CS 839: Foundation Models
Models I

Fred Sala

University of Wisconsin-Madison

Sept. 19, 2023

Announcements

- **Announcement:**
  - Check out some of the posted papers!
  - Homework will start next week

- Class roadmap:

Tuesday Sept. 19	Models I
Thursday Sept. 21	Models II
Tuesday Sept. 23	Prompting I
Thursday Sept. 28	Prompting II
Tuesday Oct. 3	Reasoning & Chain-of-Thought

Mostly Language Models

Outline

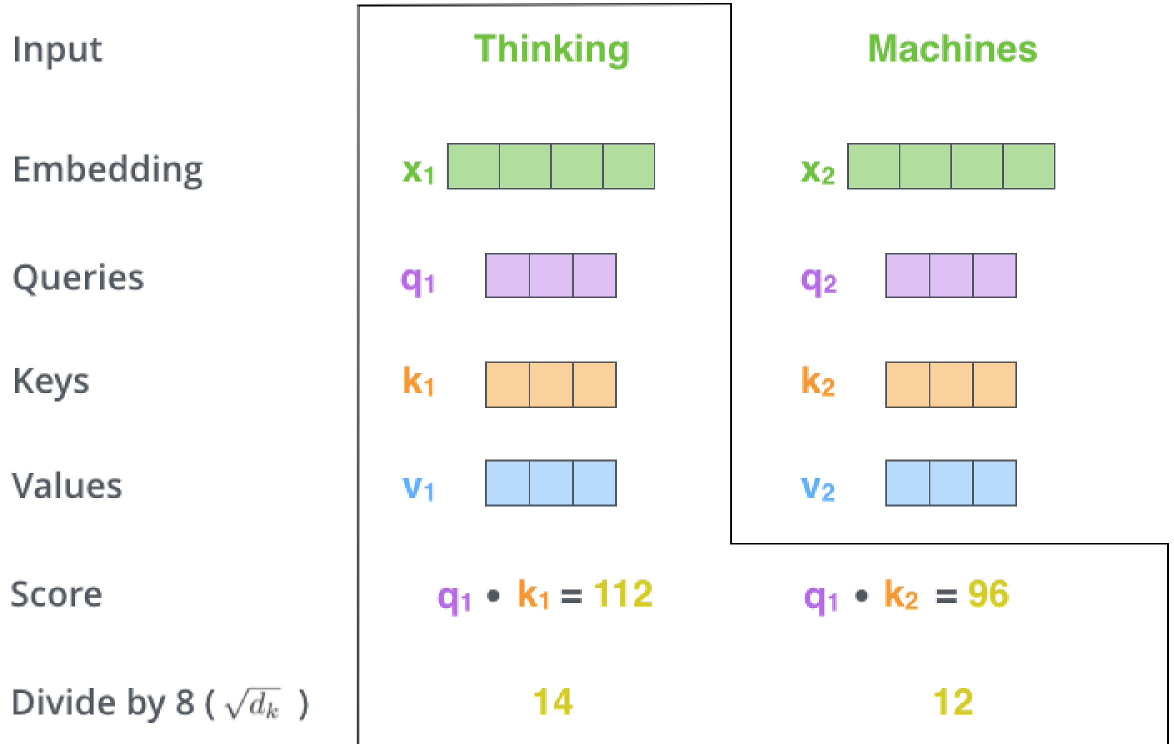
- **From Last Time**
  - Self-attention, transformers architecture
- **Encoder-only Models**
  - Example: BERT, architecture, multitask training, fine-tuning
- **Decoder-only Models**
  - Example: GPT, architecture, basic functionality

Outline

- **From Last Time**
  - Self-attention, transformers architecture
- **Encoder-only Models**
  - Example: BERT, architecture, multitask training, fine-tuning
- **Decoder-only Models**
  - Example: GPT, architecture, basic functionality

Self-Attention Review: Basic Operations

- Have **query, key, value** vectors via weight matrices: linear transformations!
- Have softmax score outputs (**focus**)
- Add up the values!



Objects:

Query
Key
Value

Credit to Jay Alammar: http://jalammar.github.io for many figures ☺

Self-Attention Review : Matrix Formulas

- Have **query, key, value** vectors via weight matrices: linear transformations!
- Have softmax score outputs (**focus**)
- Add up the values!

Objects:

Query
Key
Value

$$Q = XW_Q, K = XW_K, V = XW_V$$

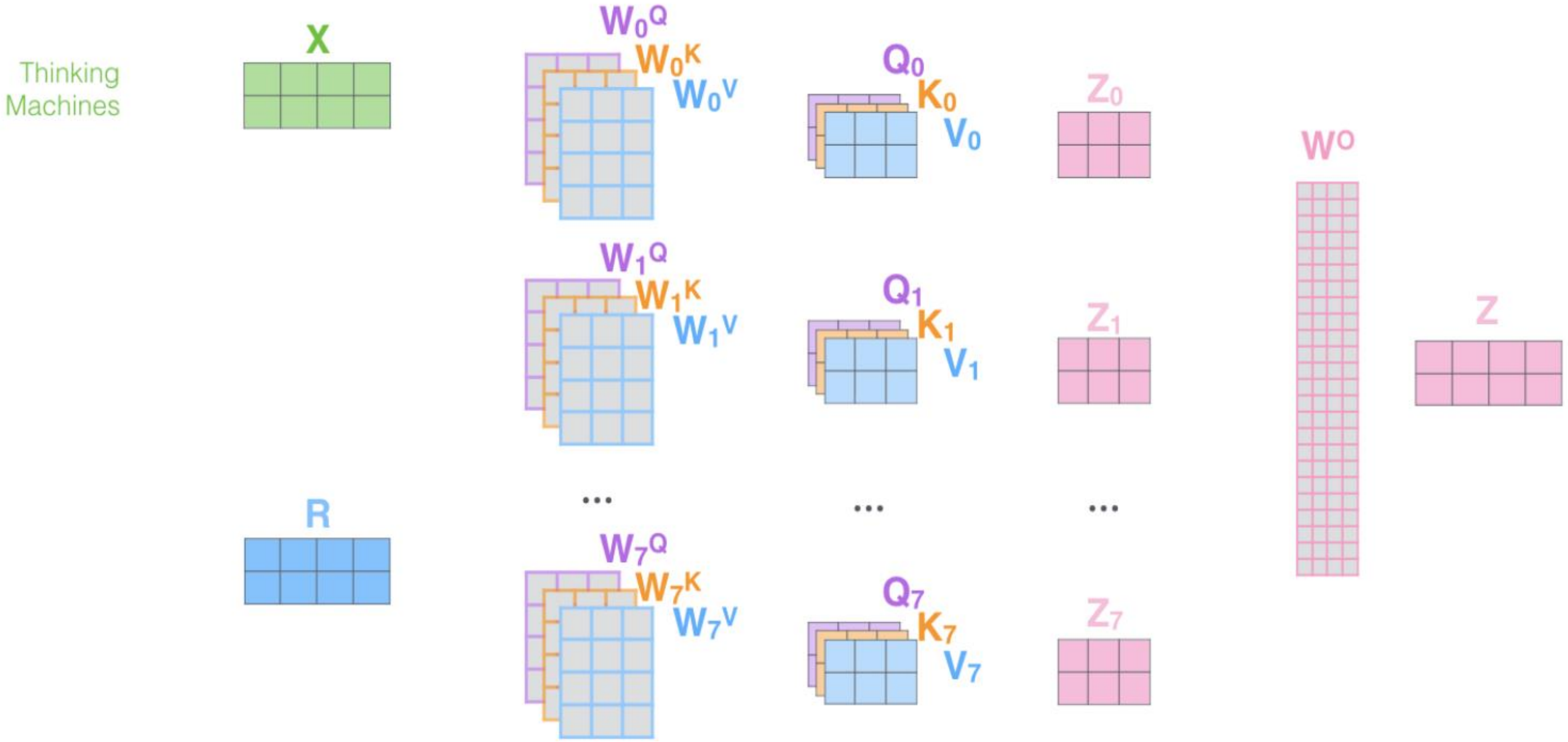
$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

$$\text{Attention}(Q, K, V) = \text{softmax}\left(X\frac{W_Q {W_K}^T}{\sqrt{d_k}}X^T\right)V$$

Self-Attention Review: Multi-head Attention

This is great but will we capture everything in one?

- Do we use just 1 kernel in CNNs? **No!**
- Do it many times in parallel: **multi-headed attention.** Concatenate outputs



Self-Attention Review: Position Encodings

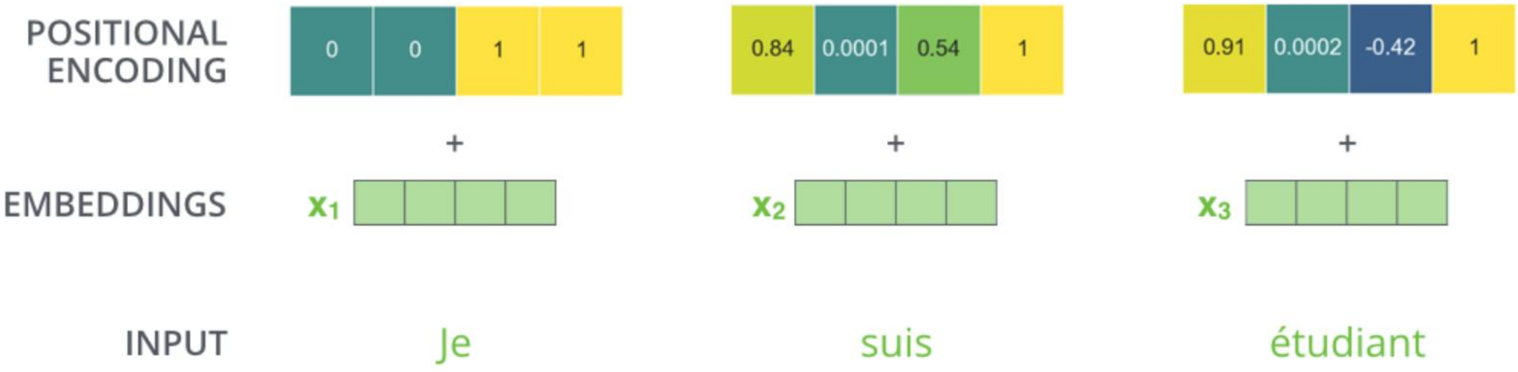
Almost have a full layer designed.

- One annoying issue: so far, order of words **(position) doesn't matter**!
- Solution: add positional encodings

$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$

$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

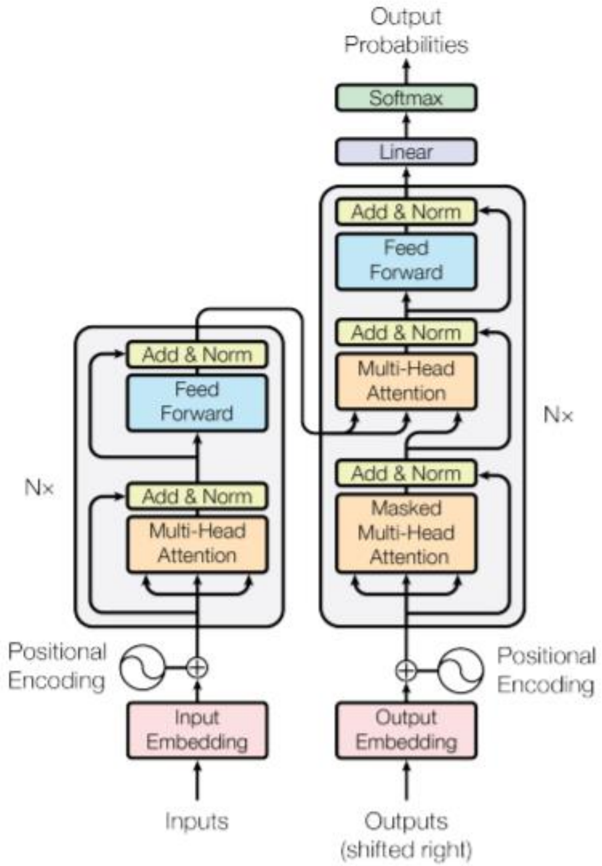
Location index



Transformers: Model Architecture

- Initial goal for an architecture: **encoder-decoder**
  - Get **rid of recurrence**
  - Replace with **self-attention**

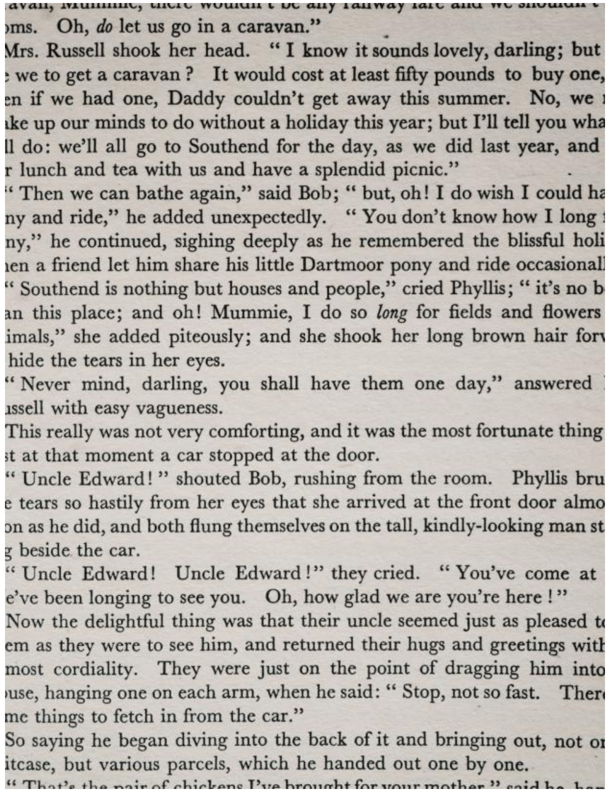
- Architecture
  - The famous picture you've seen
  - Centered on self-attention blocks



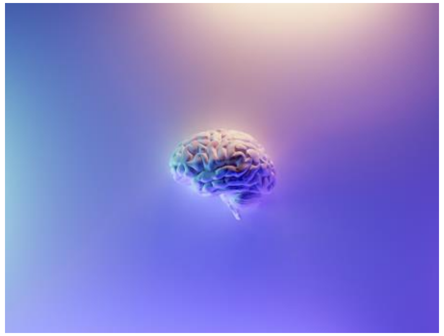
Vaswani et al. '17

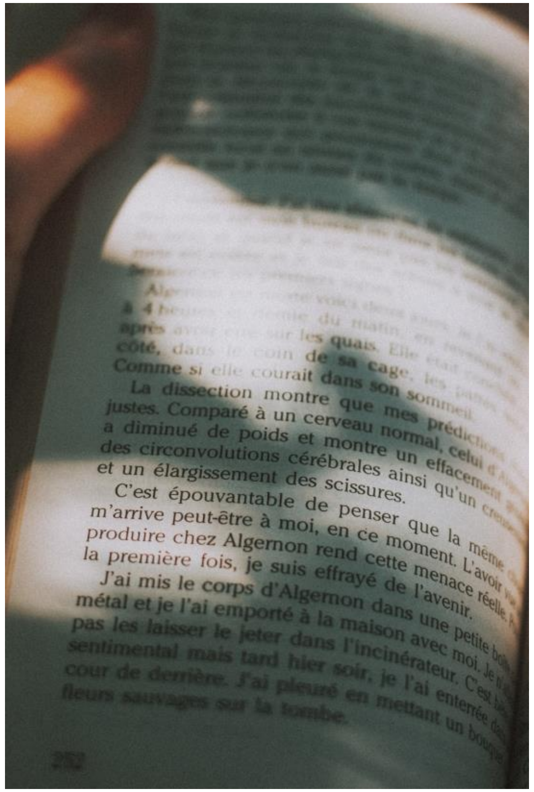
Interlude: Encoder-Decoder Models

- Translation tasks: natural encoder-decoder architecture
- Intuition:



oms. Oh, *do* let us go in a caravan."
Mrs. Russell shook her head. "I know it sounds lovely, darling; but
we to get a caravan? It would cost at least fifty pounds to buy one,
en if we had one, Daddy couldn't get away this summer. No, we
ke up our minds to do without a holiday this year; but I'll tell you wha
ll do: we'll all go to Southend for the day, as we did last year, and
r lunch and tea with us and have a splendid picnic."
"Then we can bathe again," said Bob; "but, oh! I do wish I could ha
ny and ride," he added unexpectedly. "You don't know how I long
ny," he continued, sighing deeply as he remembered the blissful holi
ien a friend let him share his little Dartmoor pony and ride occasionall
"Southend is nothing but houses and people," cried Phyllis; "it's no b
an this place; and oh! Mummie, I do so *long* for fields and flowers
imals," she added piteously; and she shook her long brown hair forw
hide the tears in her eyes.
"Never mind, darling, you shall have them one day," answered
ussell with easy vagueness.
This really was not very comforting, and it was the most fortunate thing
st at that moment a car stopped at the door.
"Uncle Edward!" shouted Bob, rushing from the room. Phyllis bru
e tears so hastily from her eyes that she arrived at the front door almo
on as he did, and both flung themselves on the tall, kindly-looking man st
g beside the car.
"Uncle Edward! Uncle Edward!" they cried. "You've come at
e've been longing to see you. Oh, how glad we are you're here!"
Now the delightful thing was that their uncle seemed just as pleased t
em as they were to see him, and returned their hugs and greetings with
most cordiality. They were just on the point of dragging him into
ouse, hanging one on each arm, when he said: "Stop, not so fast. There
me things to fetch in from the car."
So saying he began diving into the back of it and bringing out, not or
itcase, but various parcels, which he handed out one by one.

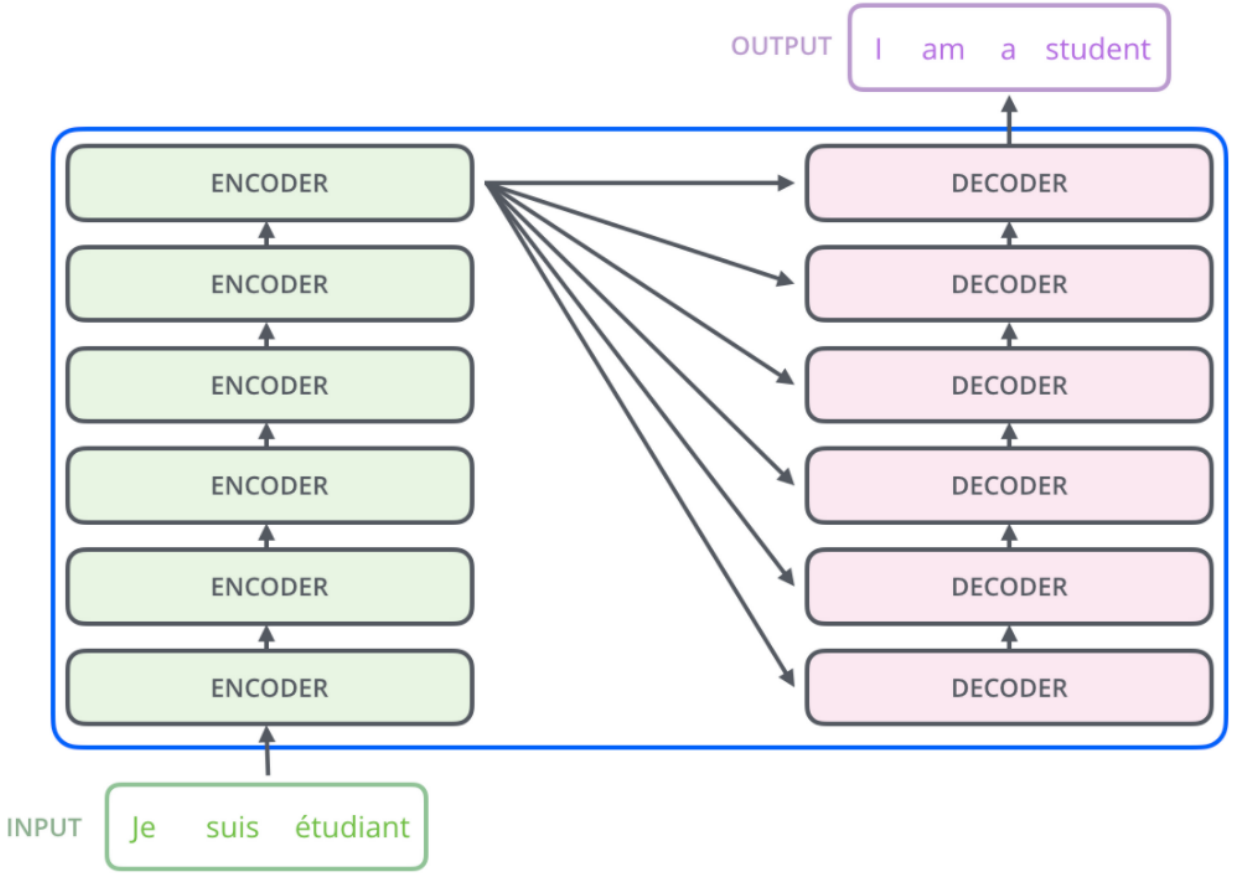




Comme si elle courait dans son sommeil.
La dissection montre que mes prédictions
justes. Comparé à un cerveau normal, celui d'A
a diminué de poids et montre un effacement
des circonvolutions cérébrales ainsi qu'un
et un élargissement des scissures.
C'est épouvantable de penser que la même
m'arrive peut-être à moi, en ce moment. L'avoir
produire chez Algernon rend cette menace réelle
la première fois, je suis effrayé de l'avenir.
J'ai mis le corps d'Algernon dans une petite bo
métal et je l'ai emporté à la maison avec moi. Je
pas les laisser le jeter dans l'incinérateur. C'est
sentimental mais tard hier soir, je l'ai enterré da
cour de derrière. J'ai pleuré en mettant un bouquet
fleurs sauvages sur la tombe.

Transformers: Architecture

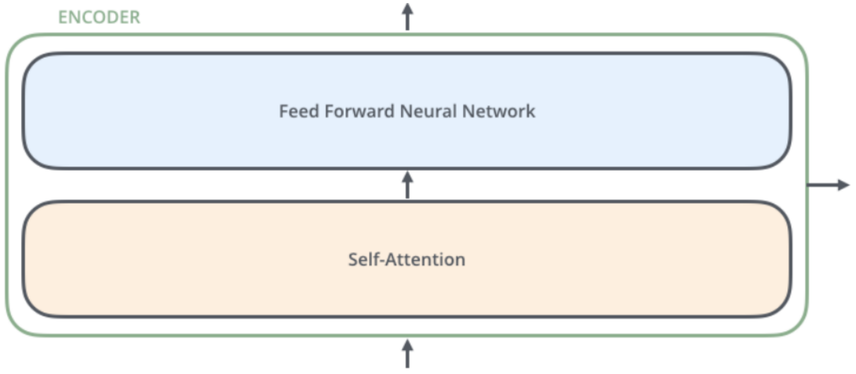
- **Sequence-sequence** model with **stacked** encoders/decoders:
  - For example, for French-English translation:

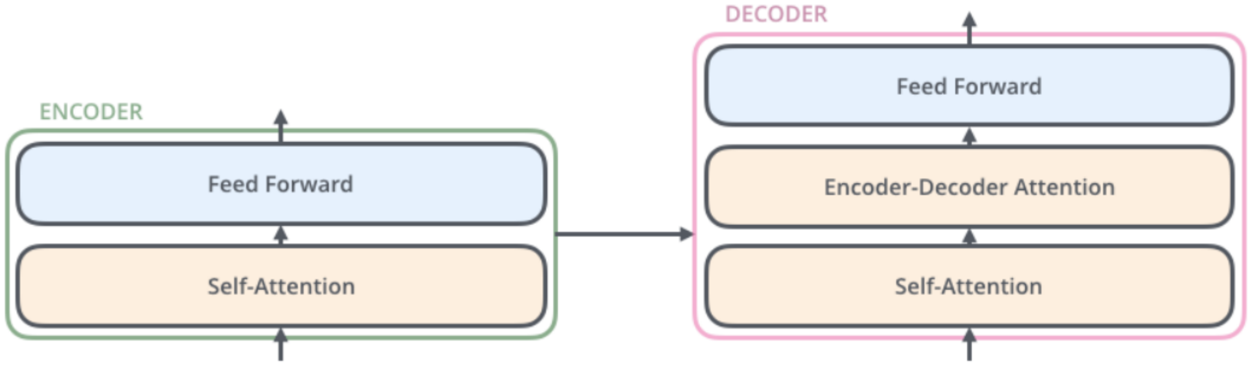


Transformers: Architecture

- Sequence-sequence model with **stacked** encoders/decoders:
  - What's inside each encoder/decoder unit?

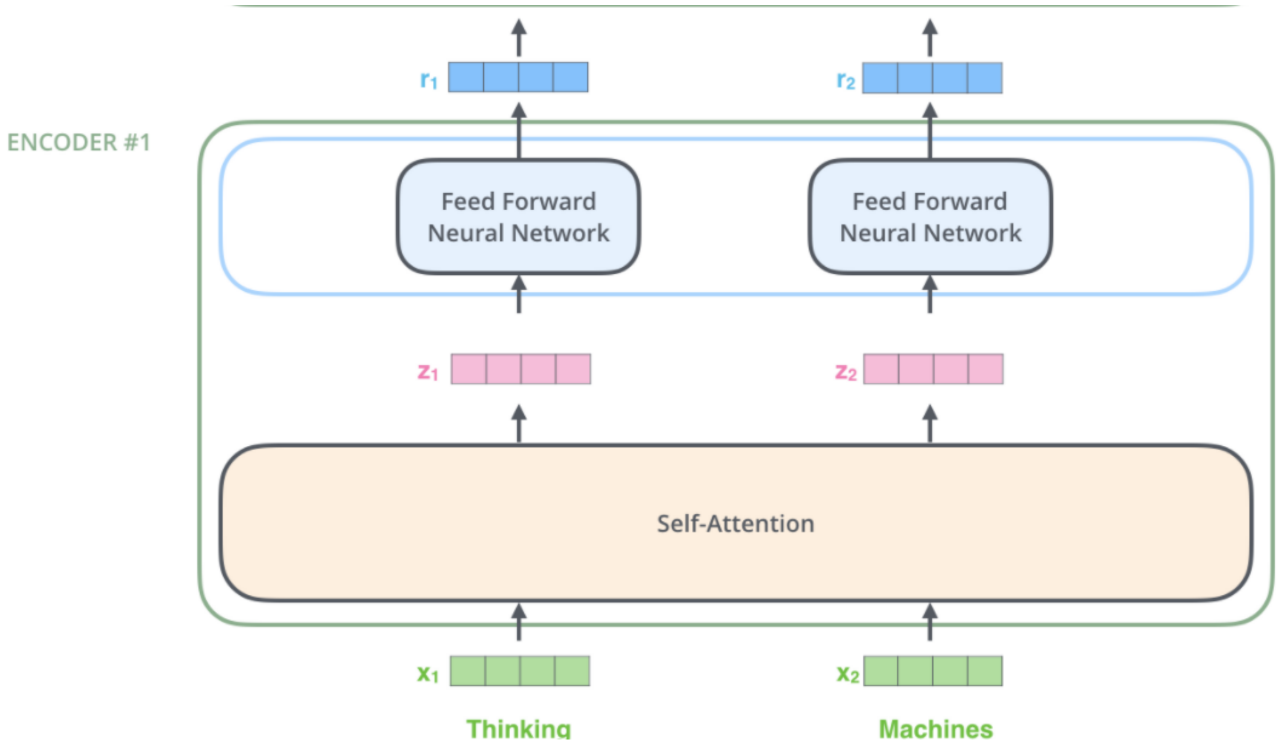
  - Focus on encoder first: **pretty simple**! 2 components:
    - Self-attention block
    - Fully-connected layers (i.e., an MLP)
    - Captures **1) interactions 2) processing** (separately!)





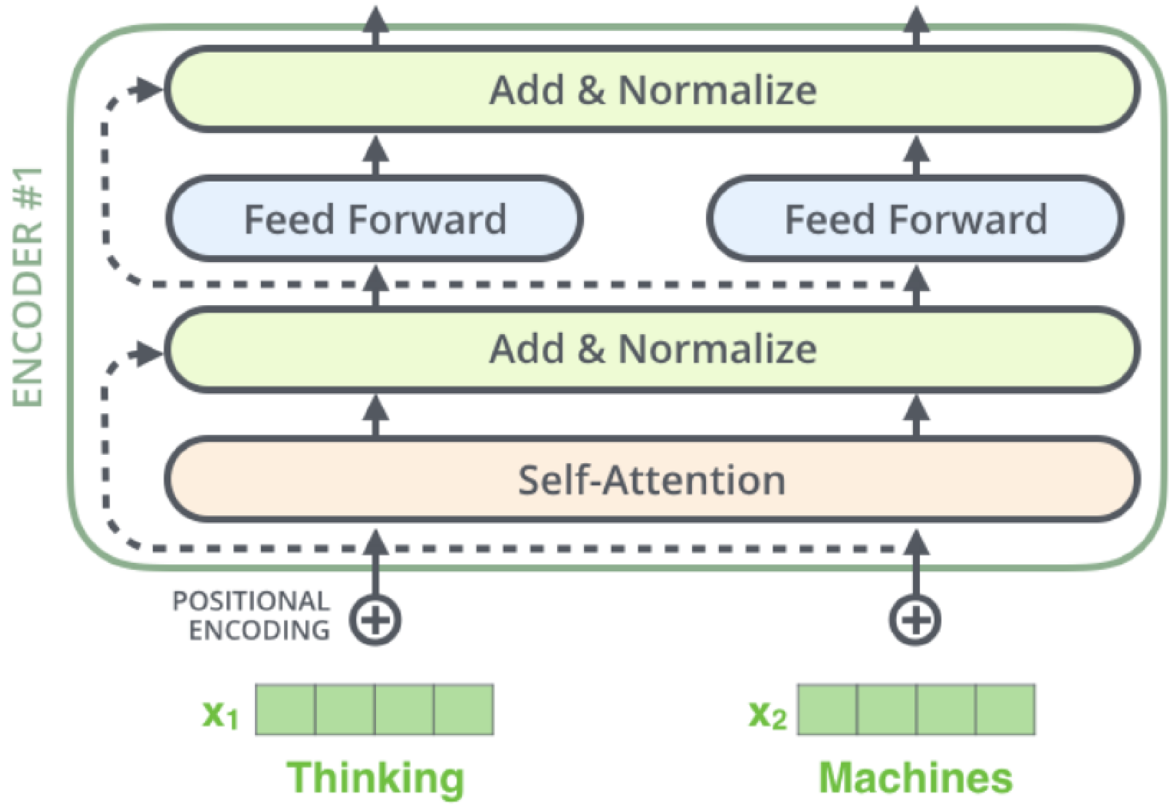
Transformers: Inside an Encoder

- Let's take a look at the encoder. Two components:
  - 1. **Self-attention** layer (covered this)
  - 2. "Independent" **feedforward nets** for each head



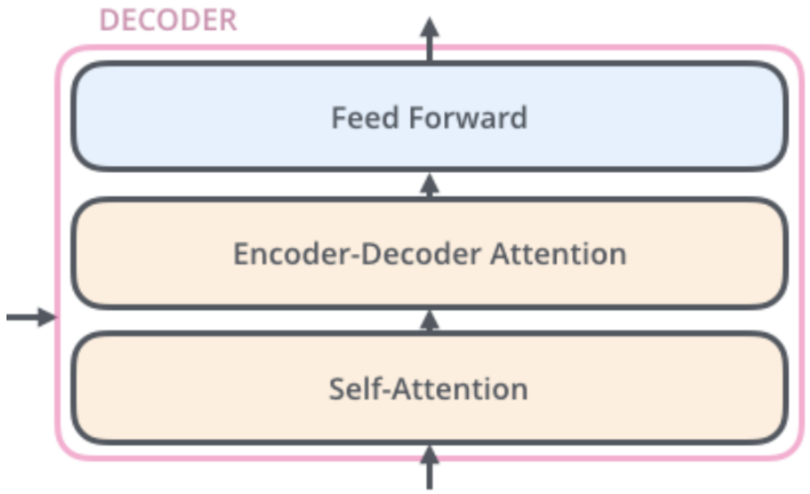
Transformers: More Tricks

- Recall a big innovation for ResNets: residual connections
  - And also layer normalizations
  - Apply to our encoder layers



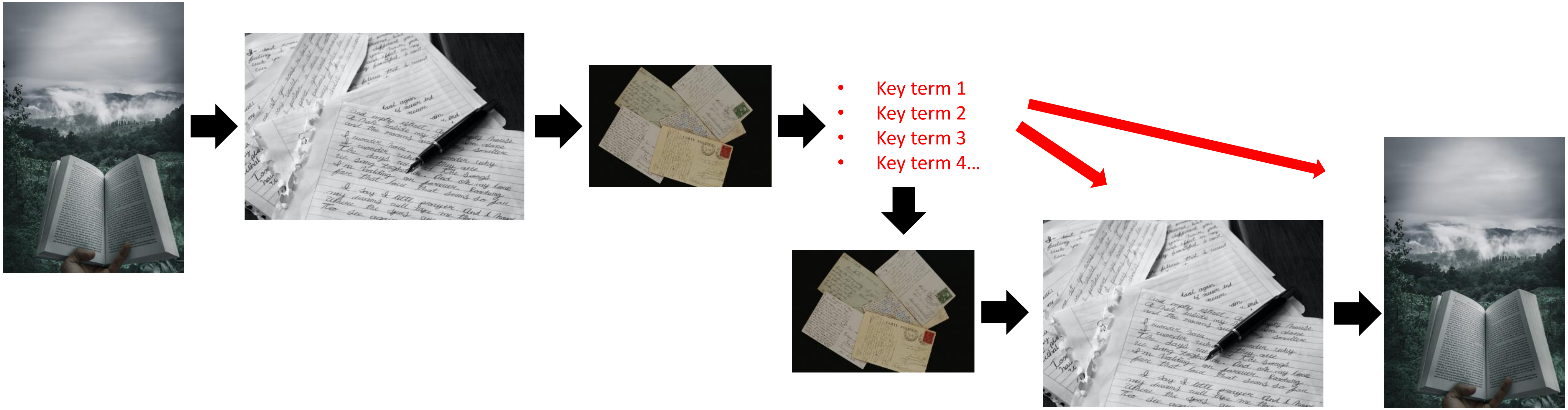
Transformers: Inside a Decoder

- Let's take a look at the decoder. Three components:
  - 1. **Self-attention** layer (covered this)
  - 2. Encoder-decoder attention **(same, but K, V come from encoder)**
  - 3. "Independent" feedforward nets for each head



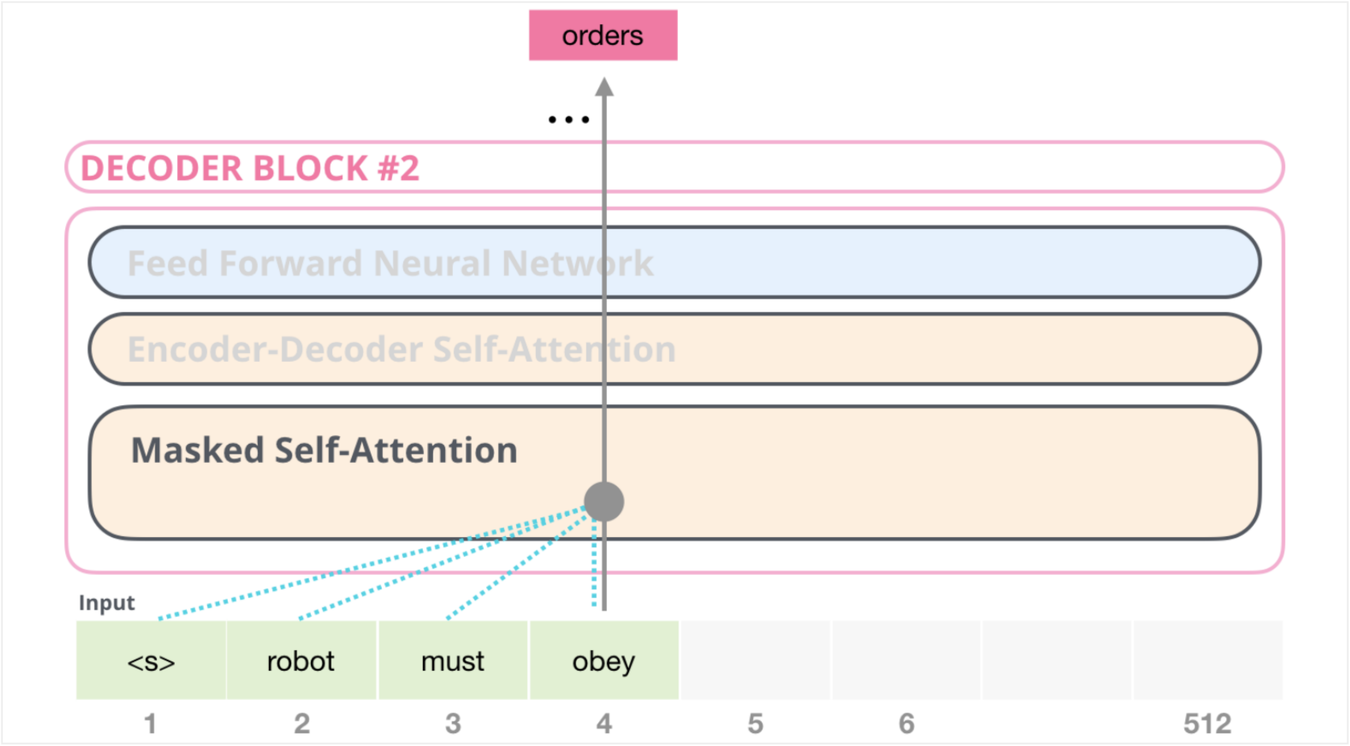
Transformers: Cross-Attention

- Why encoder-decoder attention ?
  - Recall: same as before, but K, V come from encoder
  - Actually more traditional, but… **intuition**:



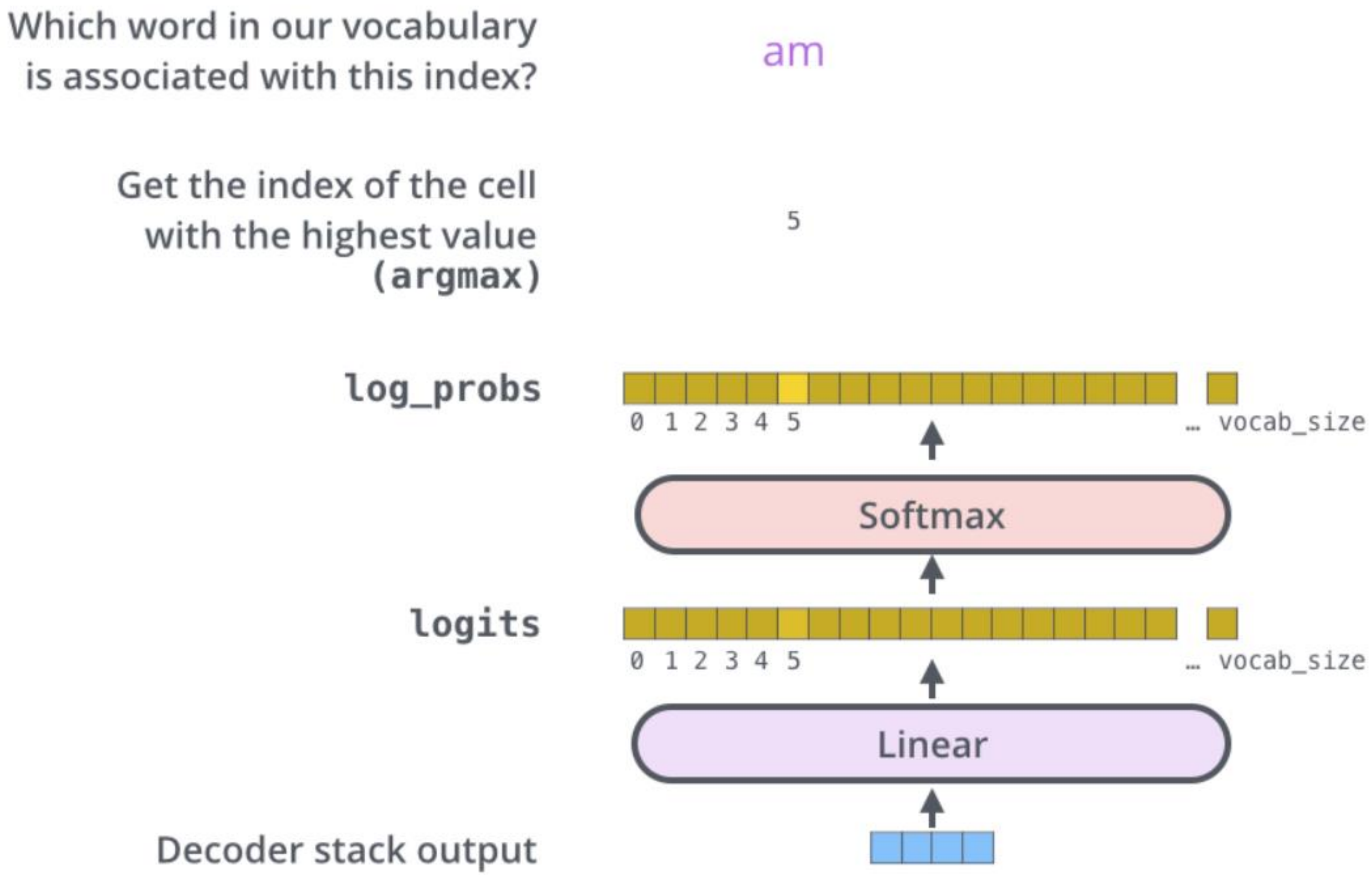
Transformers: Decoder Masking

- One more interesting bit!
  - At the decoder level, self-attention changes a bit:
  - Masked instead: block *future* words from being attended to



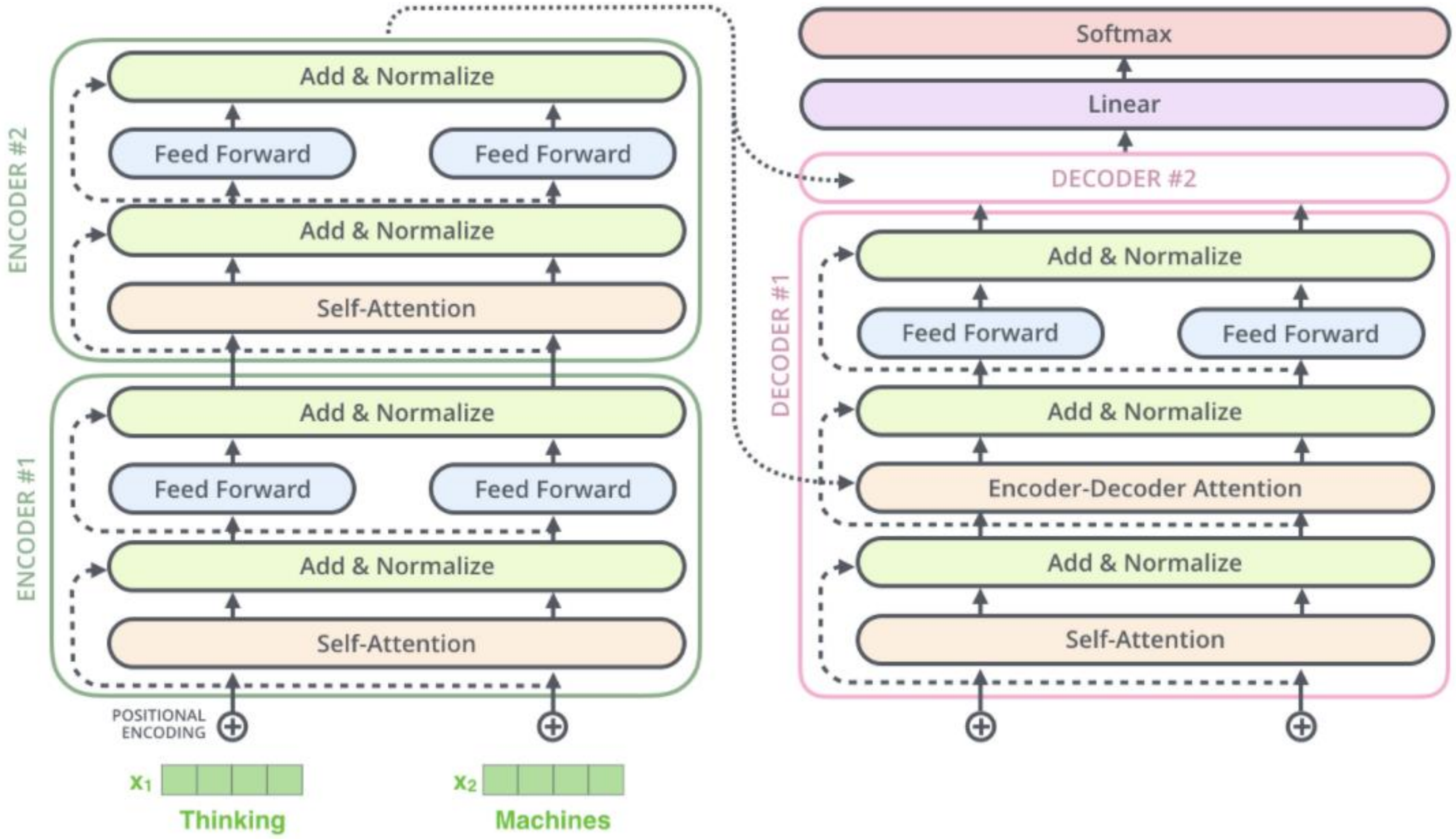
Transformers: Outputs

- Finally, let's see the final layer and outputs



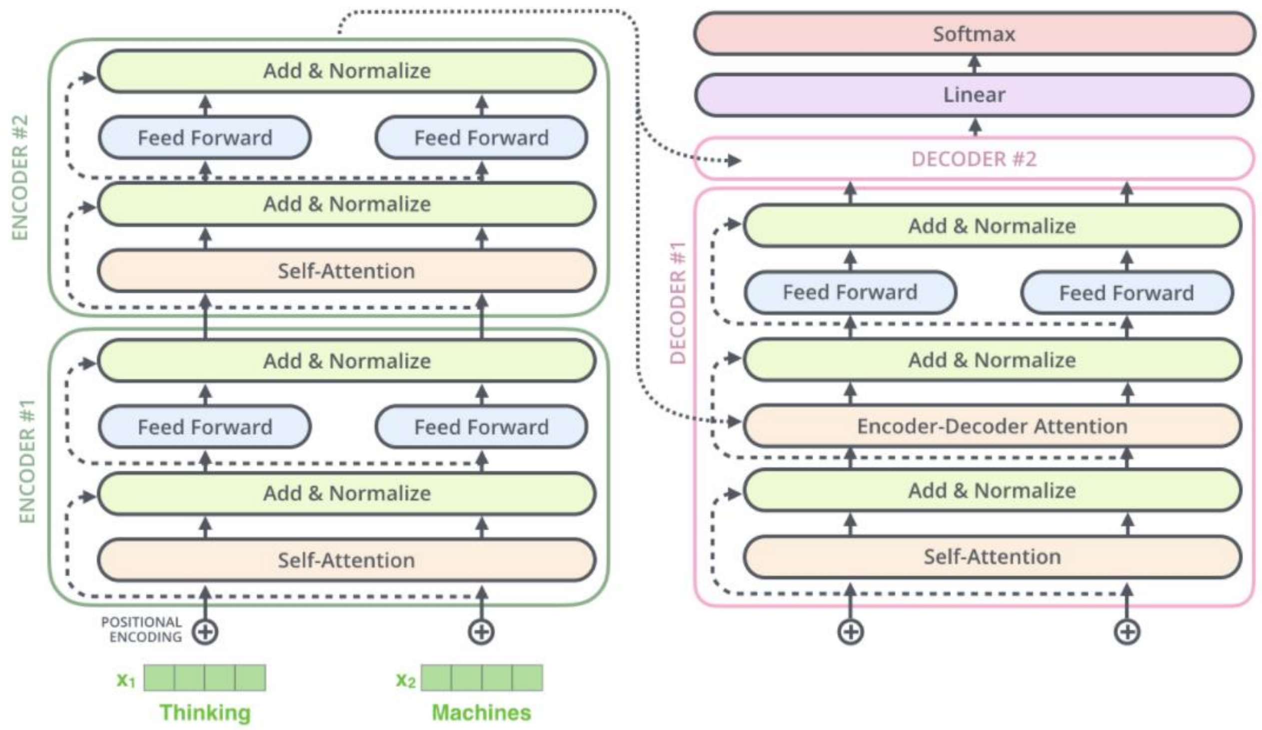
Transformers: Putting it All Together

- What does the full architecture look like?



Transformers: Training

- Data: standard datasets (WMT English-German)
  - ~5 million pairs for this dataset
  - Nothing very special: Adam optimizer



Break & Questions

Outline

- **From Last Time**
  - Self-attention, transformers architecture
- **Encoder-only Models**
  - Example: BERT, architecture, multitask training, fine-tuning
- **Decoder-only Models**
  - Example: GPT, architecture, basic functionality

Why Encoder-Decoder?

Wanted two things for translation:

- 1) **Outputs** in natural language
- 2) Tight alignment with **input**

What happens if we relax these?

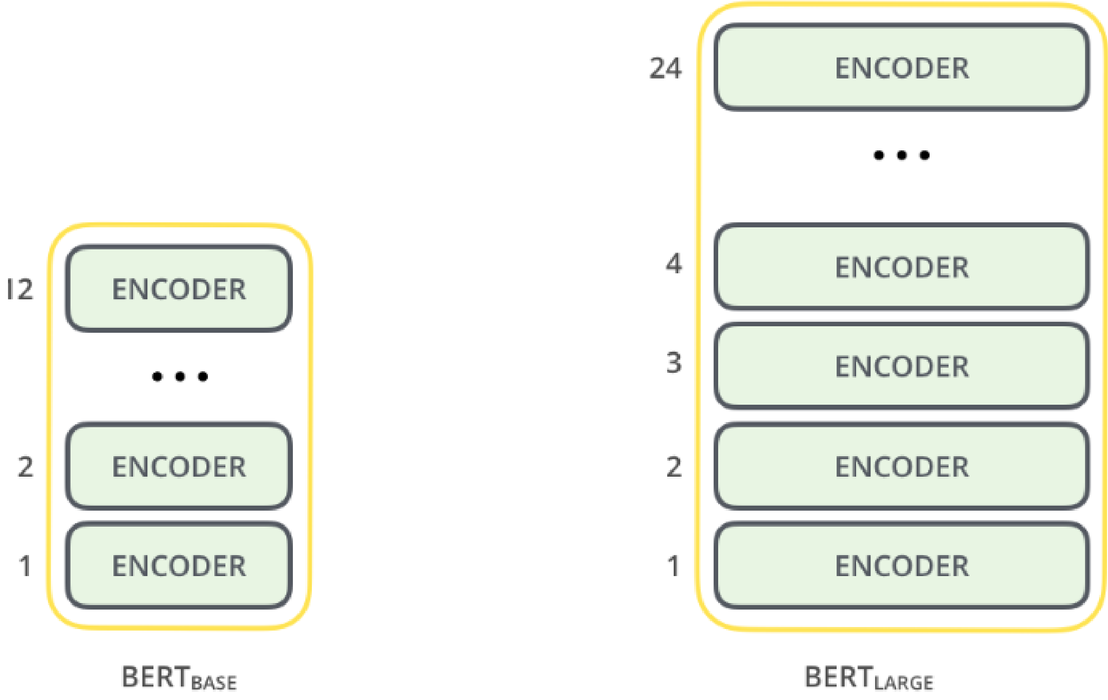
1. Encoder-only models
2. Decoder-only models

Encoder-Only Models: BERT

Let’s get rid of the first part

- 1) **Outputs** in natural language
- 2) Tight alignment with **input**

- Rip away decoders
  - Just stack encoders



Interlude: Contextual Embeddings

Q: Why is it called "BERT"?

- **A**: In a sense, follows up ELMo

- Story:
  - **2013**: "Dense" word embeddings (**Word2Vec, Glove**)
  - Downside: fixed representations per word
    - "Bank": building or riverside?
  - Need: contextual representations
    - Using language model-like techniques
    - 2018: ELMo, BERT
    - ELMo: uses LSTMs, BERT uses trasnformers

https://nlp.stanford.edu/projects/glove/

Interlude: Contextual Embeddings

Q: Why is it called “BERT”?

- **A**: In a sense, follows up ELMo

BERT acronym:

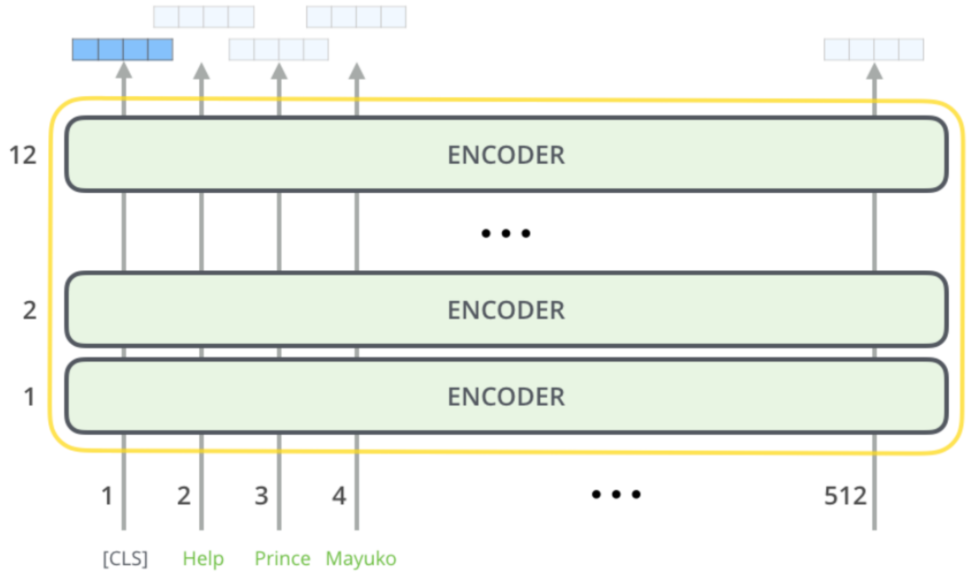
- **B**idirectional **E**ncoder **R**presentations from **T**ransformers.
- ERT should make sense,

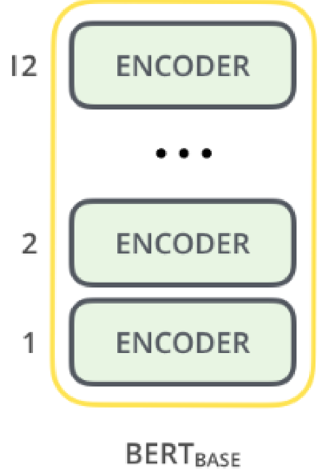
- Bidirectional: no causal masks, look at both sides of a word!
- Captured in self-attention block

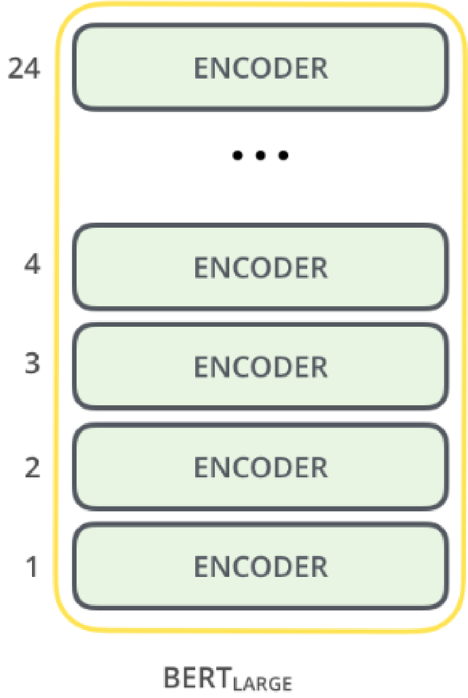
BERT: Forward Pass

BERT architecture

- Rip away decoders
  - Just stack encoders





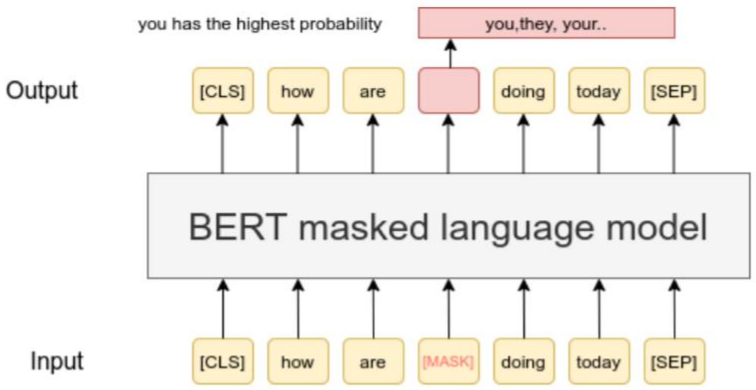


BERT: Training

Training is more interesting!

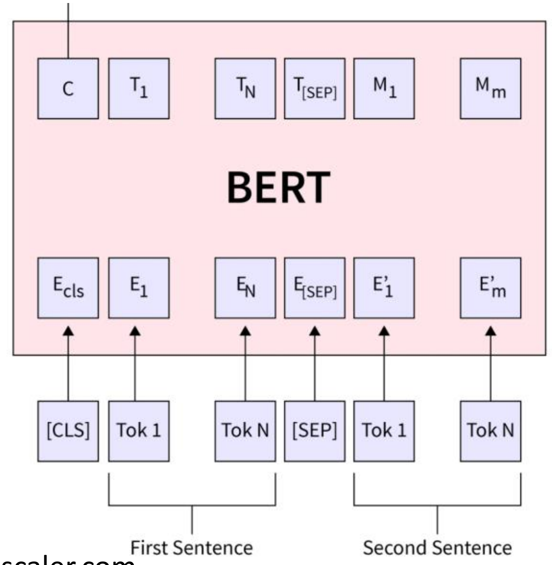
- Pretraining. Then fine-tuning on task of interest

- Back to **self-supervised learning!**
- Two tasks for **pretraining**.



sbert.net

1. Masked Language Modeling



scaler.com

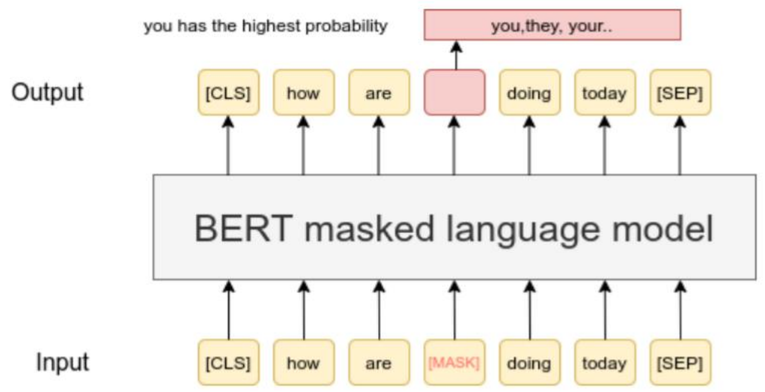
2. Next Sentence Prediction

BERT: Training Task 1

Masked Language Modeling Task

- Use [MASK] token for word to be predicted

- Which words to mask?
  - Original paper: 15% of words at random
  - But… of these
    - 10% of the time, no [MASK], flip word randomly
    - 10% of the time leave word unchanged

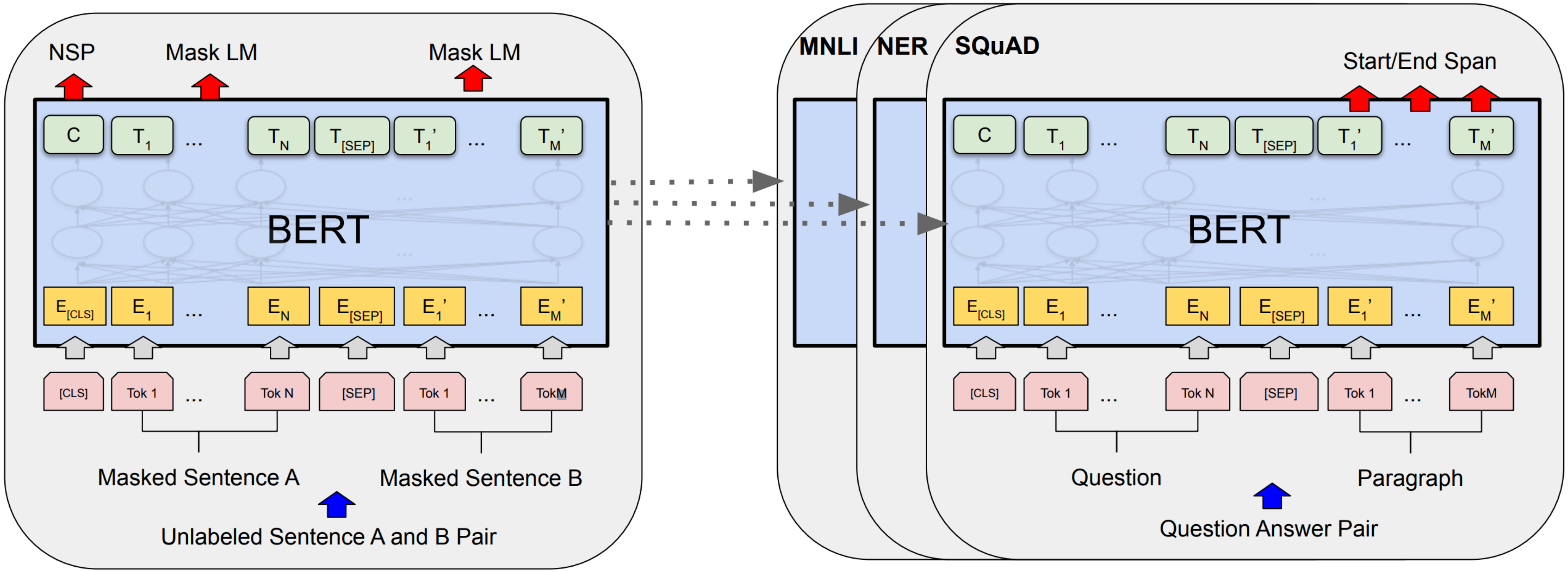


sbert.net

BERT: Training

Training is more interesting,

- Pretraining. Then fine-tuning on task of interest



Pre-training

Fine-Tuning

Devlin et al

Break & Questions

Outline

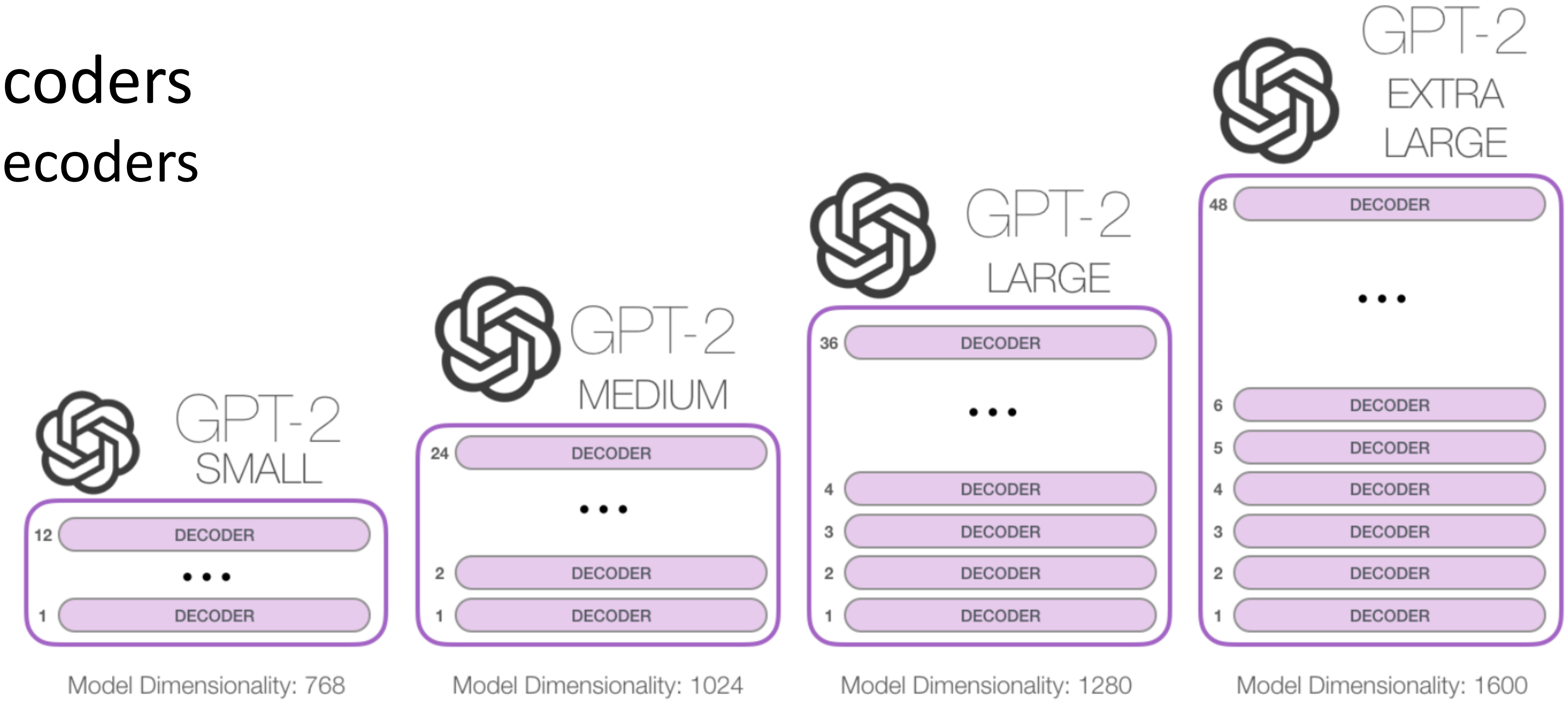
- **From Last Time**
  - Self-attention, transformers architecture
- **Encoder-only Models**
  - Example: BERT, architecture, multitask training, fine-tuning
- **Decoder-only Models**
  - Example: GPT, architecture, basic functionality

Decoder-Only Models: GPT

Let’s get rid of the first part

- 1) **Outputs** in natural language
- 2) Tight alignment with **input**

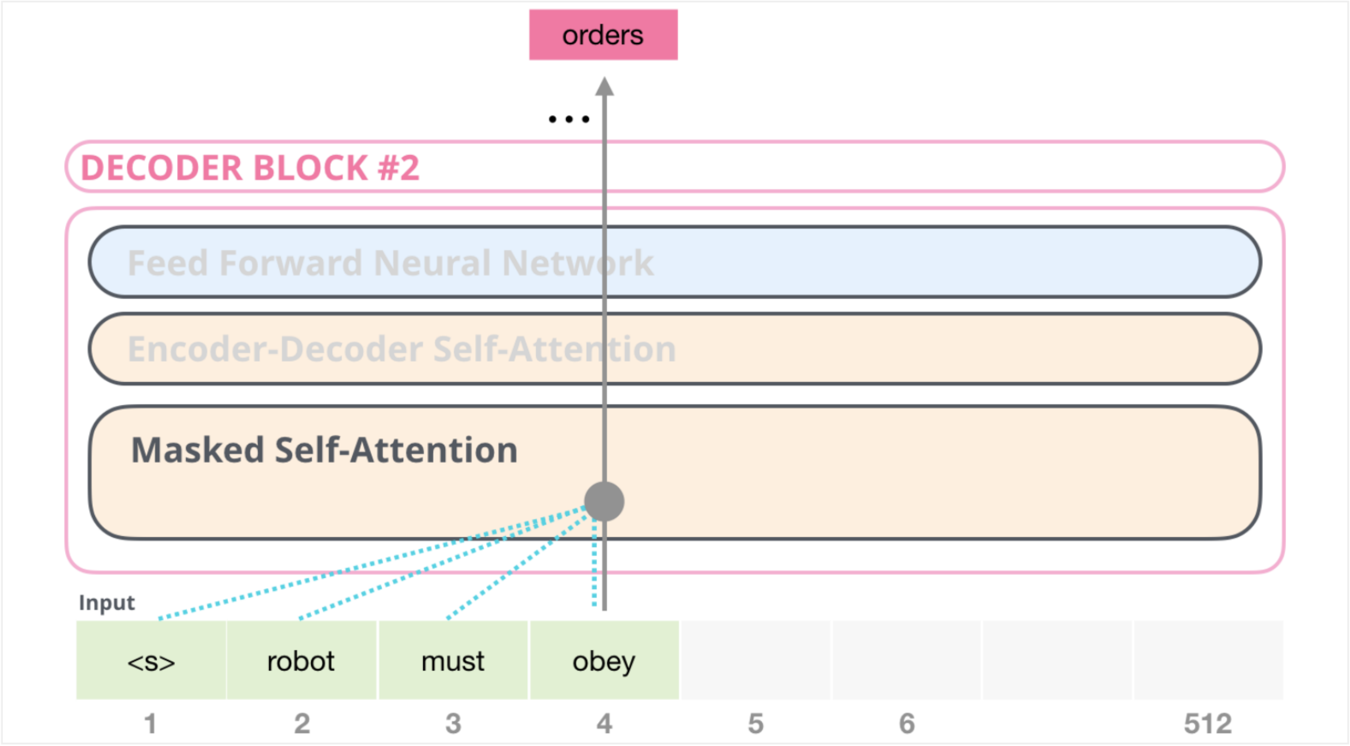
- Rip away encoders
    - Just stack decoders

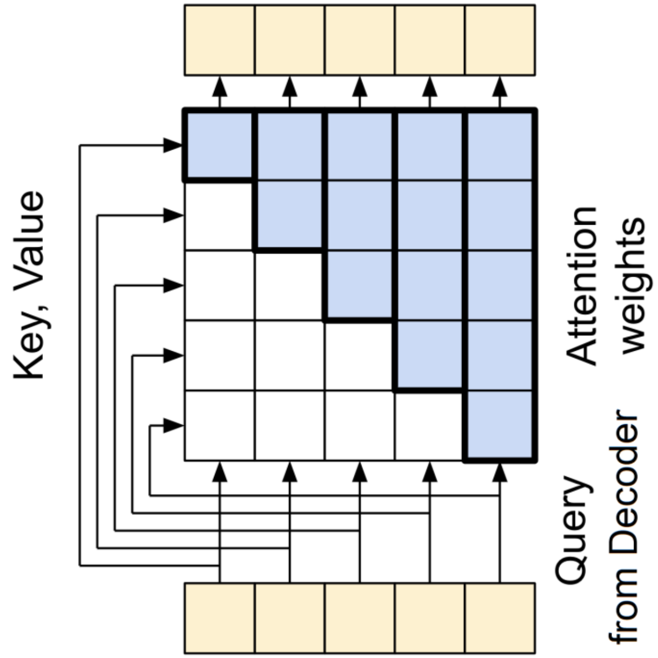


Decoder-Only Models: GPT

Rip away encoders

- Just stack decoders
- Use causal masking! NB: not a *mask token* like in BERT





PyLessons

Thank You!