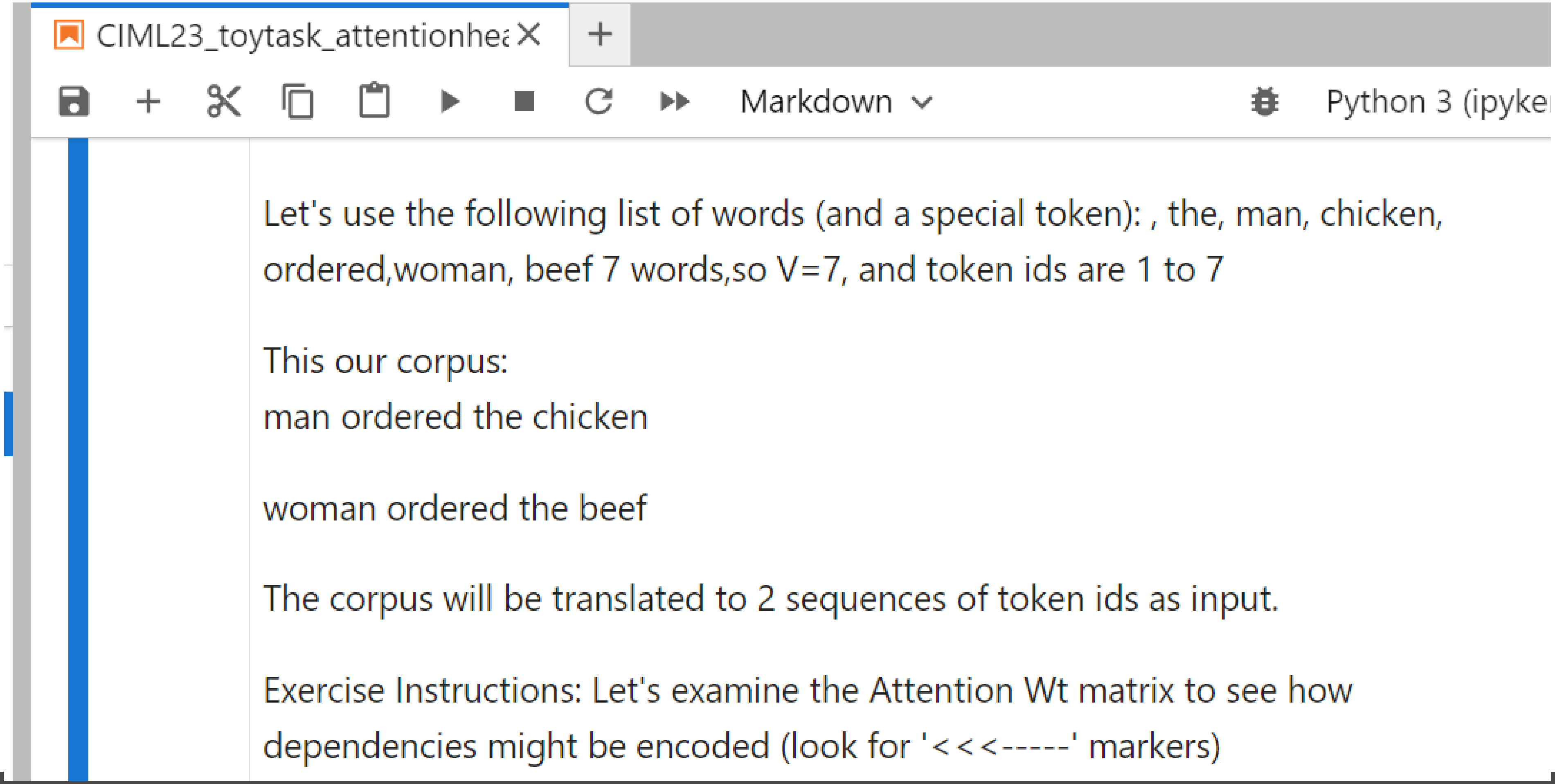
Notice that every token's embedding gets to 'interact' with every other token's embedding to make up the *TxT* elements of ***W***

$X + P \rightarrow V_{TxE}$

Finally, instead of a pre-built bigram matrix, use another embedding for a 'Value' V matrix

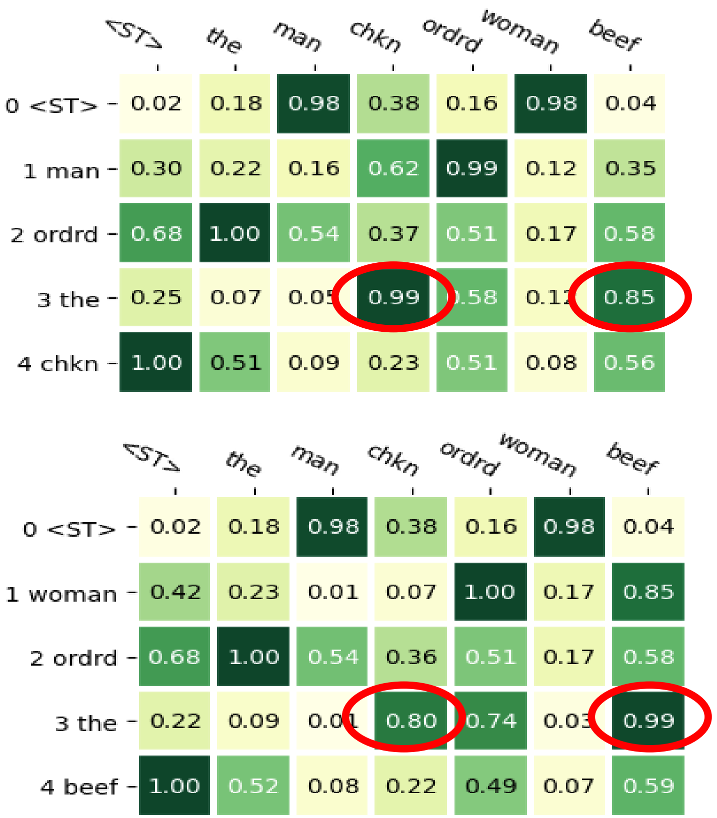
*Now, we can take W * V*

An example of attention head with a toy task:
Run the “toytask_attention notebook” and observe the printed predictions and attention weights.   Try changing H – does it help/hurt?



Output TxV predictions:

Notice that the→ [chicken or beef] predictions change depending on who's ordering



Output TxV predictions:

Notice that the→ [chicken or beef] predictions change depending on who's ordering

	<ST>	the	man	chkn	ordrd	woman	beef
0 <ST>	0.02	0.18	0.98	0.38	0.16	0.98	0.04
1 man	0.30	0.22	0.16	0.62	0.99	0.12	0.35
2 ordrd	0.68	1.00	0.54	0.37	0.51	0.17	0.58
3 the	0.25	0.07	0.05	0.99	0.58	0.12	0.85
4 chkn	1.00	0.51	0.09	0.23	0.51	0.08	0.56

	<ST>	the	man	chkn	ordrd	woman	beef
0 <ST>	0.02	0.18	0.98	0.38	0.16	0.98	0.04
1 woman	0.42	0.23	0.01	0.07	1.00	0.17	0.85
2 ordrd	0.68	1.00	0.54	0.36	0.51	0.17	0.58
3 the	0.22	0.09	0.01	0.80	0.74	0.03	0.99
4 beef	1.00	0.52	0.08	0.22	0.49	0.07	0.59

TxT Attn Wts:

Notice weights for "the" are different for 'man" or "woman"

	0 <ST>	1 man	2 ordrd	3 the	4 chkn
0 <ST>	1.00	0.00	0.00	0.00	0.00
1 man	0.06	0.94	0.00	0.00	0.00
2 ordrd	0.06	0.00	0.93	0.00	0.00
3 the	0.01	0.78	0.01	0.20	0.00
4 chkn	0.02	0.01	0.04	0.04	0.90

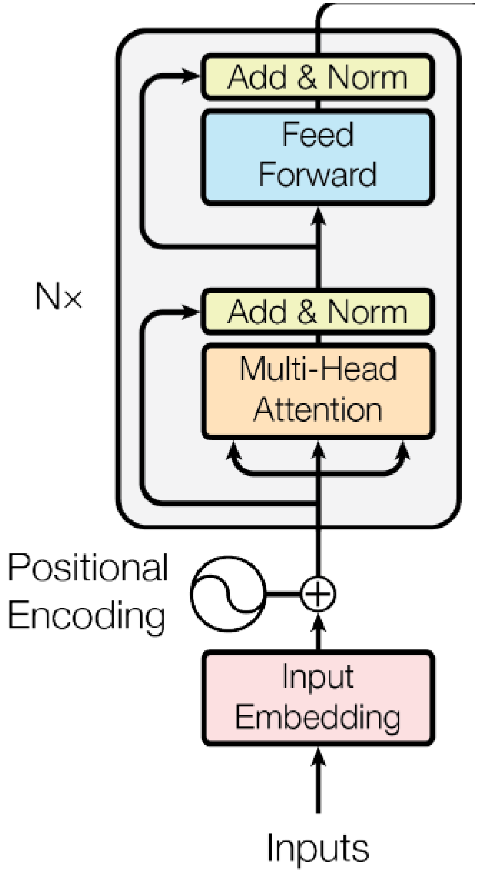
	0 <ST>	1 woman	2 ordrd	3 the	4 beef
0 <ST>	1.00	0.00	0.00	0.00	0.00
1 woman	0.10	0.90	0.00	0.00	0.00
2 ordrd	0.06	0.00	0.93	0.00	0.00
3 the	0.01	0.87	0.01	0.12	0.00
4 beef	0.01	0.02	0.04	0.04	0.89

- **pause**

Finally, a transformer

Include skip-add connections

Include Layer Normalization or DropOut layers

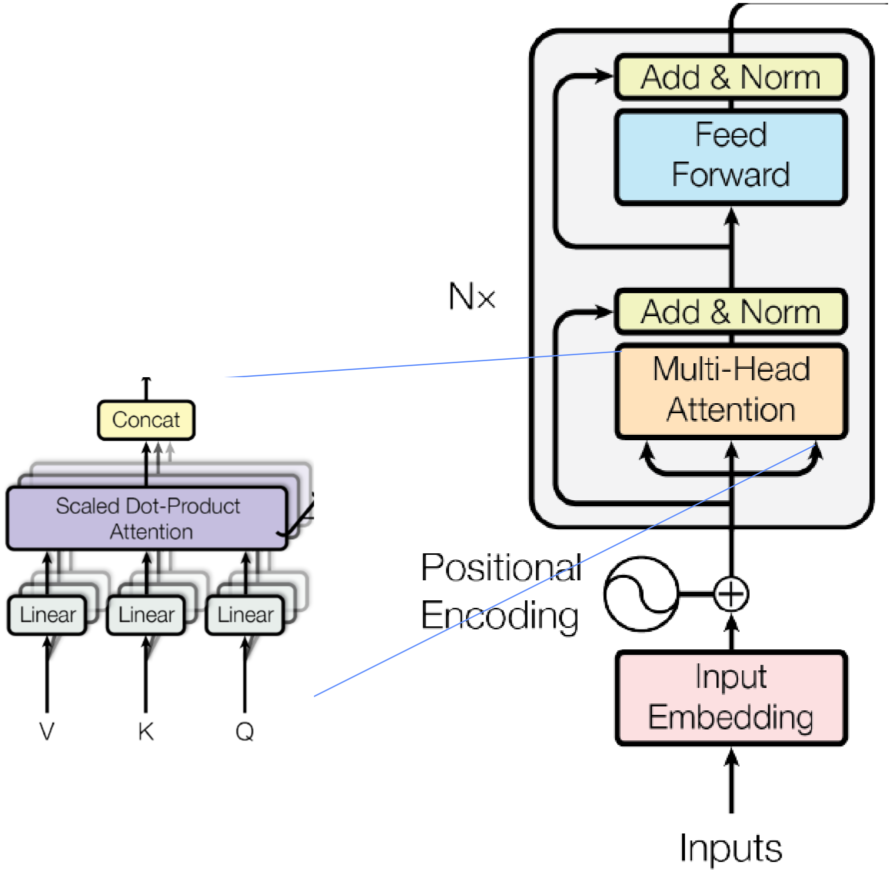


Finally, a transformer

Include skip-add connections

Include Layer Normalization or DropOut layers

Multi-Head – for N heads produce *Tx(E/N)* each



Finally, a transformer

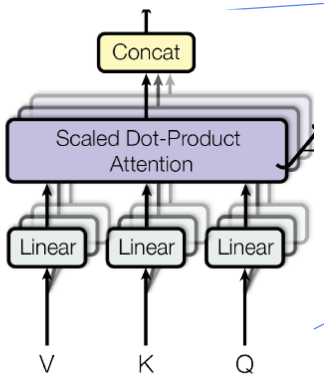
Include skip-add connections

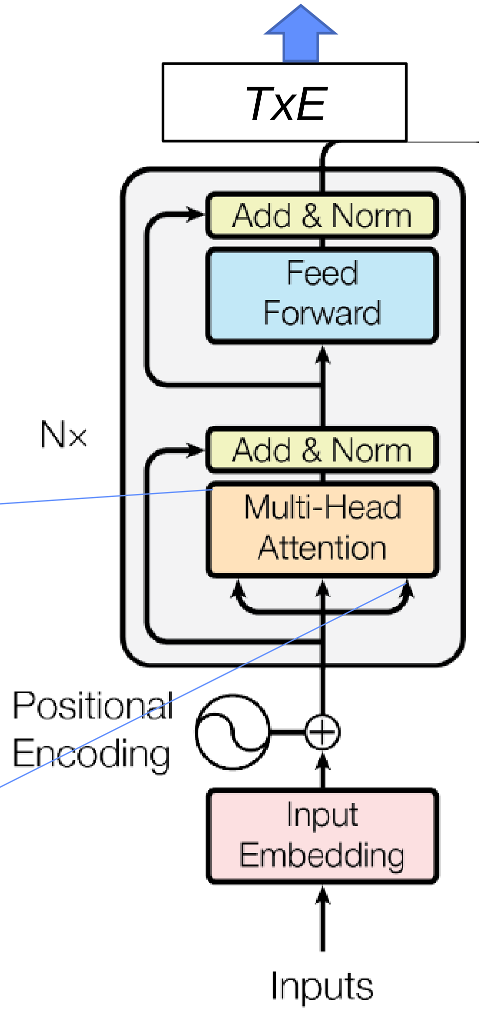
Include Layer Normalization or DropOut layers

Multi-Head – for N heads produce *Tx(E/N)* each

Add MLP layers on top and keep output *TxE*

Then stack transformers!





Finally, a transformer

Include skip-add connections

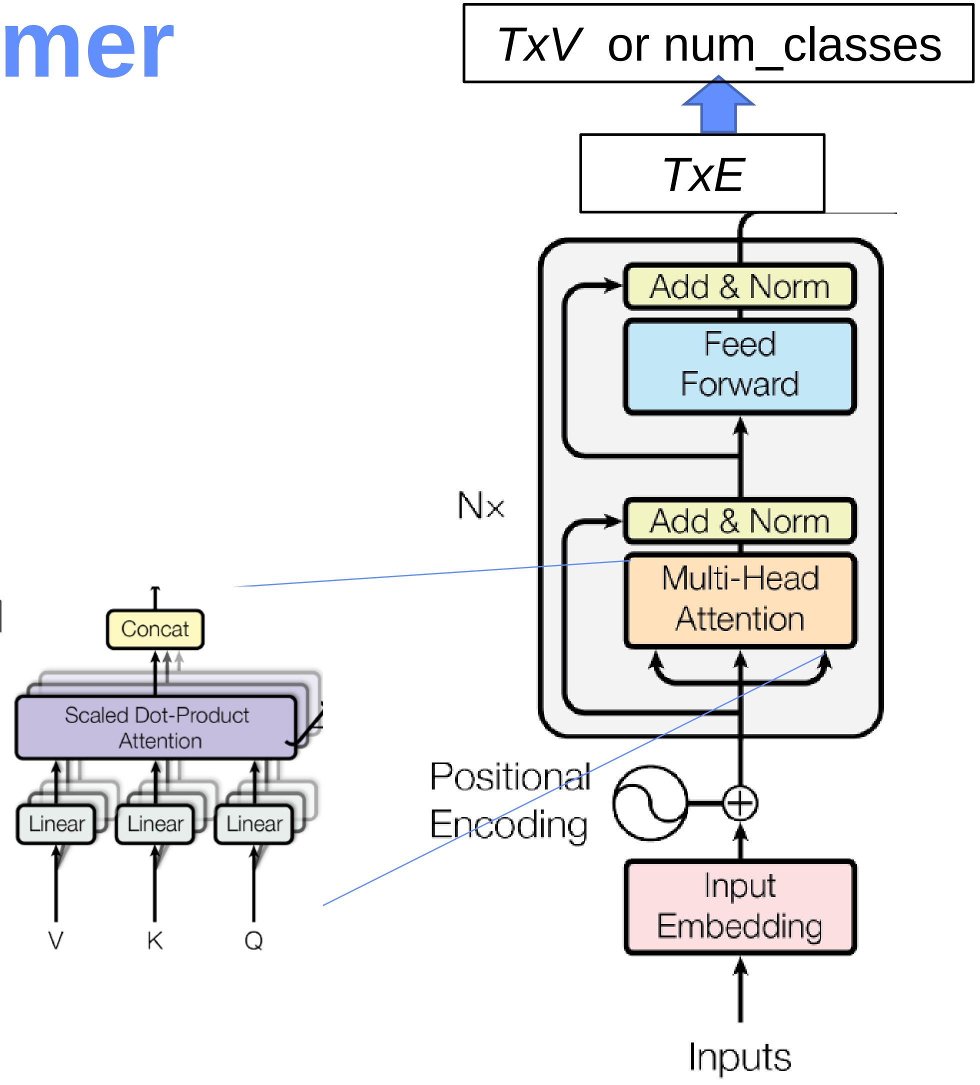
Include Layer Normalization or DropOut layers

Multi-Head – for N heads produce *Tx(E/N)* each

Add MLP layers on top and keep output *TxE*

Then stack transformers!

Finally, produce some classification or word predictions



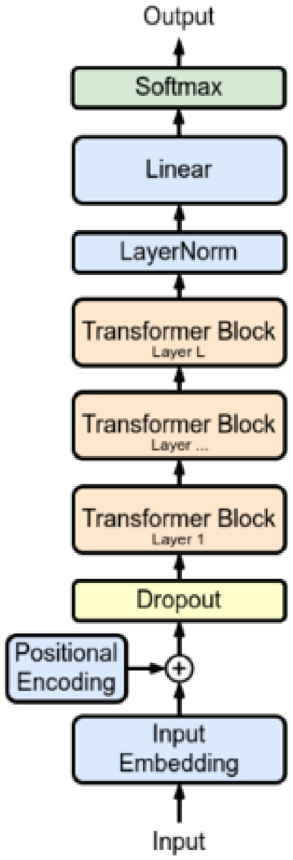
2 kinds of training strategies

GPT – predict next word only look back at prior context (which could be a whole document)

Put mask on attention weights so that predictions only depend on previous tokens

BERT – *No attention mask* so all token dependencies can influence all other tokens predictions

Special tokens help create a variety of tasks



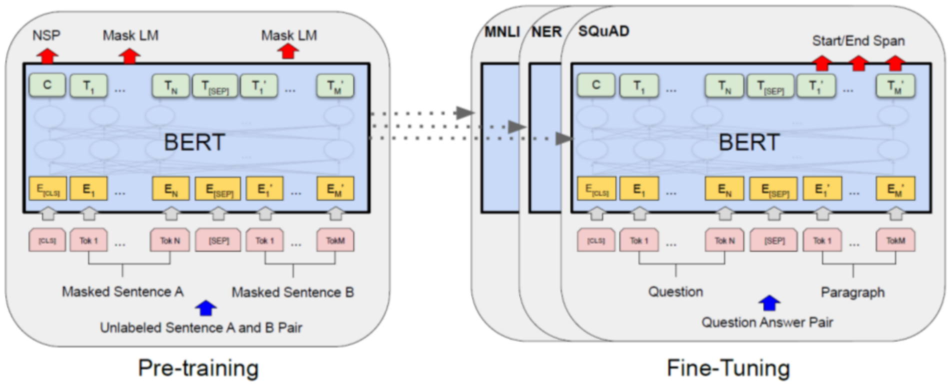
BERT (Bidirectional Encoder Representations from Transformers)

Goal: Train a model to develop general token-level AND sentence-level encoding

1 Pretrain on:

- fill-in-the-blank

- binary classification if 2 sentences go together



Devlin, etal, 2019

2 Fine tune on variety of tasks

GPT (generative pre-trained transformer)

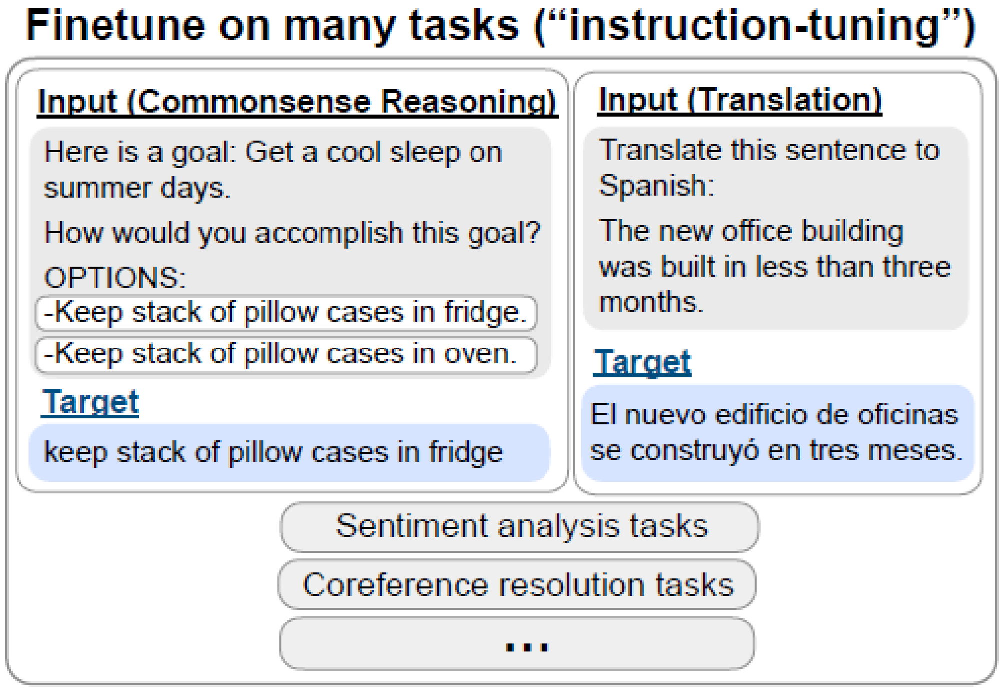
Goal: Train a transformer model at large scale so that it develops very general representations that are useful for many language tasks.

‘GPT3 shows strong performance on many NLP tasks so that with a few examples it nearly matches fine-tuned systems’

Lang. Models are Few Shot Learners
Brown, etal, 2020, openAI,

‘Instruction tuning’ improves models so they don’t need examples

Finetuned language models are zero-shot Learners. Wei et al, 2022, Google



- **Transformers for Science applications**

Can anything be cast as a kind of sentence, or an arrangement of tokens?

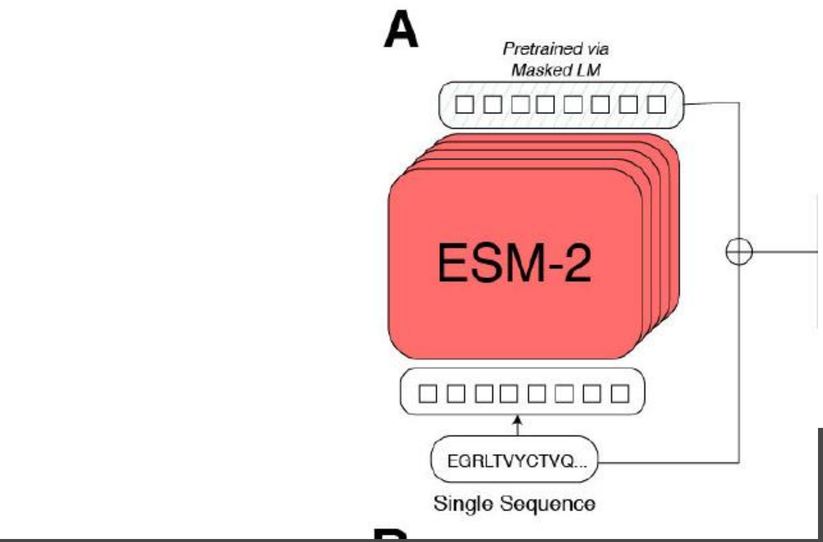
ESM Fold model

Language models of protein sequences at the scale of evolution enable accurate structure prediction

Lin etal, Meta Research 2022

Atom level structure prediction

Uses protein sequence as input to transformer layers (like LLM)



ESM Fold model

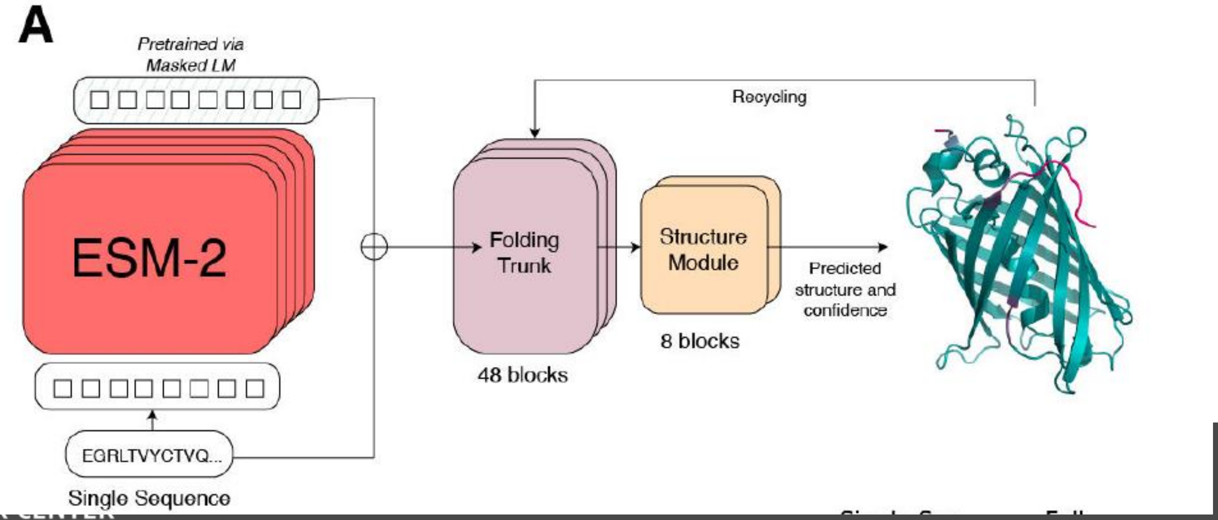
Language models of protein sequences at the scale of evolution enable accurate structure prediction

Lin etal, Meta Research 2022

Atom level structure prediction

Uses protein sequence as input to transformer layers (like LLM)

Predicts a map of protein contact which gets *iteratively refined* by a ‘folding block’ transformer and structure module (similar to AlphaFold2, but faster)

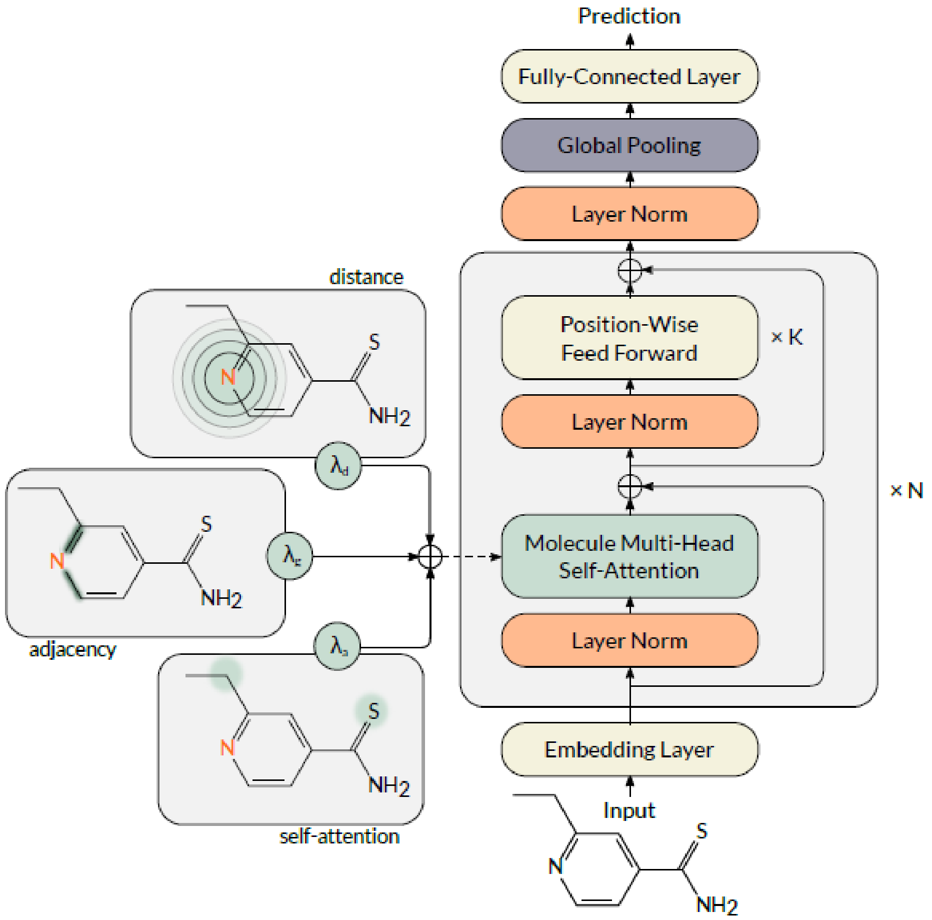


Molecule Attention Transformer
(Maziarka et al. 2020)

Molecular property prediction

Uses the set of atoms as input (like sentence tokens)

Includes spatial information by using a sum of the attention matrix, a distance matrix, and an adjacency matrix.



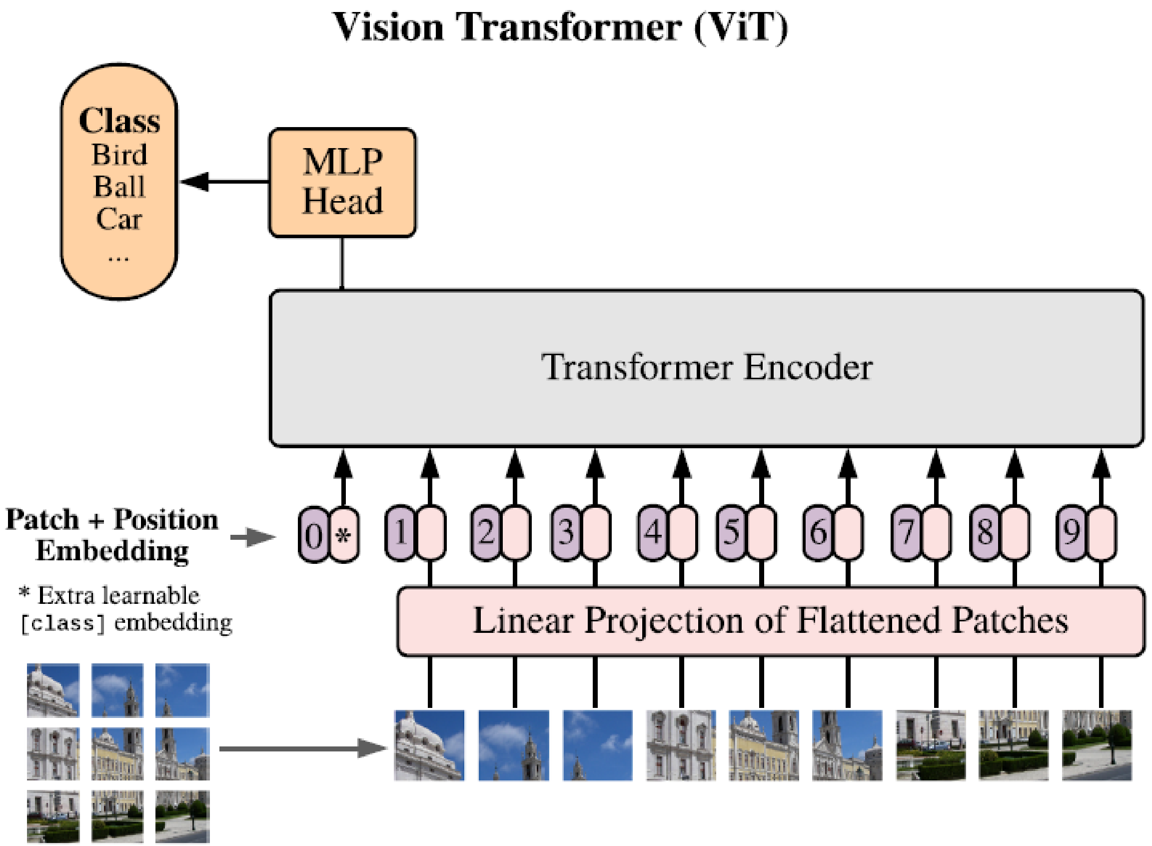
The Visual Transformer (ViT)

An image is worth 16x16 words:
Transformers for image recognition at scale
Adosovitski, et al, 2021, Google Research

Uses a sequences of image patches (16x16) like a sentence of tokens (ie 224x224 pixels is 16x16 patches of 14x14 pixels)

Uses a classification token like Bert to learn image output classes

Competitive or better than CNNs but might need more data

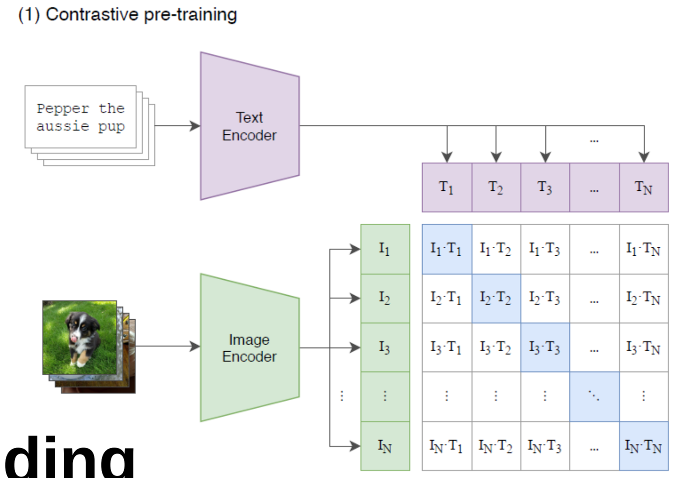


- **Combining images and text often makes DL work better, or more generic, for image or text tasks**

CLIP – Contrastive Language-Image Pretraining

Learning Transferable Visual Models From Natural Language Supervision

Radford et al, 2021, Open AI



Uses 400M images and captions for training

Learns a combination of text and image embeddings into a new **multi-modal embedding**

CLIP – Contrastive Language-Image Pretraining

Learning Transferable Visual Models From Natural Language Supervision

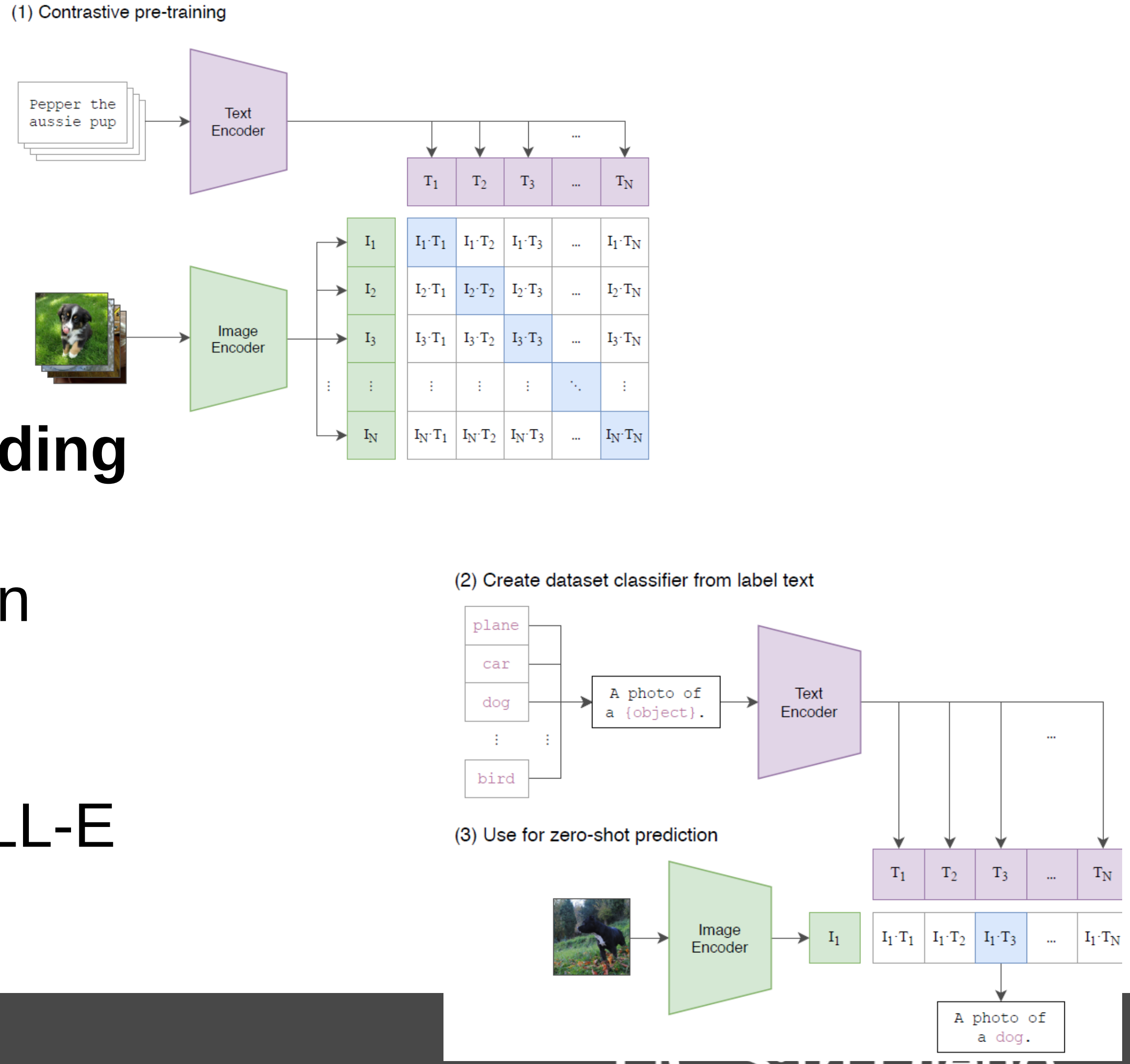
Radford et al, 2021, Open AI

Uses 400M images and captions for training

Learns a combination of text and image embeddings into a new **multi-modal embedding**

Performs classification by prompting it with an image and possible captions

Note: CLIP with diffusions gets close to DALL-E



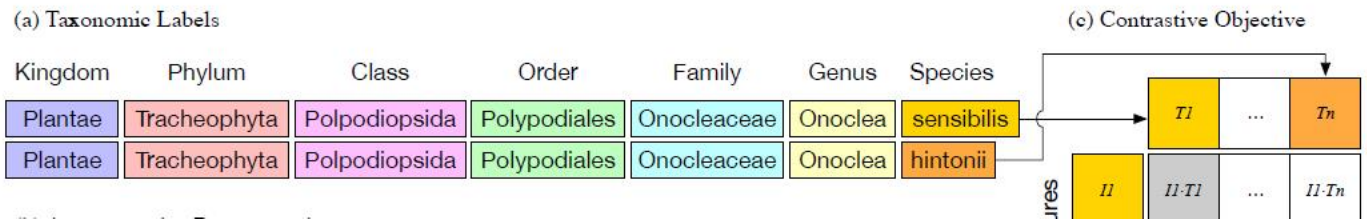
BIOCLIP: A Vision Foundation Model for the Tree of Life

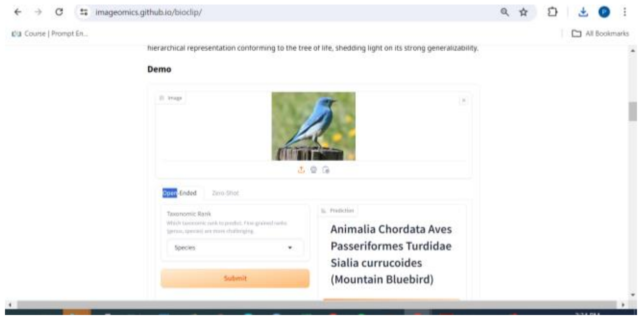
Stevens, etal 2024 OSU

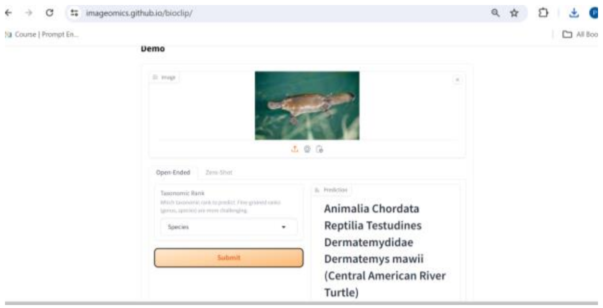
Uses pre-trained CLIP for a base

Uses Tree-of-Life 10M dataset of biology images with taxonomic labels

The taxonomic hierarchy is presented as a sequence of words for different species







end