

Teaching Language Models to Speak Biology

Aaron Feller
aaron.feller@utexas.edu

AI • SCIENCE

Tech giants see a cure for cancer in AI. But Eli Lilly's CEO finds it 'not particularly good' at solving biology or chemistry problems

By **Jake Angelo**
News Fellow

March 2, 2026, 1:37 PM ET

Add us on



BIOTECH AND PHARMA

Eli Lilly reaches $2.75 billion deal with Insilico to bring AI-developed drugs to the global market

PUBLISHED SUN, MAR 29 2026•9:36 AM EDT | UPDATED MON, MAR 30 2026•1:35 AM EDT

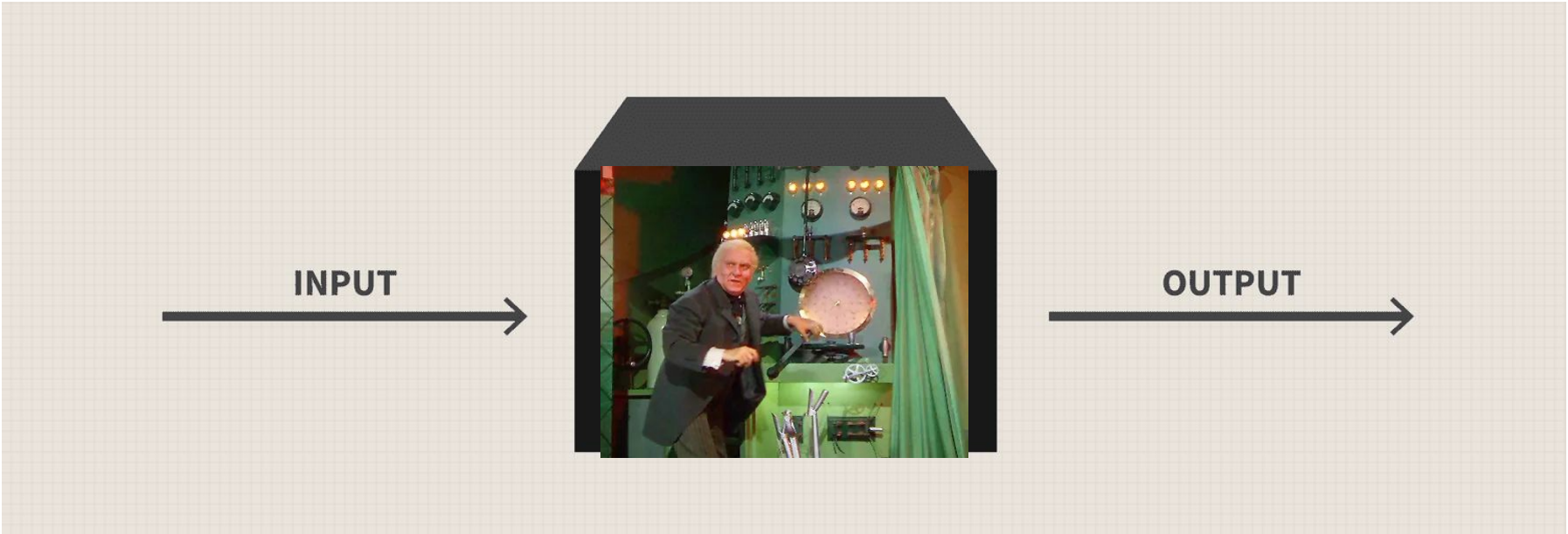
Evelyn Cheng
@IN/EVELYN-CHENG-53B23624
@CHENGEVELYN

SHARE

Roadmap

- **Inside the black box: Neural Networks and Software 2.0**
- A two-neuron toy example
- Transformers, attention, and how we get language in and out
- Applications, examples, and Google Colab

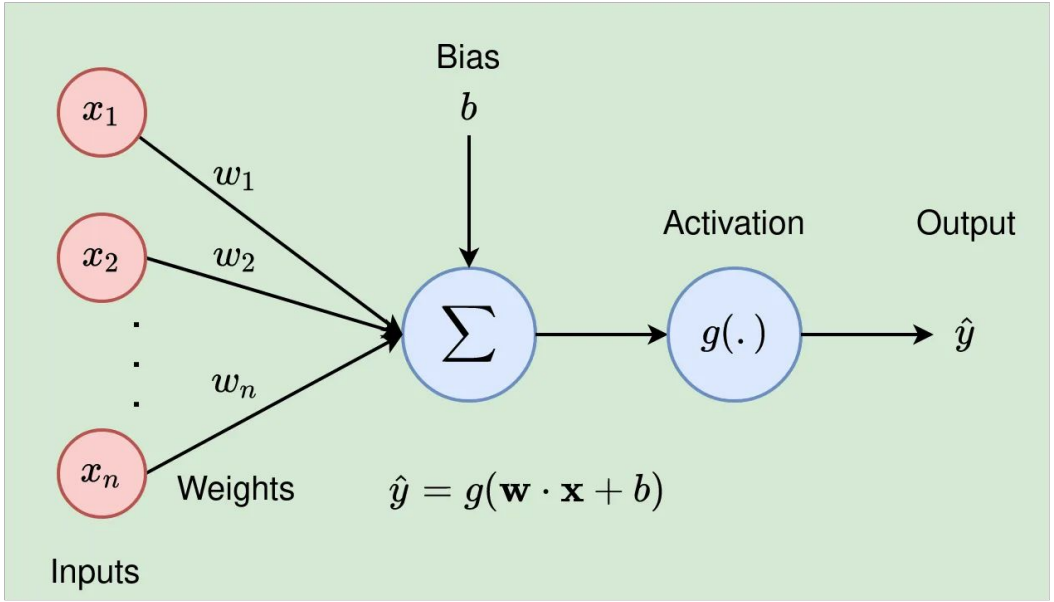
AI/ML is often depicted as a Wizard of Oz scenario



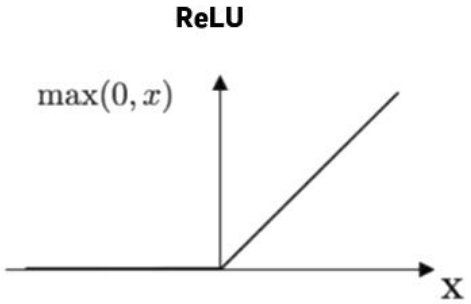
Classical methods derive the output from programmed logic
Machine learning **derives logic** from example data



The neuron is the smallest unit of machine learning

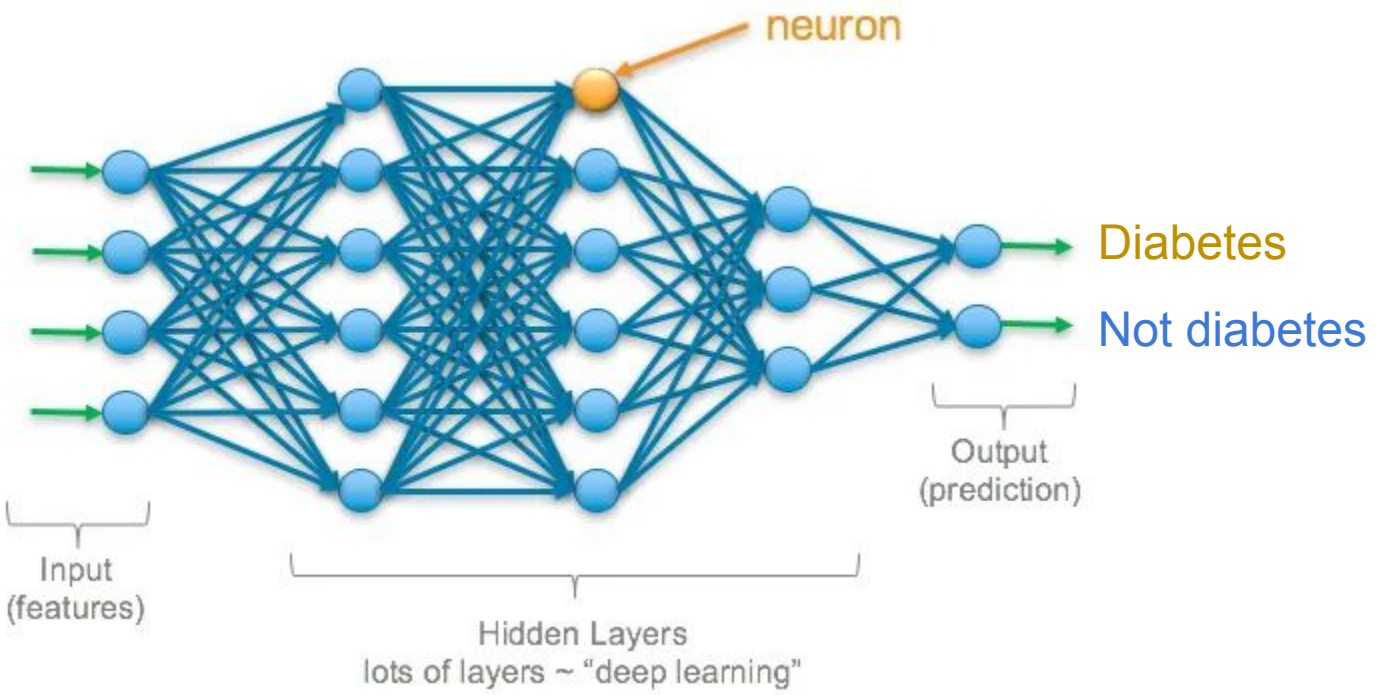


Activation function:

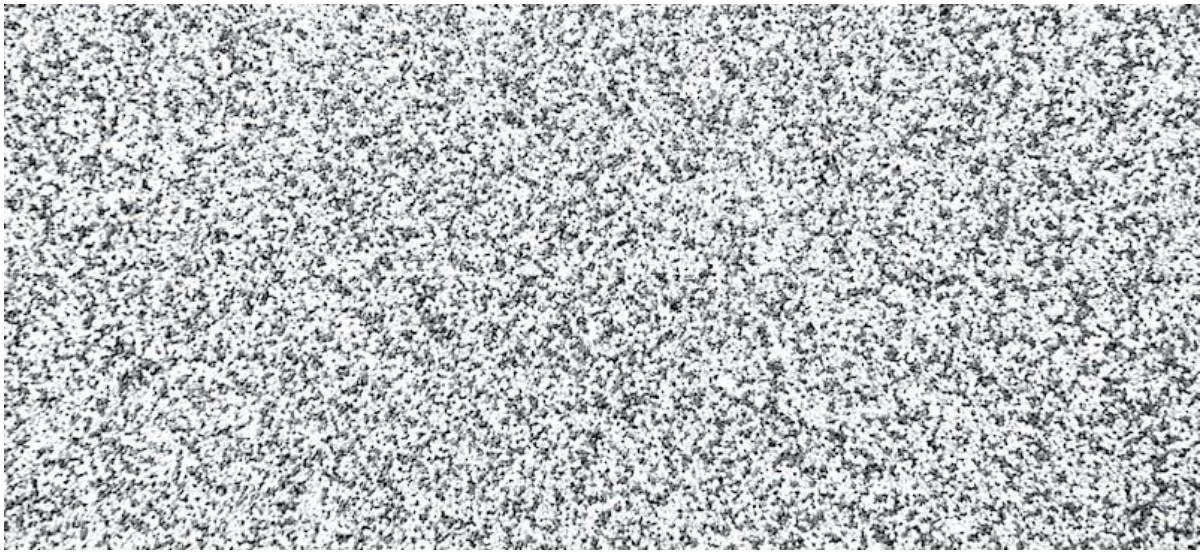


A machine learning network is made up of neurons

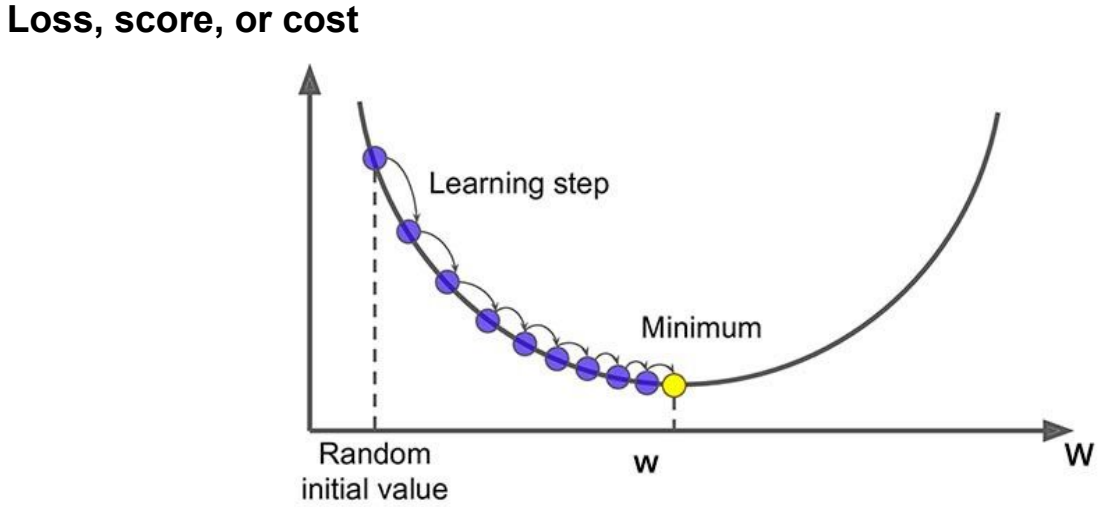
SERUM
NA
K
CL
CO2
BUN
CREAT
eGFR
GLUCOSE
CA, TOT
PROTEIN
ALBUMIN
PO4



All of the model ***parameters*** (weights and biases) start randomly initialized as values between 0 and 1



We then adapt the parameters slowly, step-by-step, until they *converge* toward a general solution

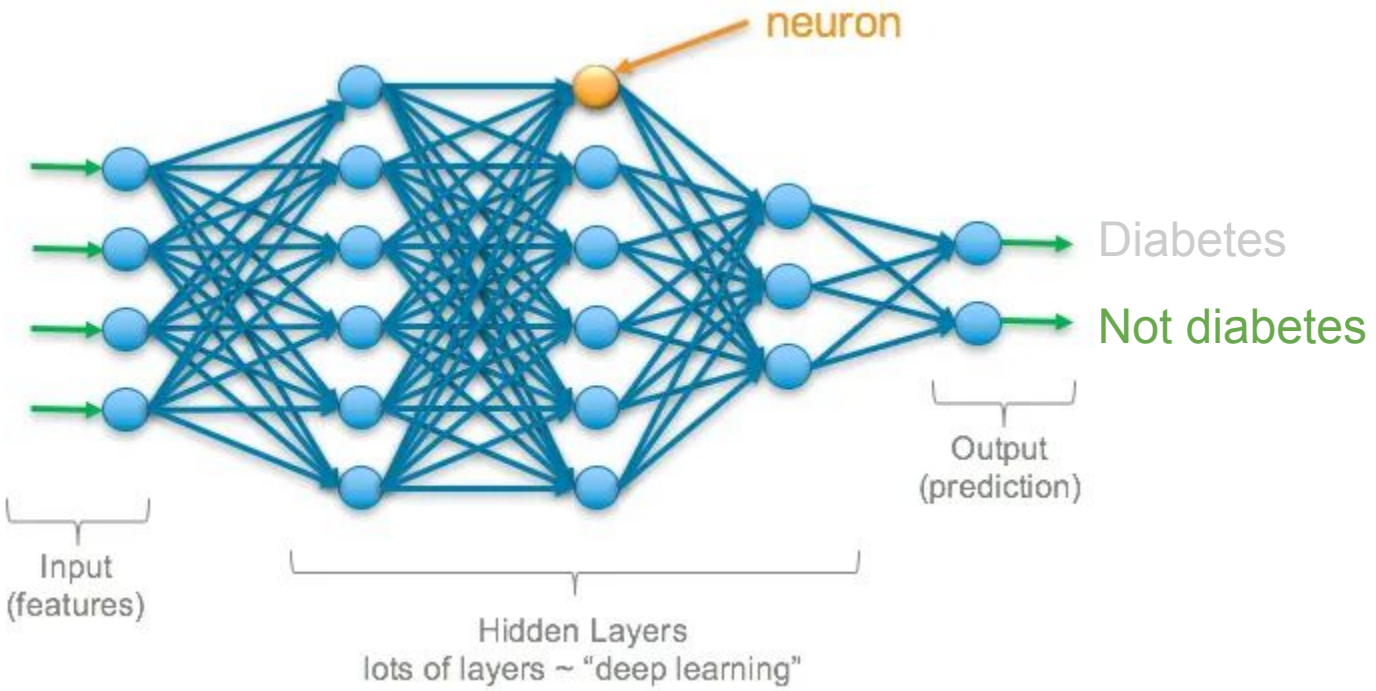


We then adapt the parameters slowly, step-by-step, until they *converge* toward a general solution



We then apply that solution on new data to make predictions for unmeasured outcomes

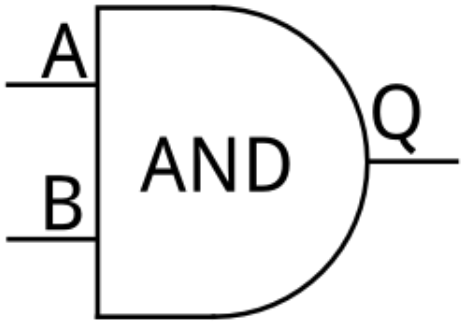




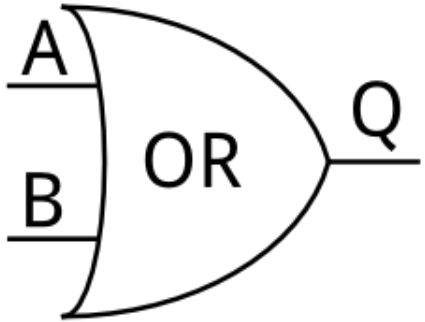
Roadmap

- Inside the black box: Neural Networks and Software 2.0
- **A two-neuron toy example**
- Transformers, attention, and how we get language in and out
- Applications, examples, and Google Colab

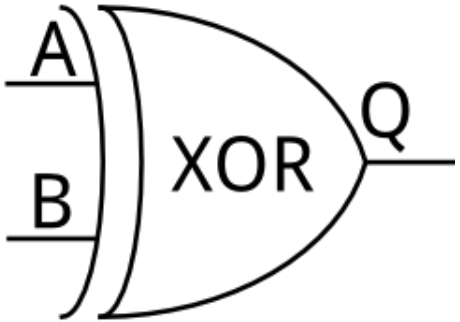
The **XOR logic gate** provides a simple non-linear solution



If A **and** B is true:
Q is true



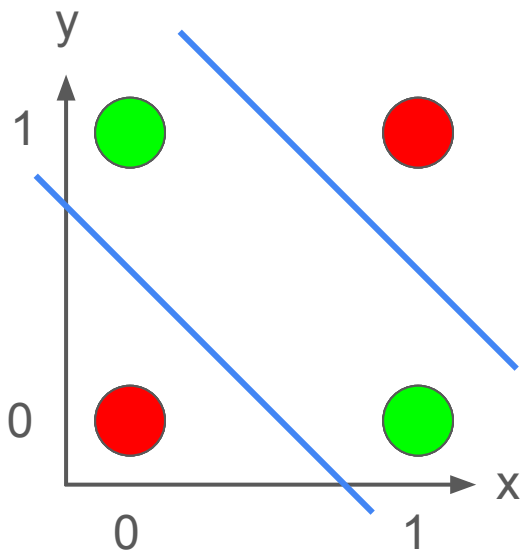
If A **or** B is true:
Q is true



If A **or** B is **eXclusively** true:
Q is true

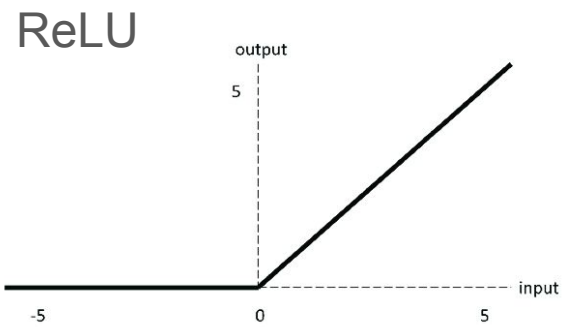
Why use neurons and ML?

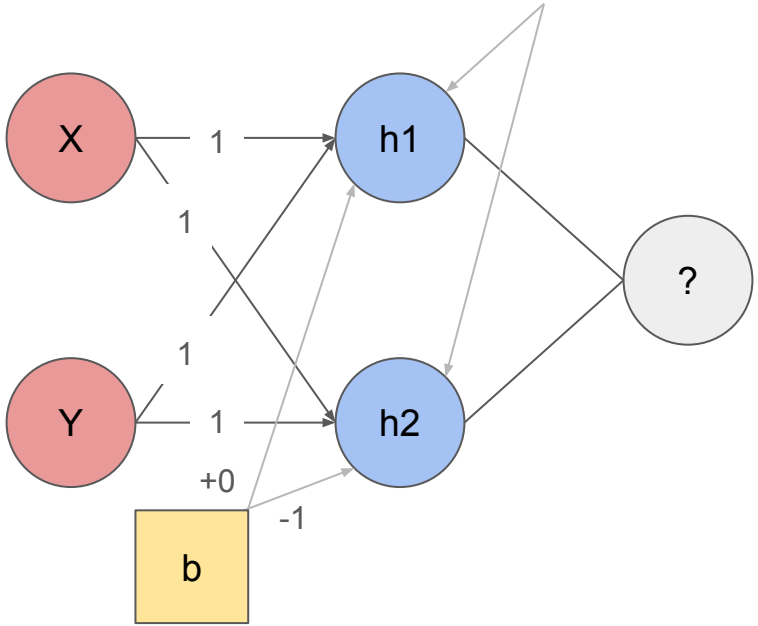
X	Y
0	0
0	1
1	0
1	1



Neural networks can restructure data

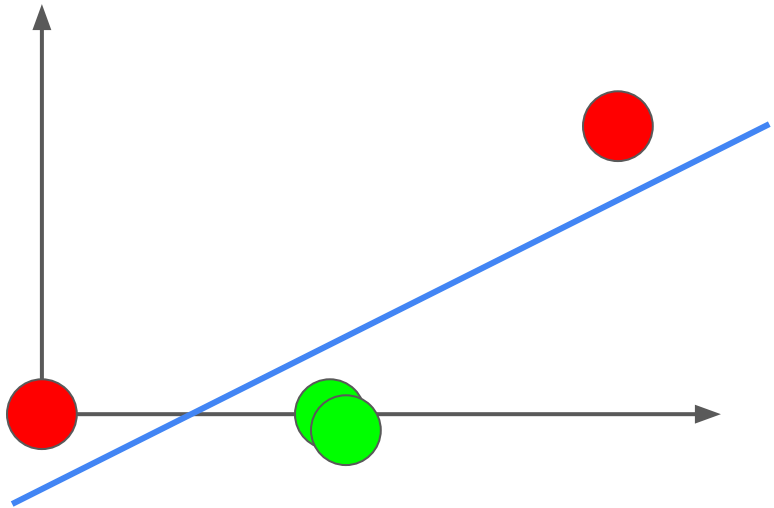
X	Y	h1	h2	XOR (X⊕Y)
0	0			0
0	1			1
1	0			1
1	1			0



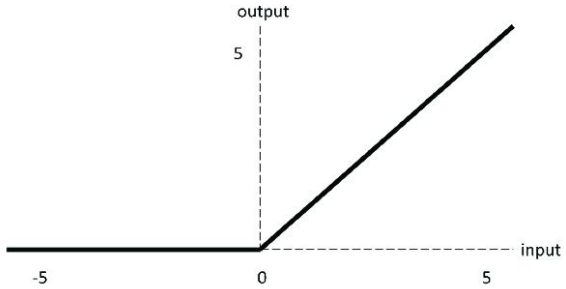


We can now linearly separate for the XOR function

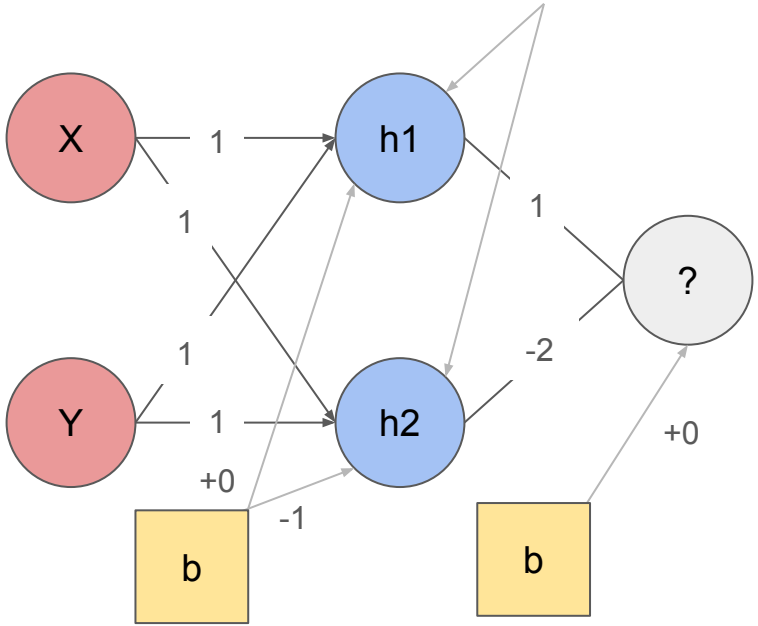
X	Y	h1	h2	XOR (X⊕Y)
0	0	0	(-1) 0	0
0	1	1	0	1
1	0	1	0	1
1	1	2	1	0



Final steps for computing the result



X	Y	h1	h2	Result	XOR (X⊕Y)
0	0	0	(-1) 0		0
0	1	1	0		1
1	0	1	0		1
1	1	2	1		0



Roadmap

- Inside the black box: Neural Networks and Software 2.0
- A two-neuron toy example
- **Transformers, attention, and how we get language in and out**
- Applications, examples, and Google Colab

Enter the Transformer



Transformers work by a mechanism called **attention**

Attention is **all you need**

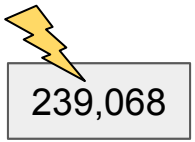
A Vaswani, N Shazeer, N Parmar… - Advances in neural …, 2017 - proceedings.neurips.cc

… to attend to **all** positions in the decoder up to and including that position. **We need** to prevent

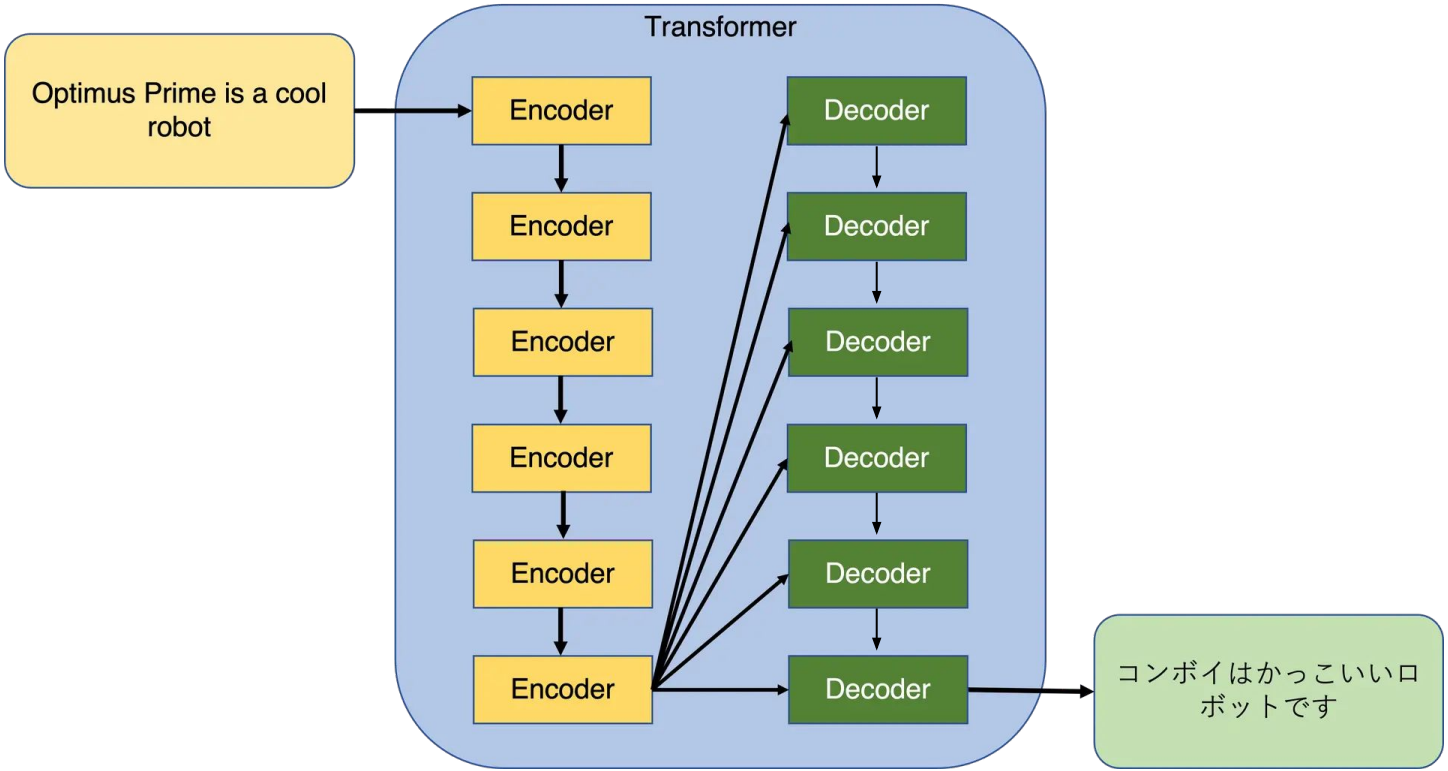
… **We** implement this inside of scaled dot-product **attention** by masking out (setting to $-\infty$) …

These are some top cited papers (from Google Scholar)

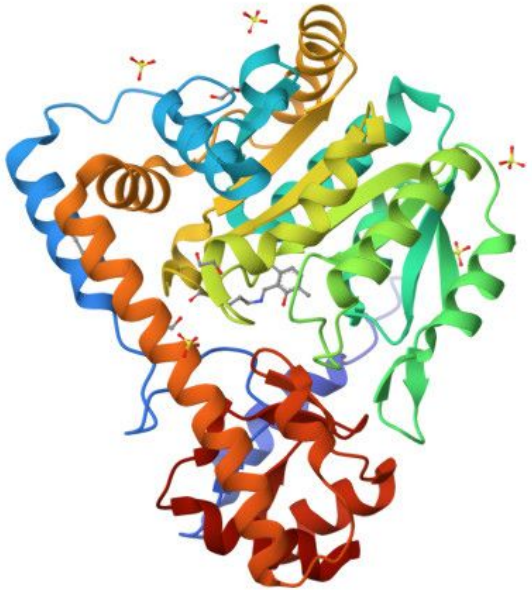
237,152	Protein measurement with the folin phenol reagent.	Biology lab technique	1951	*J. Biol. Chem.*
310,877	Cleavage of structural proteins during the assembly of the head of bacteriophage T4.	Biology lab technique	1970	*Nature*
258,913	A rapid and sensitive method for the quantitation of microgram quantities of protein utilizing the principle of protein-dye binding.	Biology lab technique	1976	*Anal. Biochem.*
91,073	DNA sequencing with chain-terminating inhibitors.	Biology lab technique	1977	*Proc. Natl Acad. Sci. USA*



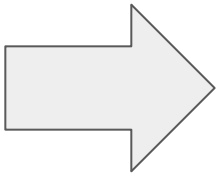
The transformer was initially created to translate language



Why should we care?



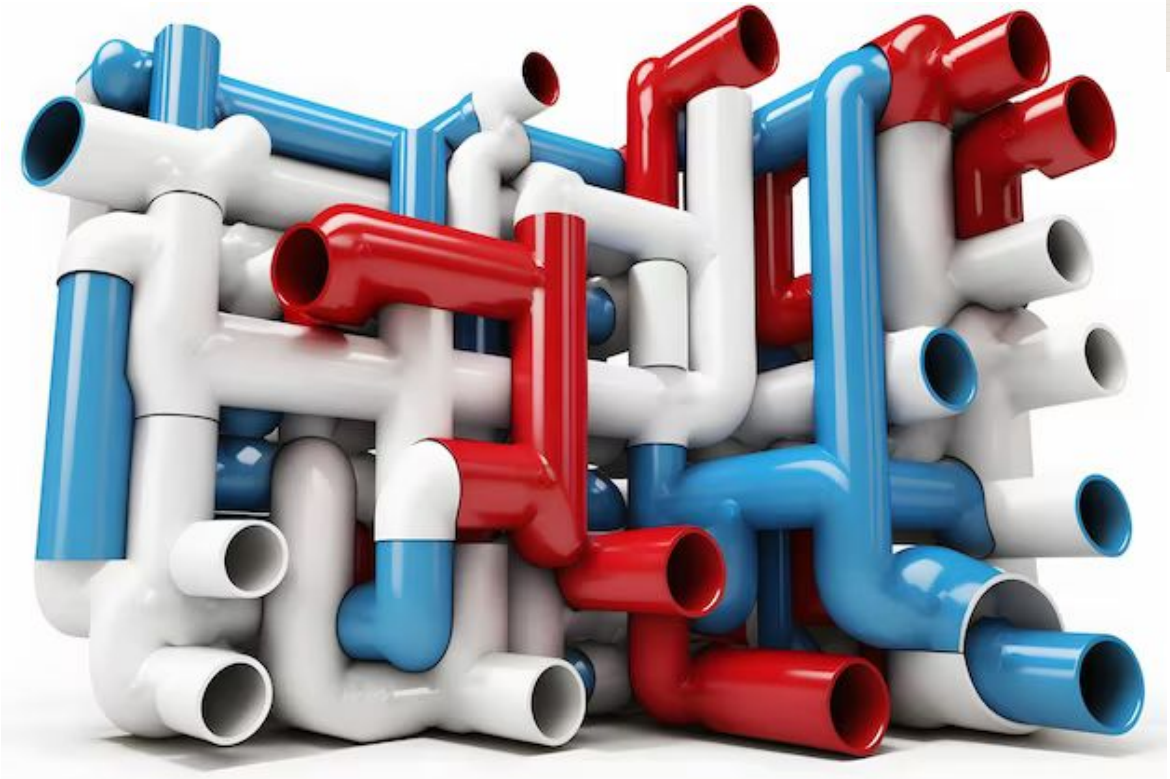
High dimensional
physical object

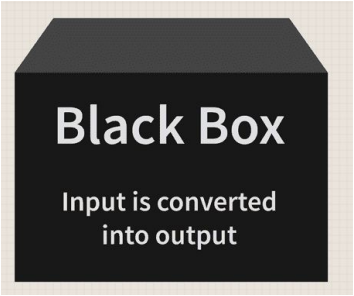




Low(er) dimensional
numerical object

How does the transformer work?



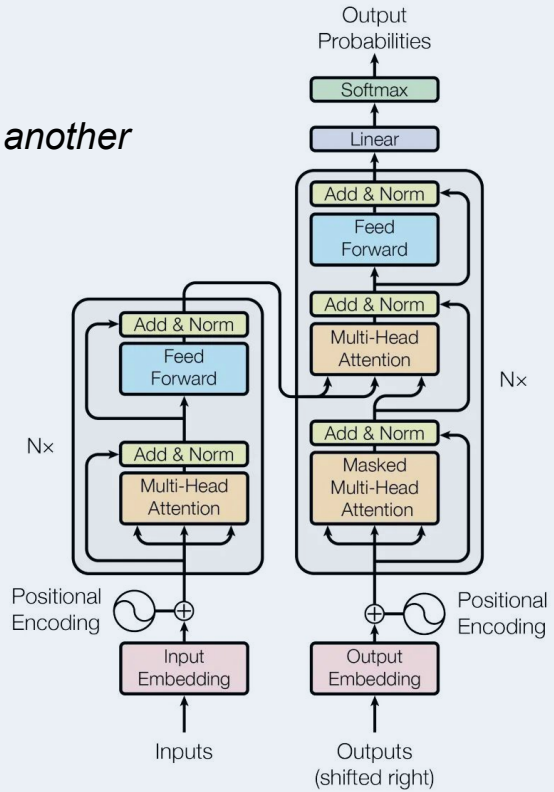


A deep dive into ML modeling with Transformers

Transformer

Transforms one data type to another

Initially created by Google for text translation from one language to another

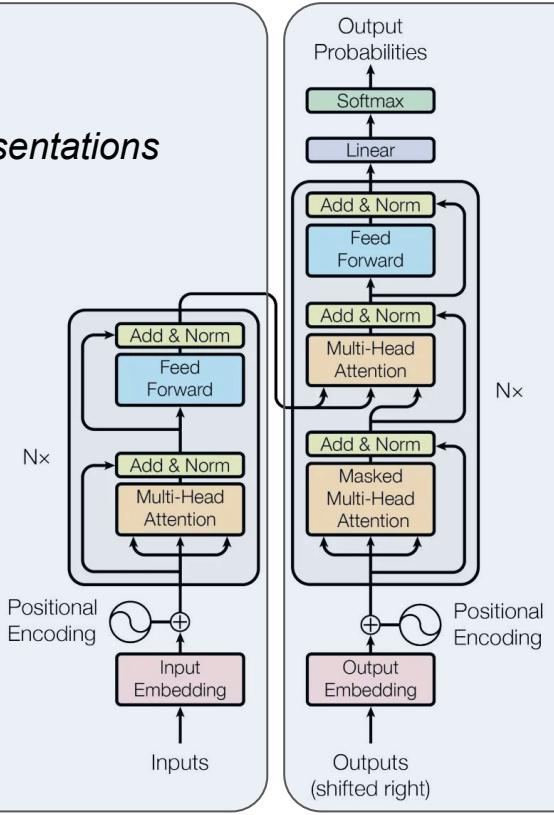


Transformer can be seen as two parts, BERT and GPT

BERT

Bidirectional Encoder Representations from Transformers

Uses transfer learning to continue learning from the previously seen data while adding user-specific tasks and layers



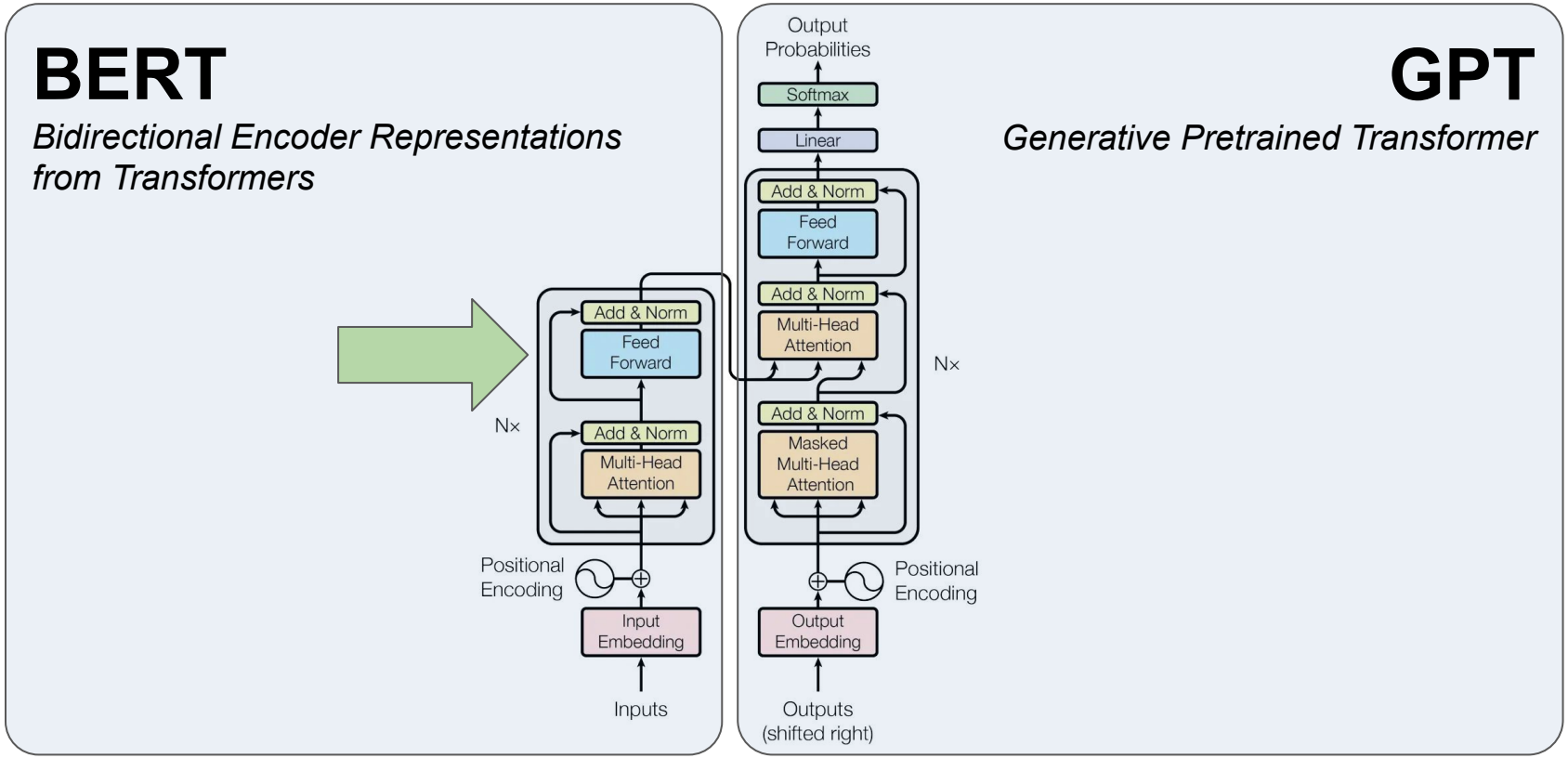
GPT

Generative Pretrained Transformer

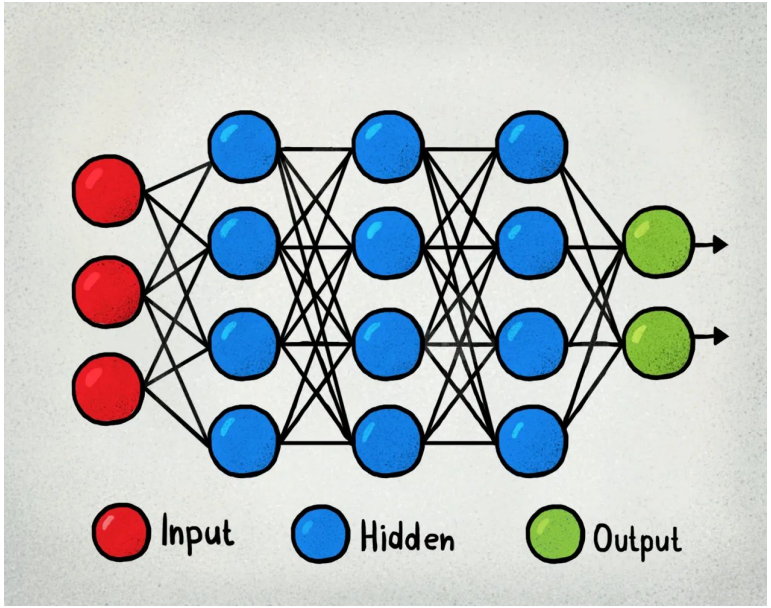
Decodes from pre-learned embeddings to present outputs that match user prompts

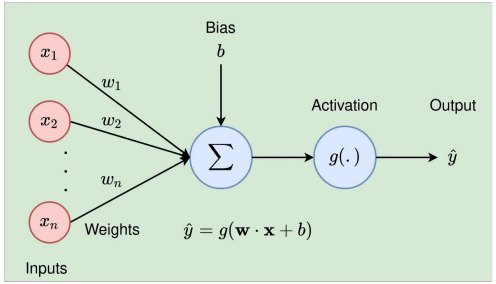
OpenAI

Transformer can be seen as two parts, BERT and GPT

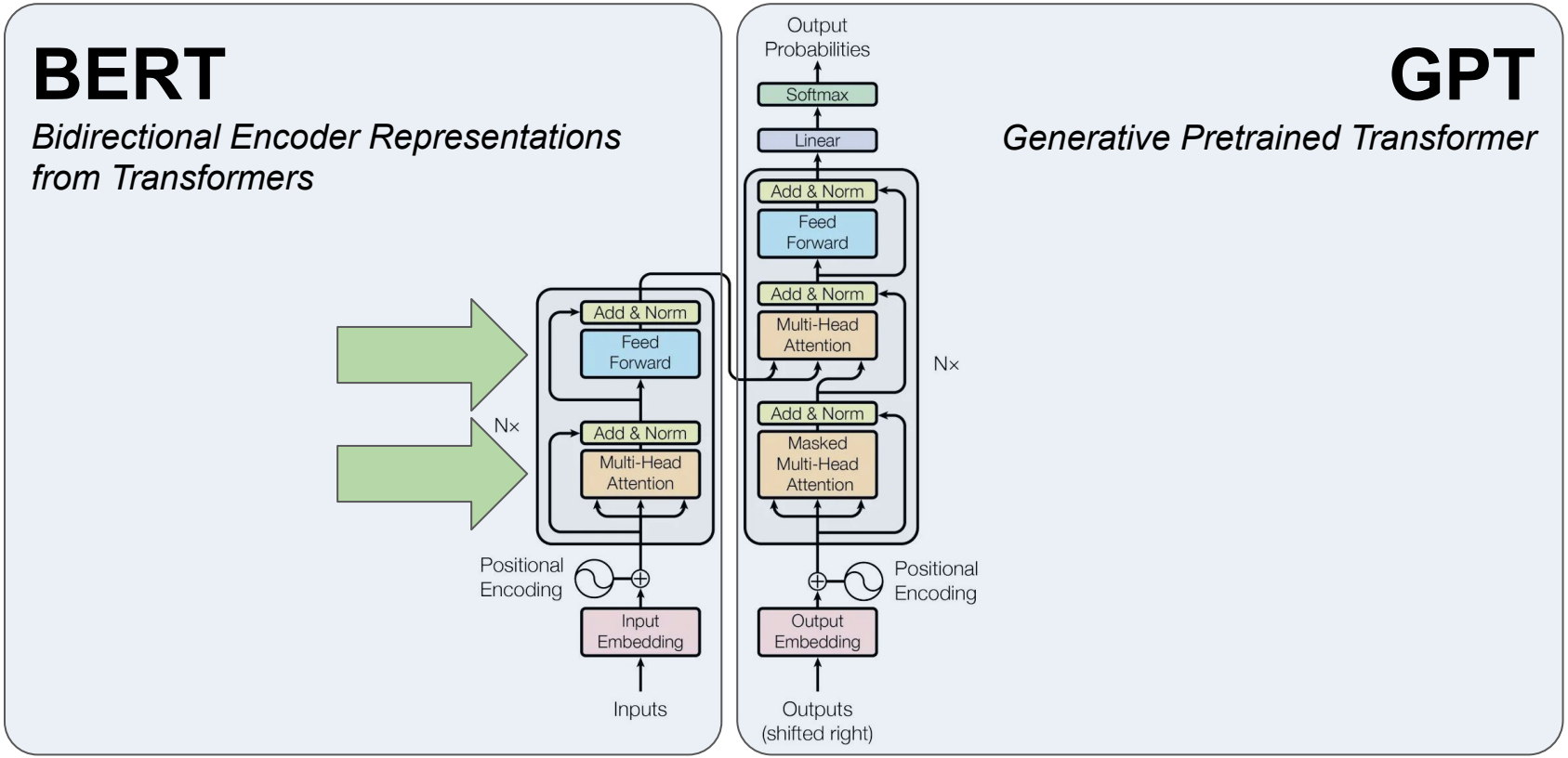


FCFF (Fully Connected Feed Forward) networks made up of *layers* of neurons, where all neurons are connected to those in the *prior* layer

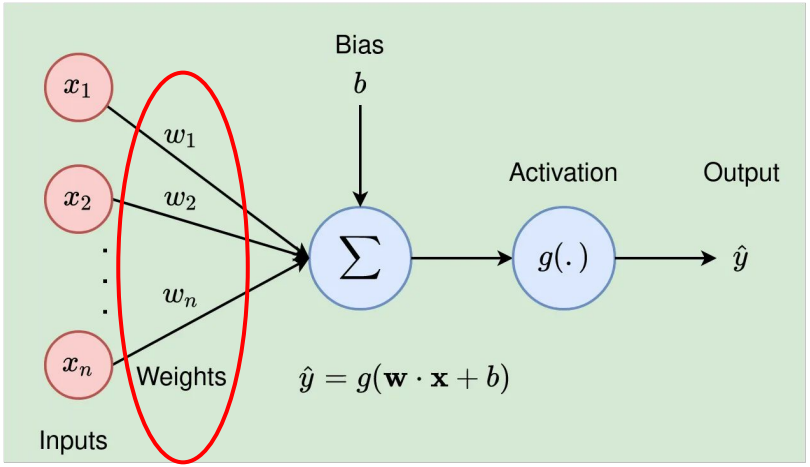


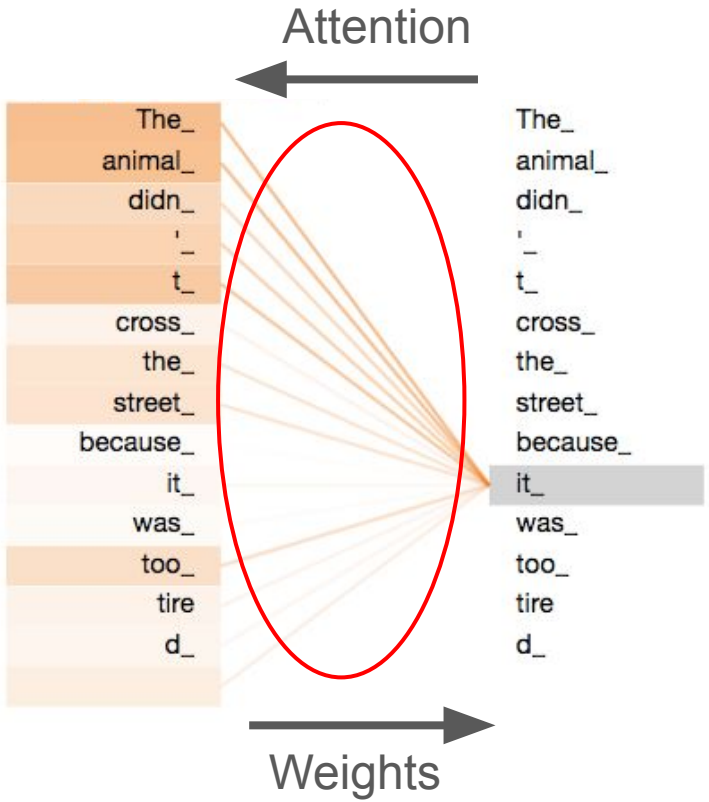


Transformer can be seen as two parts, BERT and GPT

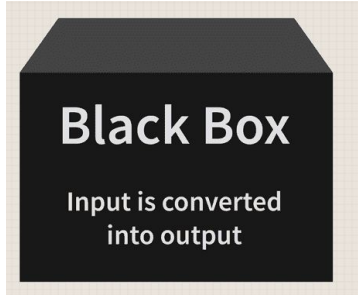


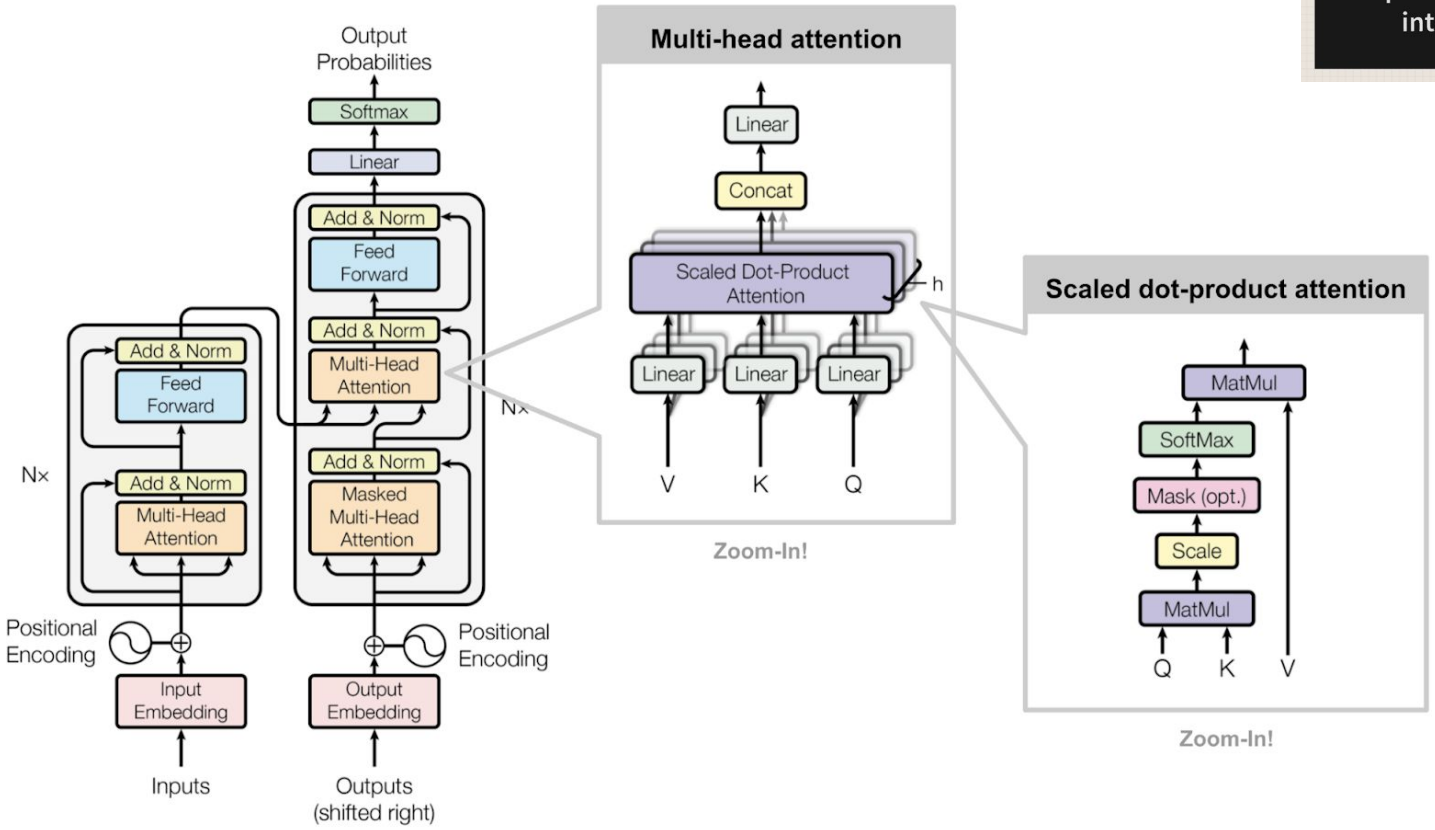
Attention allows the model to dynamically choose the weights for the inputs to each neuron



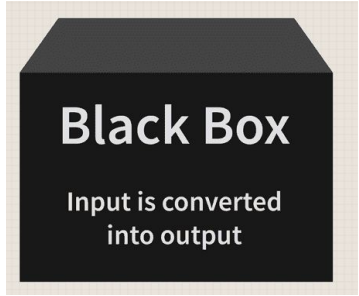


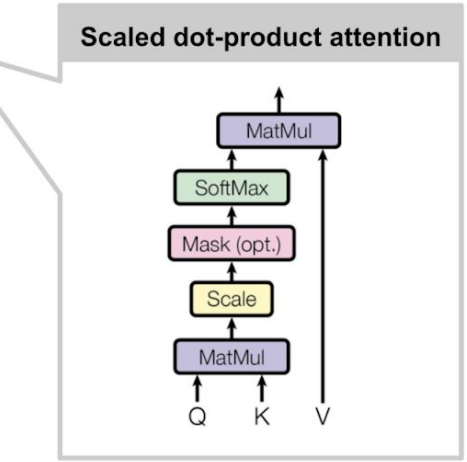
How does this work?





Queries, keys, and values are the heart of attention

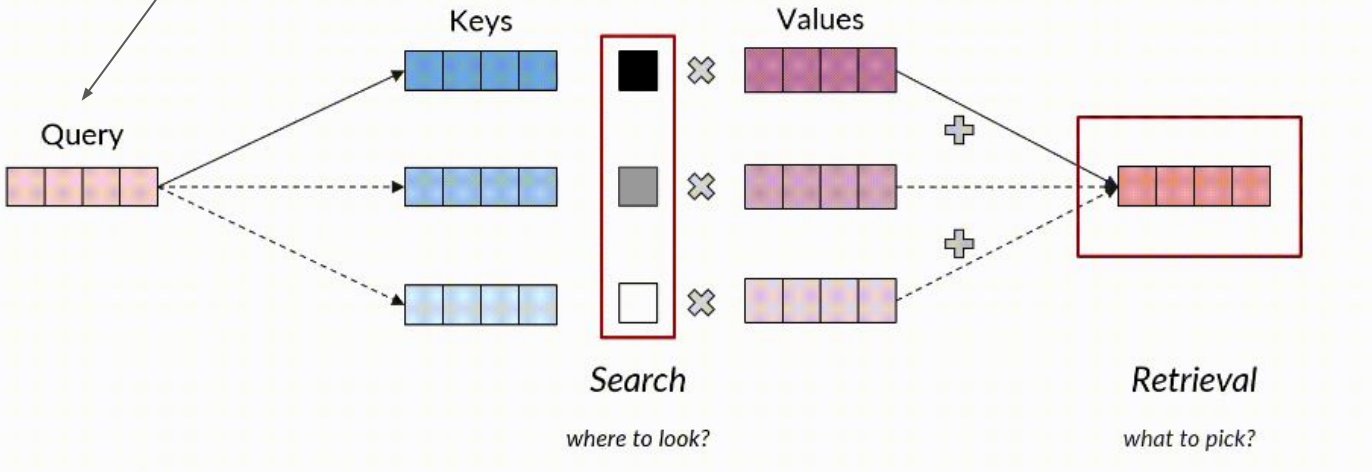




Prior layer output

Input

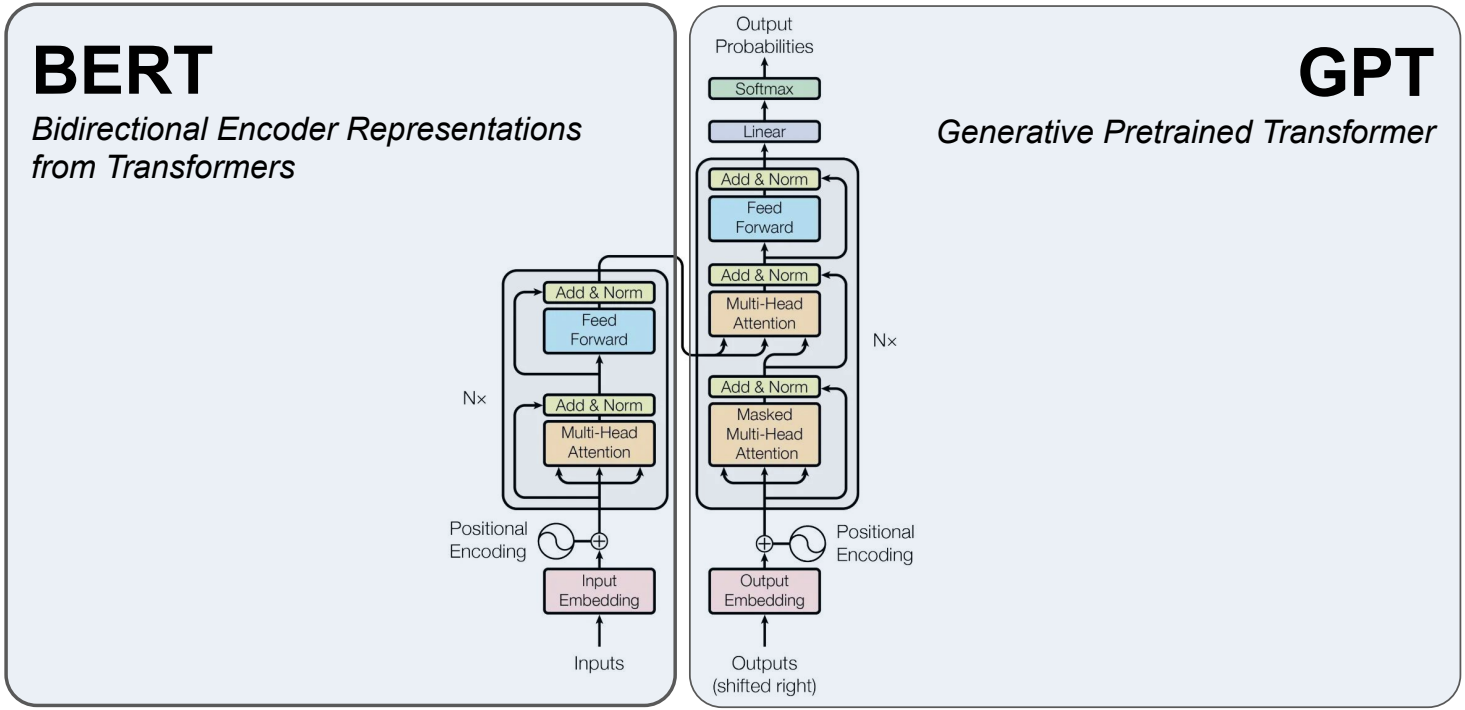
Transformation is learned from random initialization



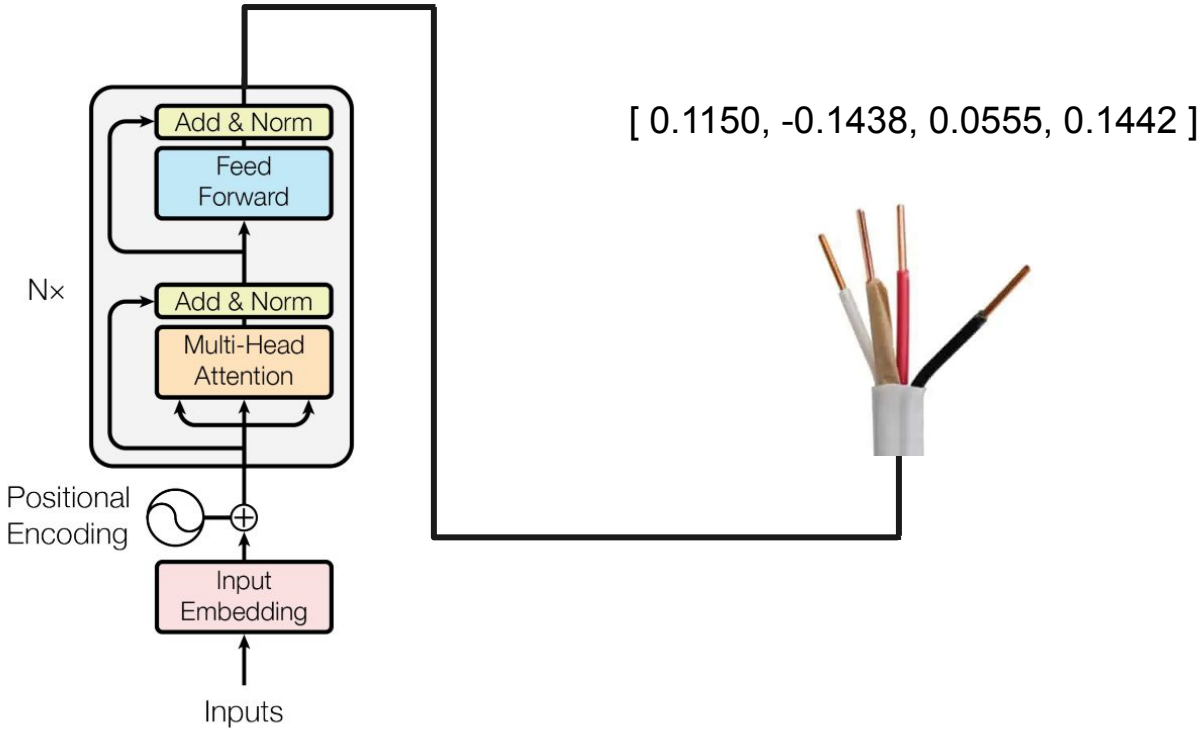
Single head of attention

	a	fluffy	blue	creature	roamed	the	verdant	forest
	$\vec{E}_1 \xrightarrow{W_Q} \vec{Q}_1$	$\vec{E}_2 \xrightarrow{W_Q} \vec{Q}_2$	$\vec{E}_3 \xrightarrow{W_Q} \vec{Q}_3$	$\vec{E}_4 \xrightarrow{W_Q} \vec{Q}_4$	$\vec{E}_5 \xrightarrow{W_Q} \vec{Q}_5$	$\vec{E}_6 \xrightarrow{W_Q} \vec{Q}_6$	$\vec{E}_7 \xrightarrow{W_Q} \vec{Q}_7$	$\vec{E}_8 \xrightarrow{W_Q} \vec{Q}_8$
a $\vec{E}_1 \xrightarrow{W_K} \vec{K}_1$	+0.7	−83.7	−24.7	−27.8	−5.2	−89.3	−45.2	−36.1
fluffy $\vec{E}_2 \xrightarrow{W_K} \vec{K}_2$	−73.4	+2.9	−5.4	+93.0	−48.2	−87.3	−49.7	+7.8
blue $\vec{E}_3 \xrightarrow{W_K} \vec{K}_3$	−53.4	−5.7	+1.8	+93.4	−55.6	−56.0	−26.1	−62.1
creature $\vec{E}_4 \xrightarrow{W_K} \vec{K}_4$	−21.5	−29.7	−56.1	+4.9	−32.4	−92.3	−9.5	−28.1
roamed $\vec{E}_5 \xrightarrow{W_K} \vec{K}_5$	−20.1	−40.9	−87.8	−55.4	+0.6	−64.7	−96.7	−18.9
the $\vec{E}_6 \xrightarrow{W_K} \vec{K}_6$	−87.9	−33.3	−22.6	−31.4	+5.5	+0.6	−4.6	−96.8
verdant $\vec{E}_7 \xrightarrow{W_K} \vec{K}_7$	−41.2	−55.5	−42.3	−59.8	−79.0	−97.9	+3.7	+93.8
forest $\vec{E}_8 \xrightarrow{W_K} \vec{K}_8$	−58.9	−75.5	−91.1	−90.6	−75.6	−89.0	−70.8	+4.7

While GPT is often used for generation, BERT is common for ***representation learning***



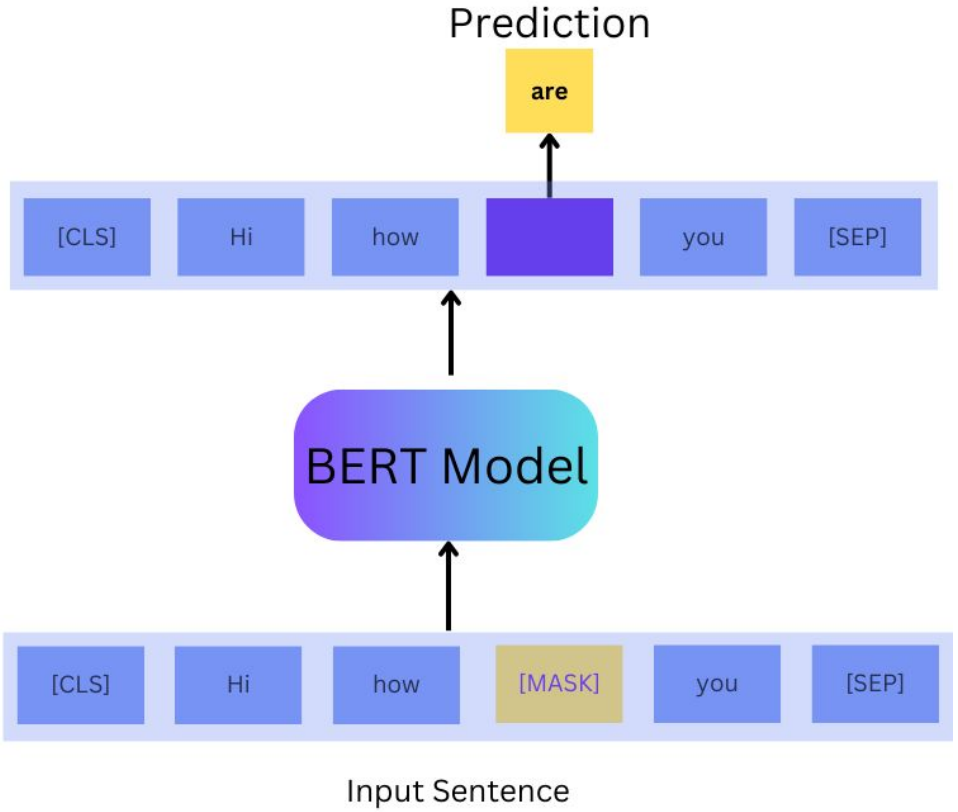
If we cut out the GPT (decoder), the output of the encoder (BERT) generates an embedding



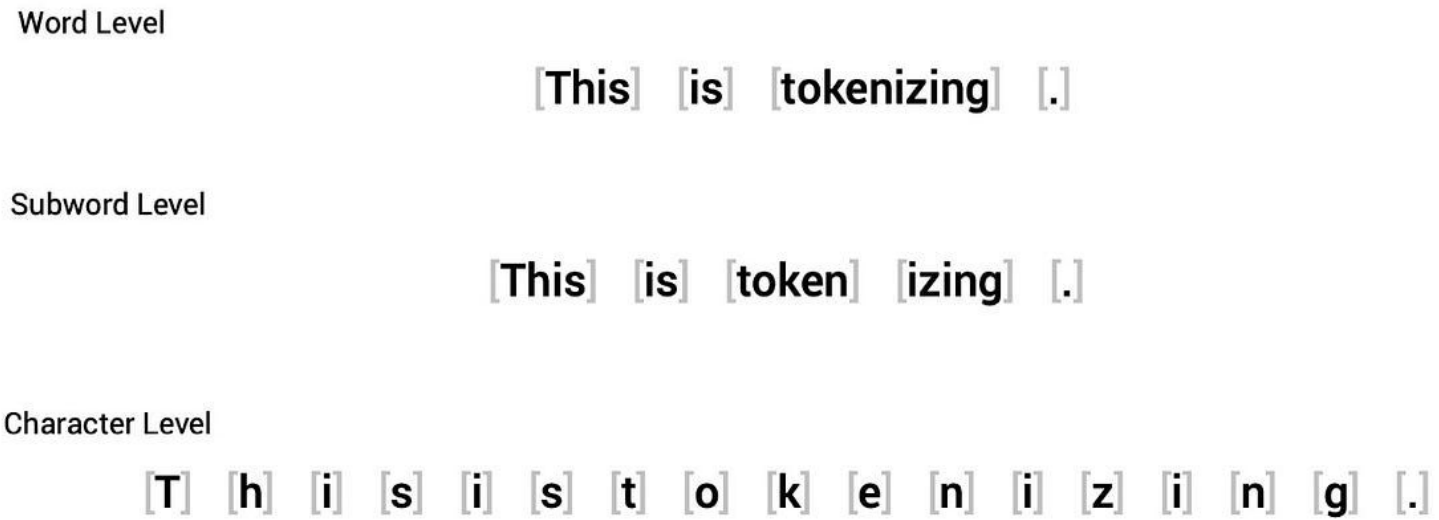
We train the model with an unlabeled corpus of data

Input sentence:

“Hi how are you?”



We get text into a model through tokenization



The token is converted into a vector of numbers

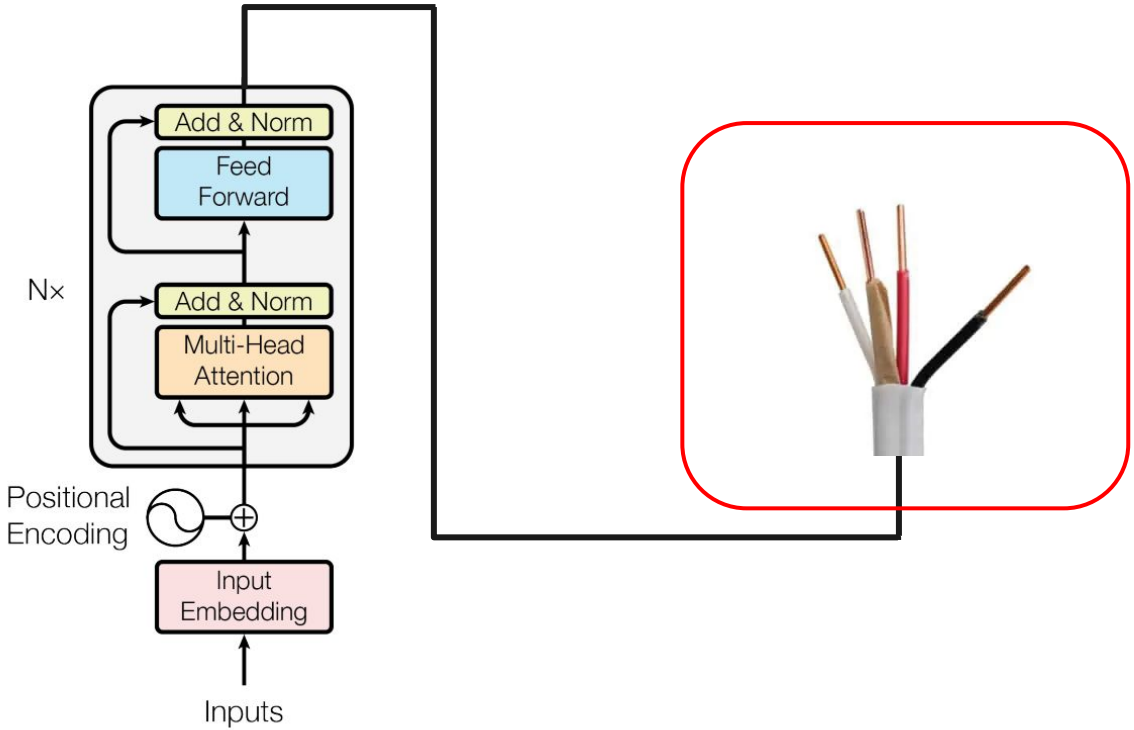
Programmed transformation

Learned transformation

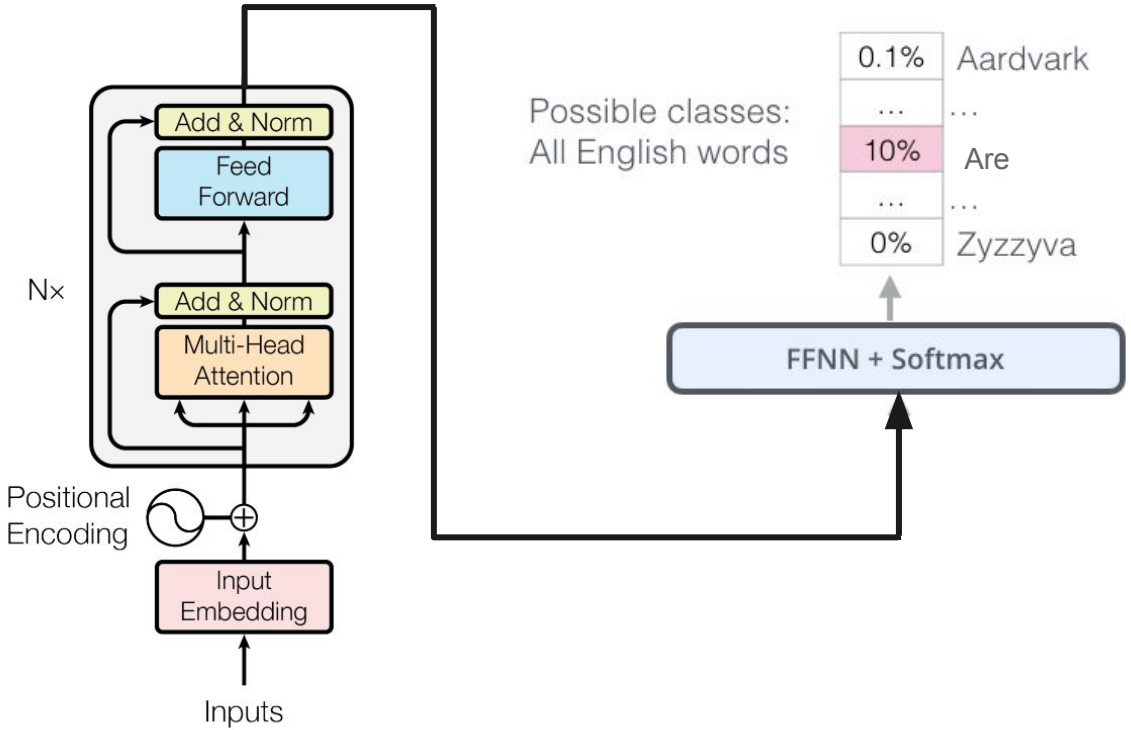
Token String		Token ID
this	->	
is	->	
token	->	
izing	->	
.	->	



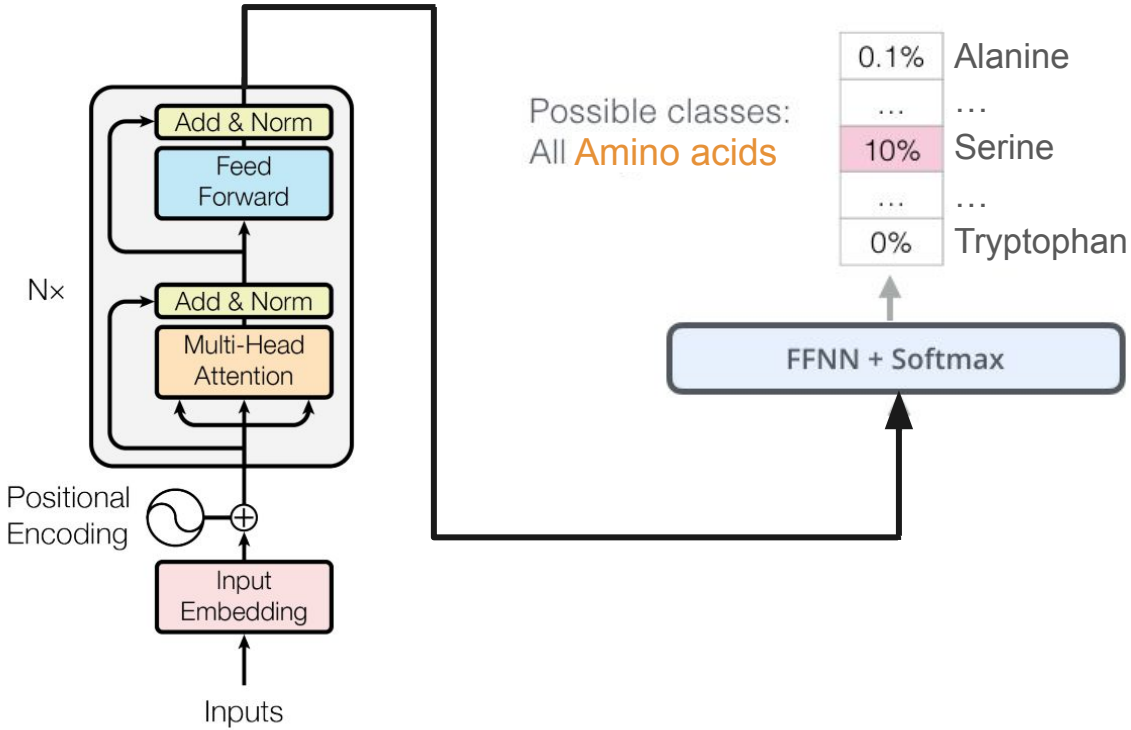
How do we get the outputs?



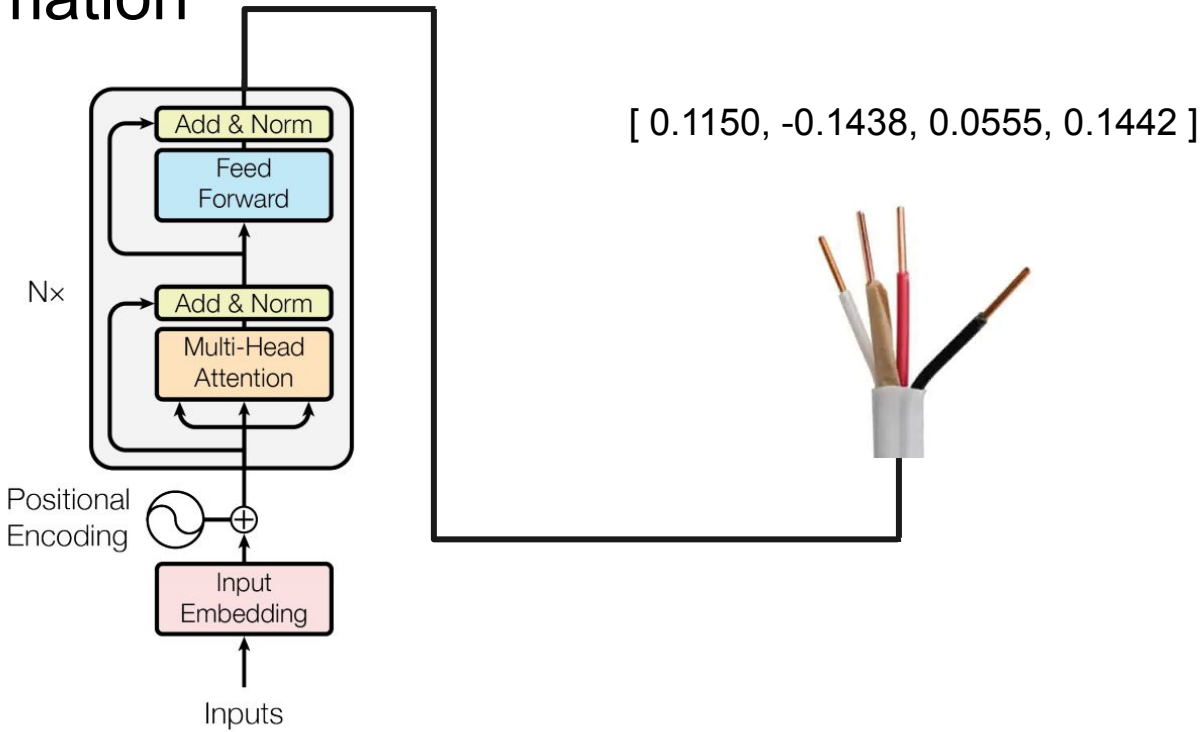
This training infuses the model with a general understanding of language



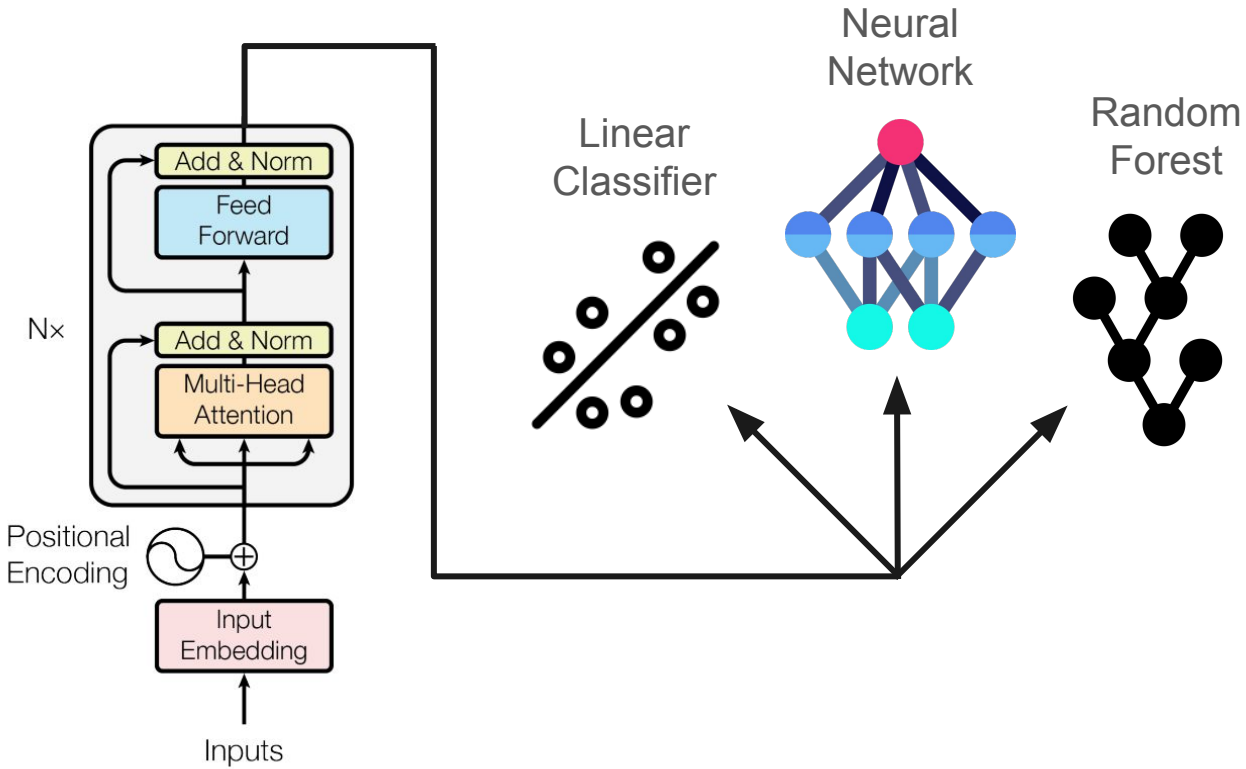
Or, in our case, proteins!



After pretraining, we have a model that has incorporated knowledge of protein-ness and outputs a vector encoding that information



We can use the encoder for downstream tasks (transfer learning)



Let’s recap:

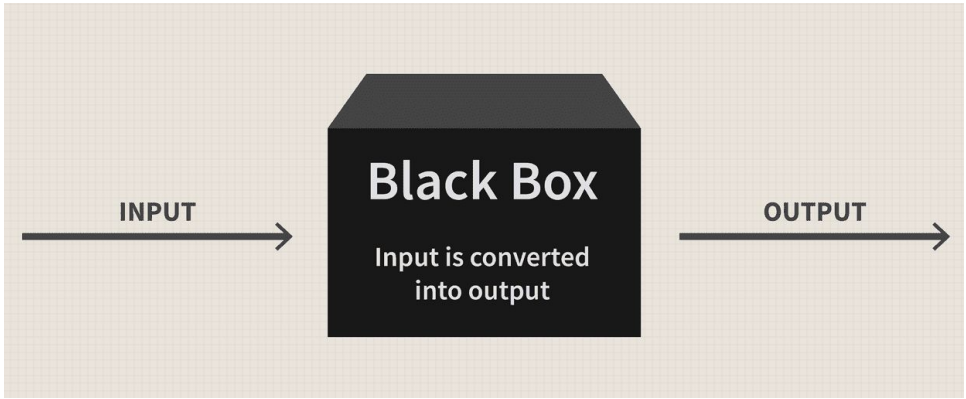
M L <mask> I R

↓

5, 8, 0, 6, 19

↓

[-0.07, 0.41, -0.02, …]
[0.11, -0.14, 0.05, …]
[-0.51, -0.06, -0.20, …]
[-0.34, 0.08, 0.44, …]
[0.10, -0.92, 0.10, …]



Let’s recap:

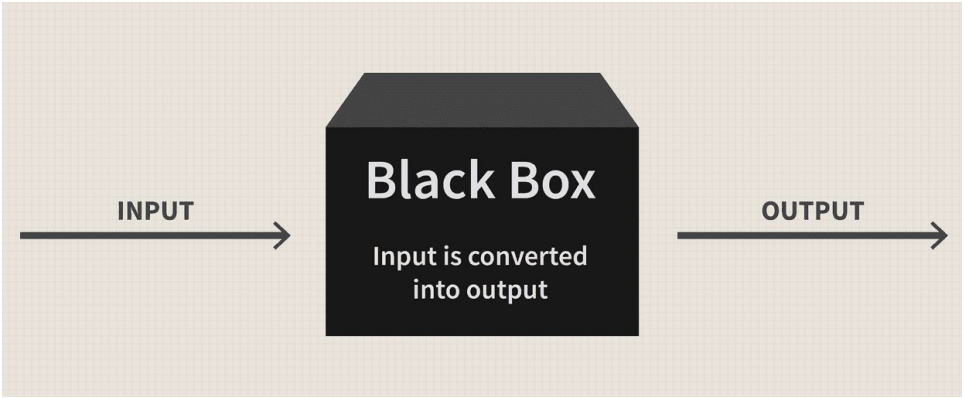
M L <mask> I R

↓

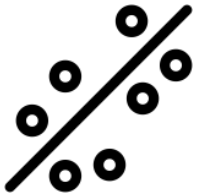
5, 8, 0, 6, 19

↓

[-0.07, 0.41, -0.02, …]
[0.11, -0.14, 0.05, …]
[-0.51, -0.06, -0.20, …]
[-0.34, 0.08, 0.44, …]
[0.10, -0.92, 0.10, …]



Linear Classifier

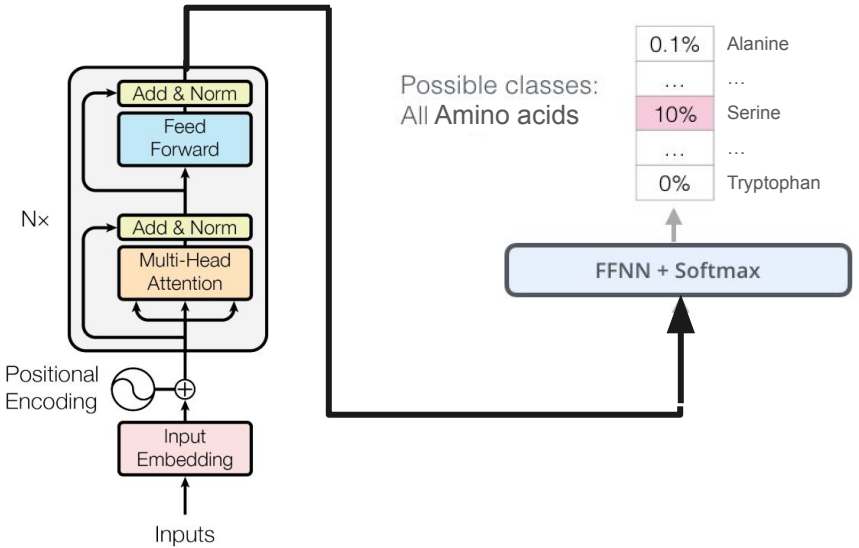


Roadmap

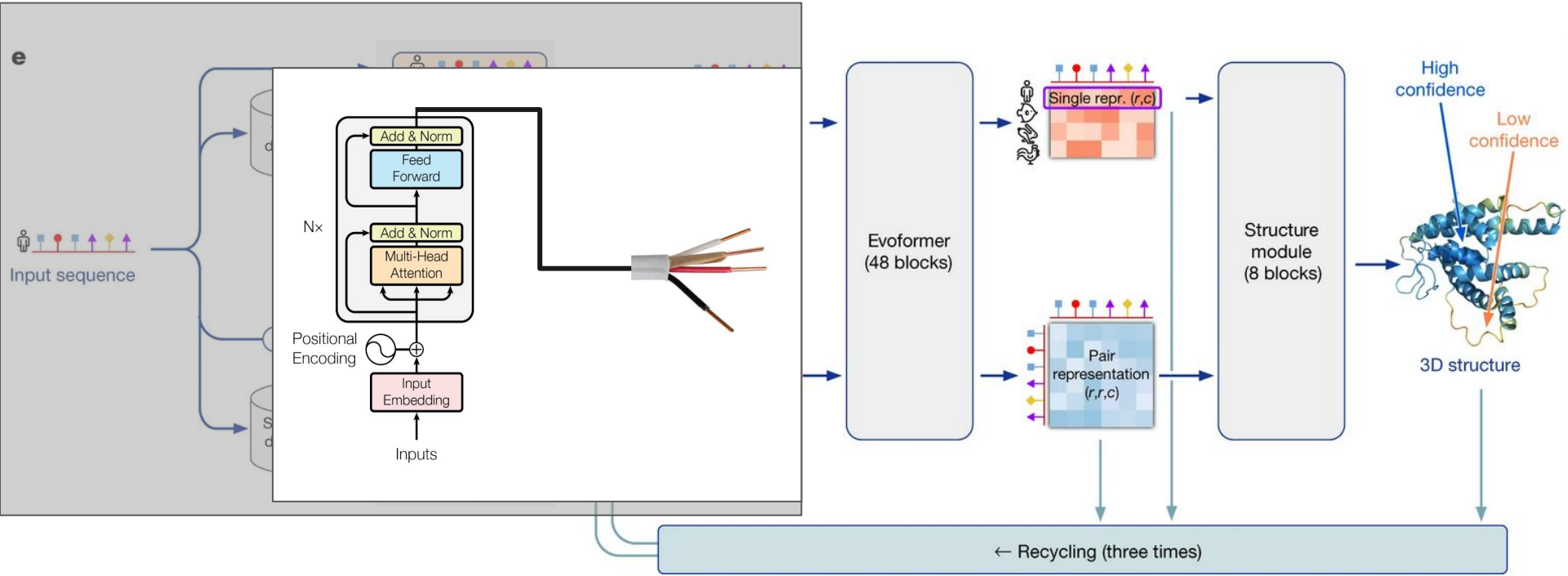
- Inside the black box: Neural Networks and Software 2.0
- A two-neuron toy example
- Transformers: attention and how we get language in and out
- **Applications, examples, and Google Colab**

Meta trained several large protein language models called Evolutionary Scale Model (ESM)

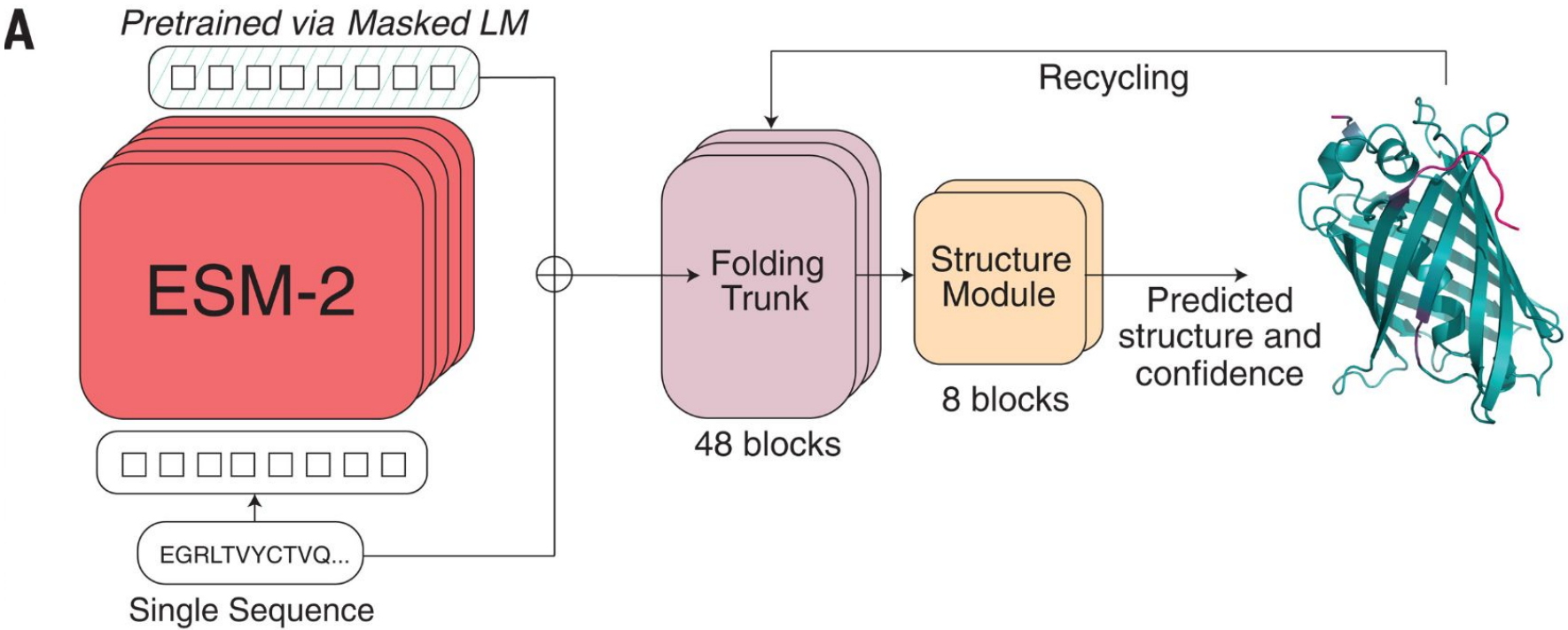




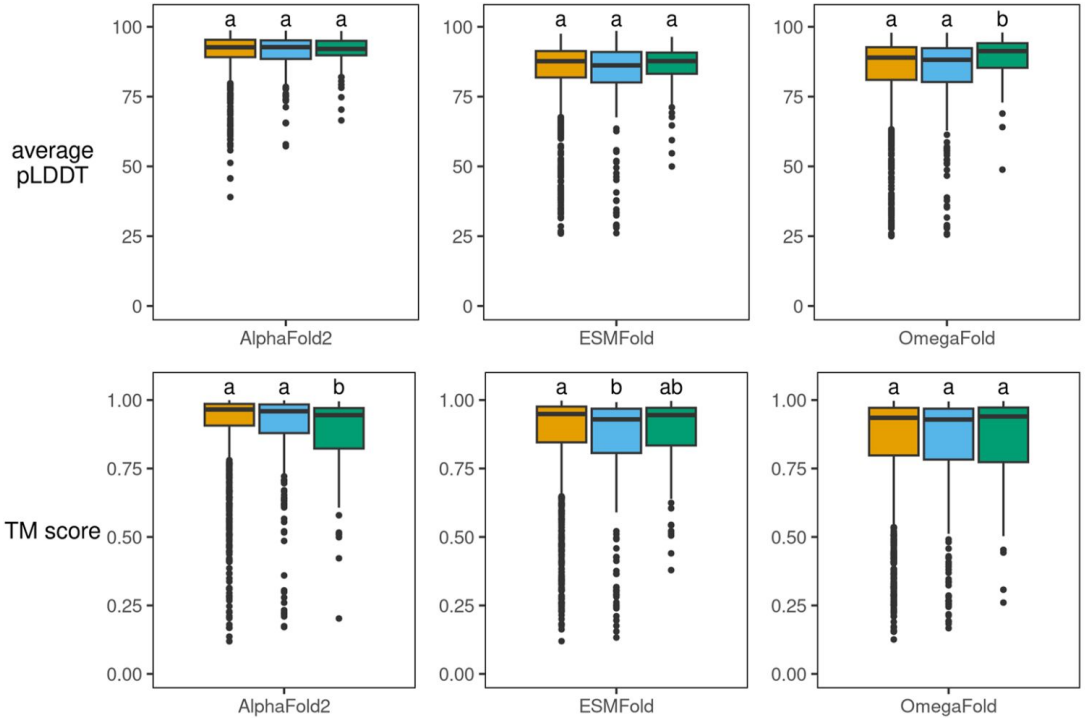
Alphafold's Evoformer and Structure model



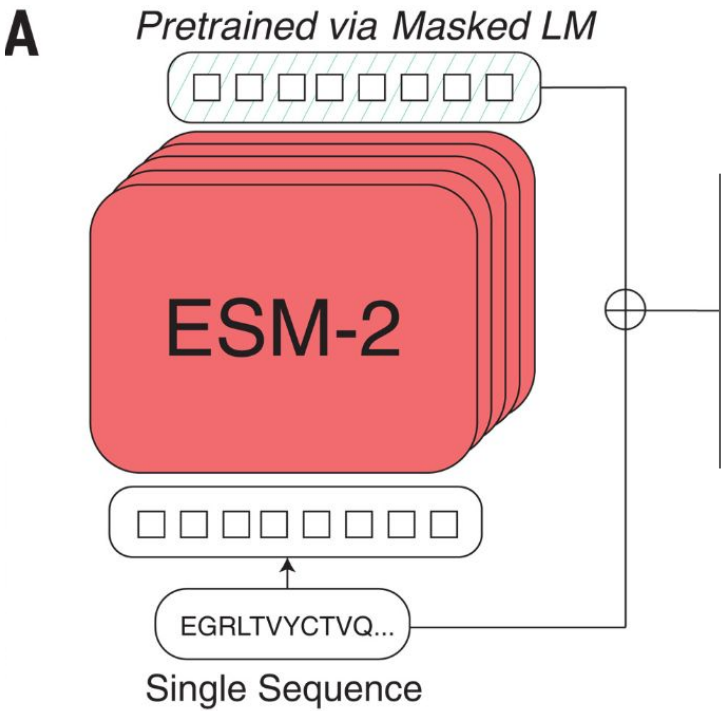
ESM-fold gets similar accuracy with a single sequence



It’s up to 60x faster, though accuracy is slightly lower

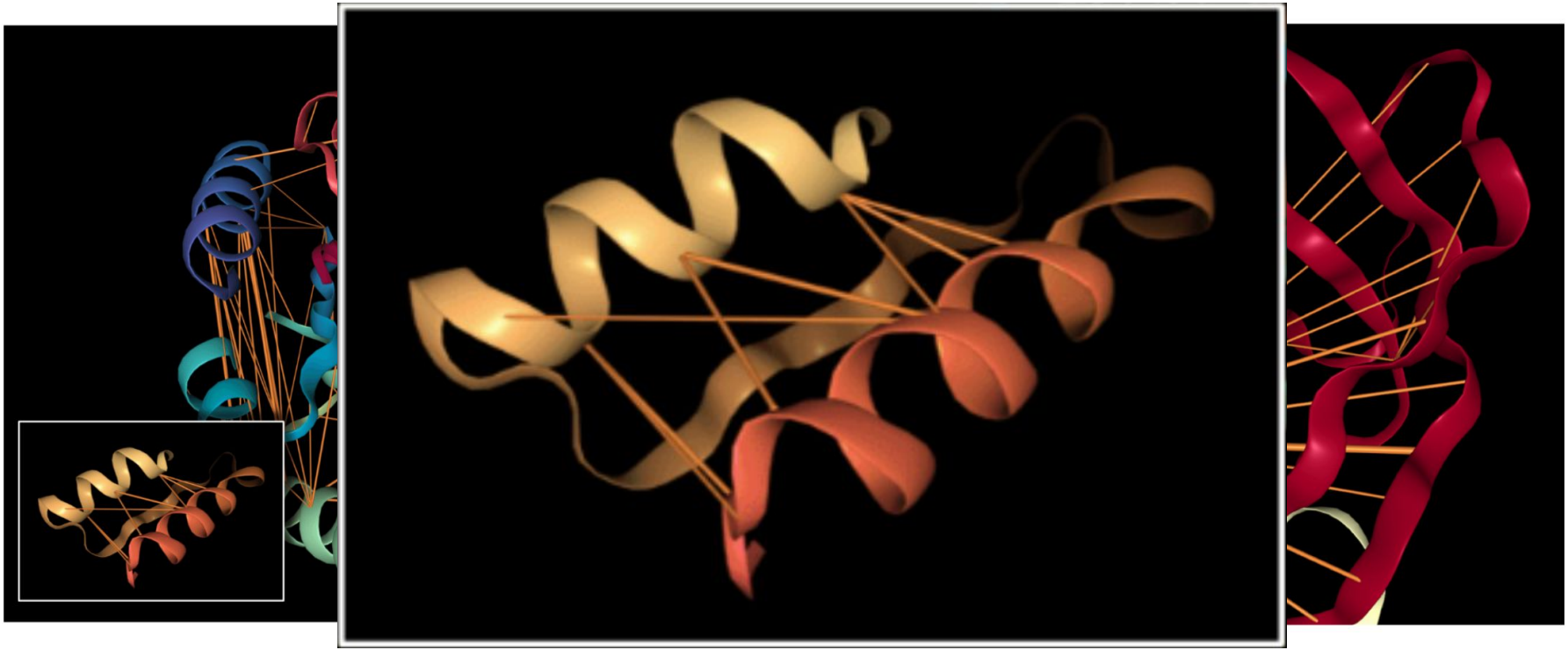


Models like ESM can excel at a variety of predictive tasks

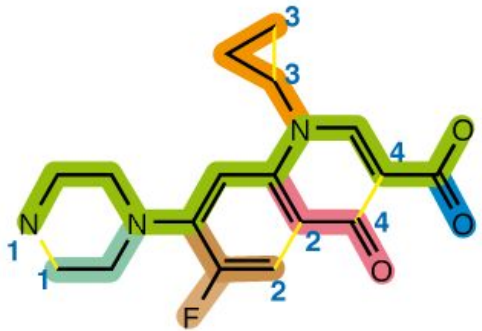


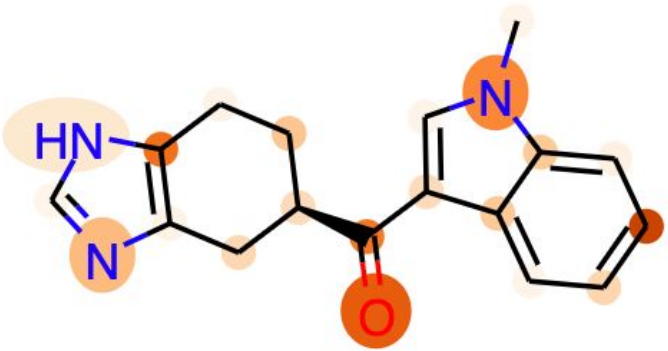
- Antimicrobial Efficacy
- Thermal Stability
- Subcellular Localization
- Enzyme Function Classification
- Drug Binding Affinity

We can probe the models to better understand them



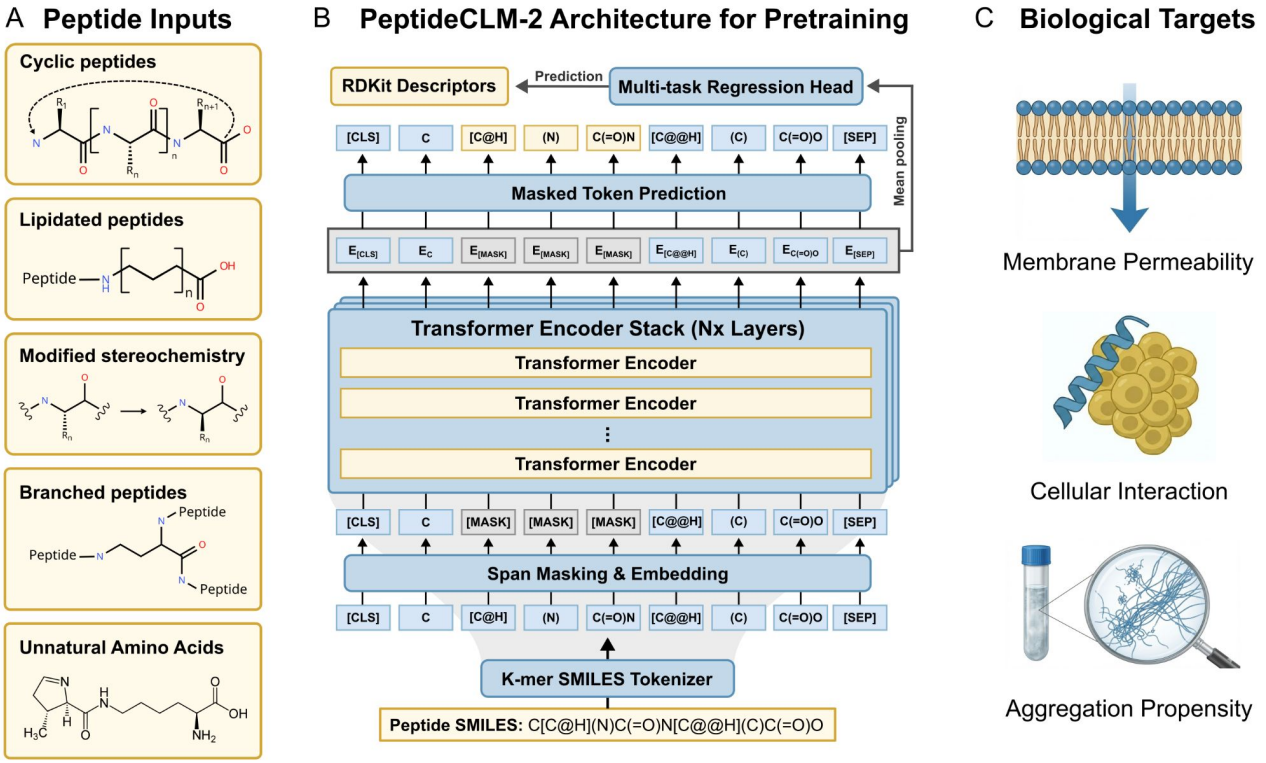
We can do the same thing for small molecules



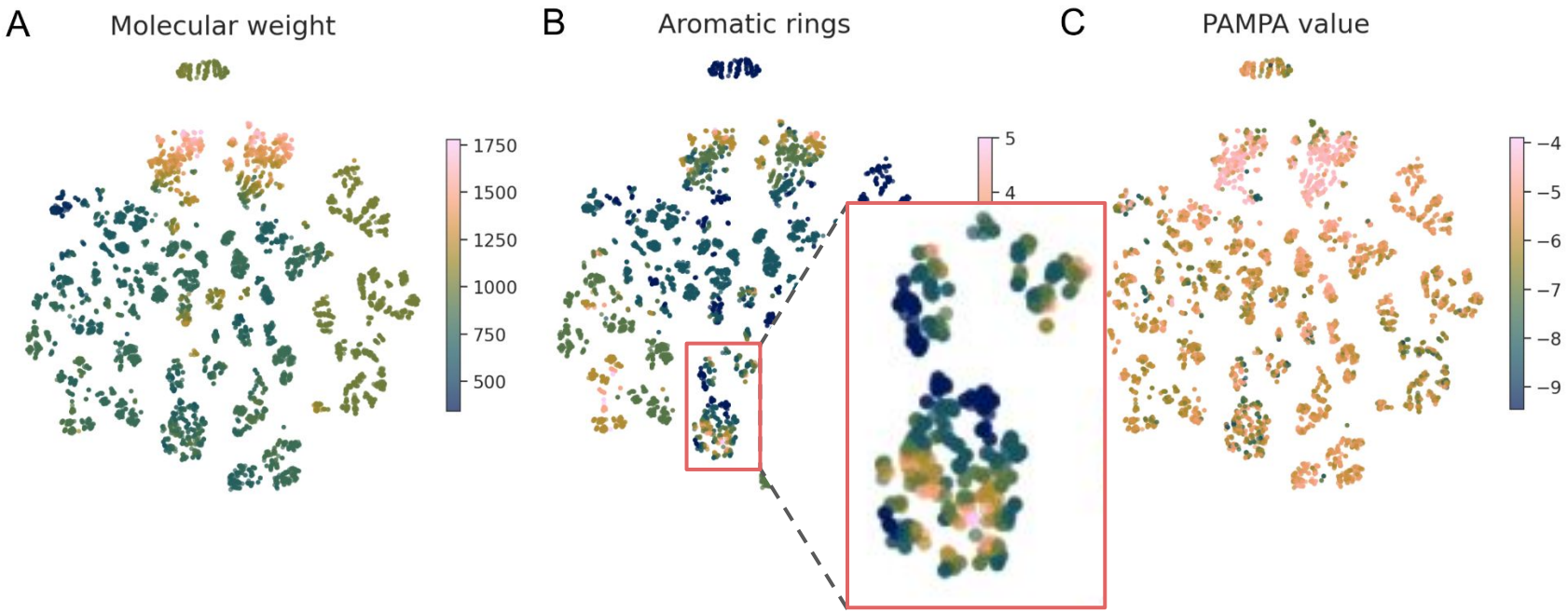


N1CCN(CC1)C(C(F)=C2)=CC(=C2C4=O)N(C3CC3)C=C4C(=O)O

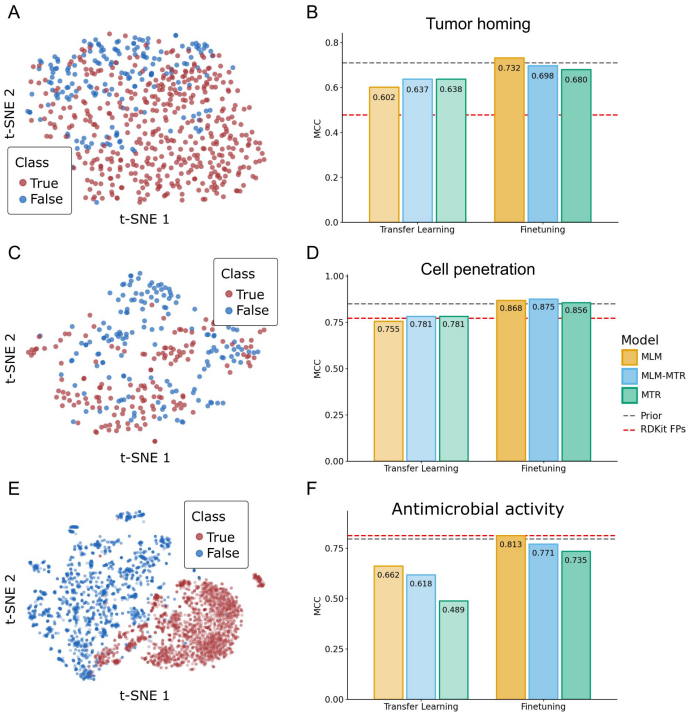
My work has focused on using SMILES for peptides

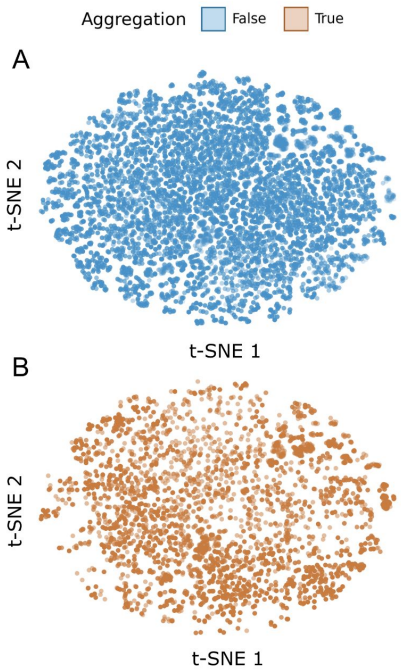


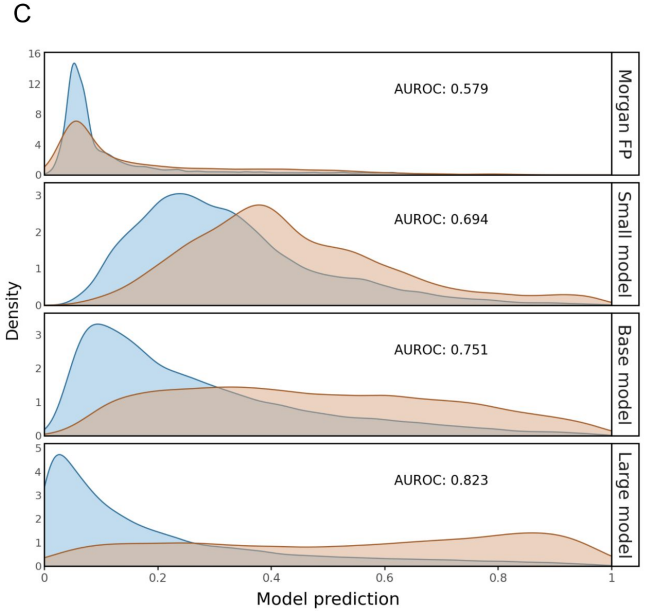
Model embeddings are able to separate by structural features and physical qualities



Finetuning the model results in higher accuracy metrics than custom architectures and static molecular fingerprints







Resources:

Metagenomic Atlas: https://esmatlas.com/

Github for ESMC (newest version): https://github.com/evolutionaryscale/esm

ESMFold: https://colab.research.google.com/github/sokrypton/ColabFold/blob/main/ESMFold.ipynb

Example Notebook:
https://colab.research.google.com/drive/13DgnFzTUJWU8luyU0SKhSjlqjEqz1YO9?usp=sharing

3blue1brown explanation of attention:
https://medium.com/lazy-by-design/attention-in-transformers-3blue1brown-fbba4e826ab4

BioML Society homepage:

www.biomlsociety.org