Transformers, LLMs, pretraining, instruction-tuning

Reasoning in AI, week 3

Brief overview of LLM training

Next Token Prediction

Note: All equations in this course at best true up to transposes of matrices or permuting axes of tensors

Input: ~~Tokens $x_1, x_2, \ldots, x_n \in T$ (e.g. $T = \{1, \ldots 128{,}000\}$)~~

Vectors $\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_n \in \mathbb{R}^d$

Output: ~~Token x_{n+1}~~ Vector $\vec{x}_{n+1} \in \mathbb{R}^d$

t –th standard basis vector

$$t \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

$d \times |T|$ embedding matrix E – embedding of token t is Ee_t

$|T| \times d$ unembedding matrix U If $U\vec{y} = e_t$ then $\vec{y}$ is prediction that $\Pr[x_{n+1} = t] = 1$

A vector $\vec{y} \in \mathbb{R}^d$ defines distribution $D_{\vec{y}}$ over T with $\Pr[D_{\vec{y}} = t] \propto \exp(\tau^{-1}(U\vec{y})_t)$

τ=temperature

$\tau = \infty$: distribution is uniform

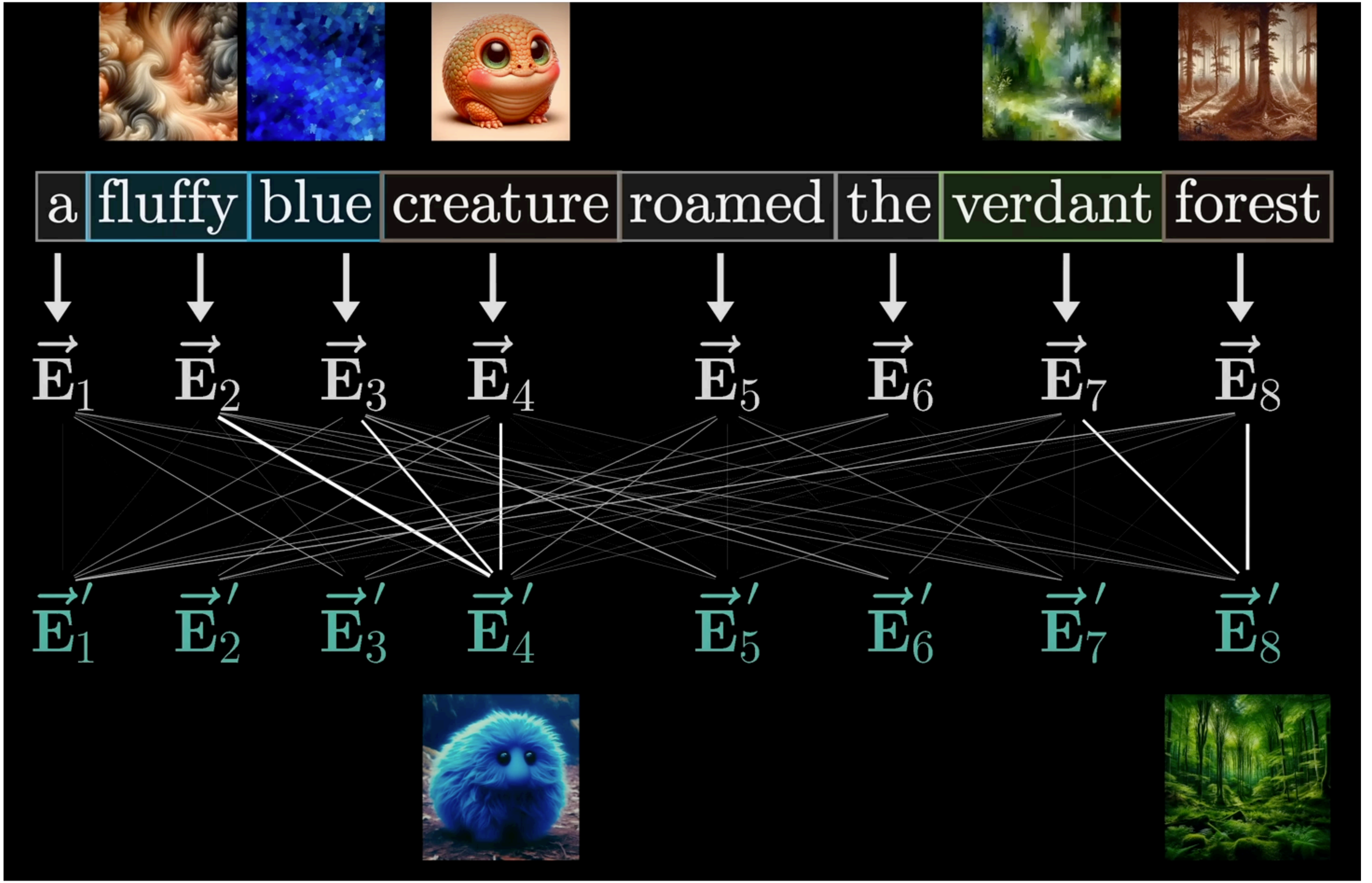
$\tau = 0$: all weight on $\arg\max_t (U\vec{y})_t$

Architecture for Next Token Prediction: Map from $\mathbb{R}^{dn}$ to $\mathbb{R}^d$

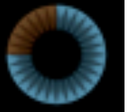
History

- Previous models for text generation: RNN, LSTM, GRU
- Common issues:
- - Temporal Bottleneck: Vanishing gradients stop many RNN architectures from learning long-range dependencies
- - Parallelisation Bottleneck: RNN states depend on previous time step hidden state, so must be computed in series

Transformer


a fluffy blue creature roamed the verdant forest
$\vec{E}_1$ $\vec{E}_2$ $\vec{E}_3$ $\vec{E}_4$ $\vec{E}_5$ $\vec{E}_6$ $\vec{E}_7$ $\vec{E}_8$
$\vec{E}'_1$ $\vec{E}'_2$ $\vec{E}'_3$ $\vec{E}'_4$ $\vec{E}'_5$ $\vec{E}'_6$ $\vec{E}'_7$ $\vec{E}'_8$

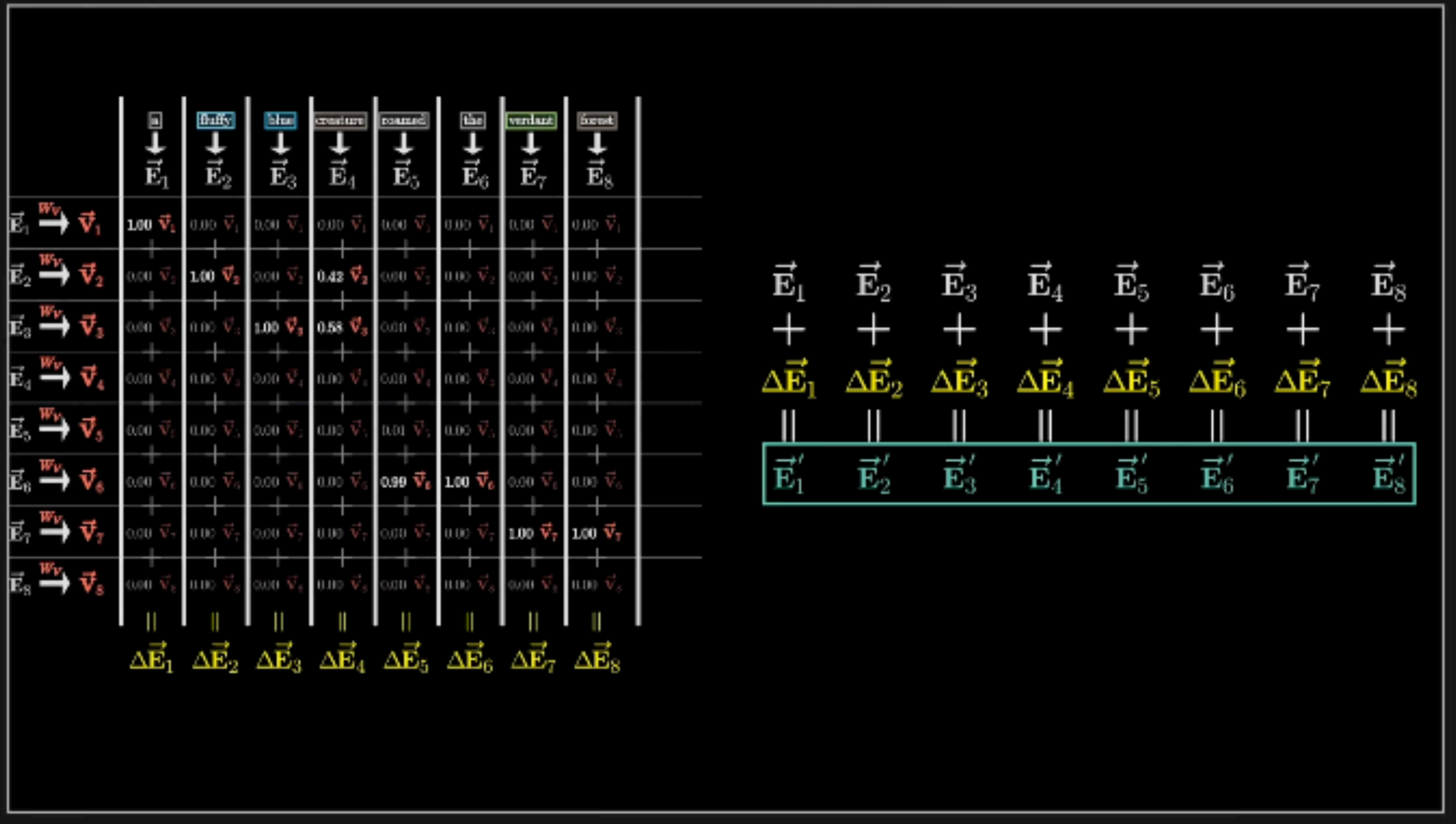
	a $\vec{E}_1 \xrightarrow{W_Q} \vec{Q}_1$	fluffy $\vec{E}_2 \xrightarrow{W_Q} \vec{Q}_2$	blue $\vec{E}_3 \xrightarrow{W_Q} \vec{Q}_3$	creature $\vec{E}_4 \xrightarrow{W_Q} \vec{Q}_4$	roamed $\vec{E}_5 \xrightarrow{W_Q} \vec{Q}_5$	the $\vec{E}_6 \xrightarrow{W_Q} \vec{Q}_6$	verdant $\vec{E}_7 \xrightarrow{W_Q} \vec{Q}_7$	forest $\vec{E}_8 \xrightarrow{W_Q} \vec{Q}_8$
a $\vec{E}_1 \xrightarrow{W_K} \vec{K}_1$	$\vec{K}_1 \cdot \vec{Q}_1$	$\vec{K}_1 \cdot \vec{Q}_2$	$\vec{K}_1 \cdot \vec{Q}_3$	$\vec{K}_1 \cdot \vec{Q}_4$	$\vec{K}_1 \cdot \vec{Q}_5$	$\vec{K}_1 \cdot \vec{Q}_6$	$\vec{K}_1 \cdot \vec{Q}_7$	$\vec{K}_1 \cdot \vec{Q}_8$
fluffy $\vec{E}_2 \xrightarrow{W_K} \vec{K}_2$	$\vec{K}_2 \cdot \vec{Q}_1$	$\vec{K}_2 \cdot \vec{Q}_2$	$\vec{K}_2 \cdot \vec{Q}_3$	$\vec{K}_2 \cdot \vec{Q}_4$	$\vec{K}_2 \cdot \vec{Q}_5$	$\vec{K}_2 \cdot \vec{Q}_6$	$\vec{K}_2 \cdot \vec{Q}_7$	$\vec{K}_2 \cdot \vec{Q}_8$
blue $\vec{E}_3 \xrightarrow{W_K} \vec{K}_3$	$\vec{K}_3 \cdot \vec{Q}_1$	$\vec{K}_3 \cdot \vec{Q}_2$	$\vec{K}_3 \cdot \vec{Q}_3$	$\vec{K}_3 \cdot \vec{Q}_4$	$\vec{K}_3 \cdot \vec{Q}_5$	$\vec{K}_3 \cdot \vec{Q}_6$	$\vec{K}_3 \cdot \vec{Q}_7$	$\vec{K}_3 \cdot \vec{Q}_8$
creature $\vec{E}_4 \xrightarrow{W_K} \vec{K}_4$	$\vec{K}_4 \cdot \vec{Q}_1$	$\vec{K}_4 \cdot \vec{Q}_2$	$\vec{K}_4 \cdot \vec{Q}_3$	$\vec{K}_4 \cdot \vec{Q}_4$	$\vec{K}_4 \cdot \vec{Q}_5$	$\vec{K}_4 \cdot \vec{Q}_6$	$\vec{K}_4 \cdot \vec{Q}_7$	$\vec{K}_4 \cdot \vec{Q}_8$
roamed $\vec{E}_5 \xrightarrow{W_K} \vec{K}_5$	$\vec{K}_5 \cdot \vec{Q}_1$	$\vec{K}_5 \cdot \vec{Q}_2$	$\vec{K}_5 \cdot \vec{Q}_3$	$\vec{K}_5 \cdot \vec{Q}_4$	$\vec{K}_5 \cdot \vec{Q}_5$	$\vec{K}_5 \cdot \vec{Q}_6$	$\vec{K}_5 \cdot \vec{Q}_7$	$\vec{K}_5 \cdot \vec{Q}_8$
the $\vec{E}_6 \xrightarrow{W_K} \vec{K}_6$	$\vec{K}_6 \cdot \vec{Q}_1$	$\vec{K}_6 \cdot \vec{Q}_2$	$\vec{K}_6 \cdot \vec{Q}_3$	$\vec{K}_6 \cdot \vec{Q}_4$	$\vec{K}_6 \cdot \vec{Q}_5$	$\vec{K}_6 \cdot \vec{Q}_6$	$\vec{K}_6 \cdot \vec{Q}_7$	$\vec{K}_6 \cdot \vec{Q}_8$
verdant $\vec{E}_7 \xrightarrow{W_K} \vec{K}_7$	$\vec{K}_7 \cdot \vec{Q}_1$	$\vec{K}_7 \cdot \vec{Q}_2$	$\vec{K}_7 \cdot \vec{Q}_3$	$\vec{K}_7 \cdot \vec{Q}_4$	$\vec{K}_7 \cdot \vec{Q}_5$	$\vec{K}_7 \cdot \vec{Q}_6$	$\vec{K}_7 \cdot \vec{Q}_7$	$\vec{K}_7 \cdot \vec{Q}_8$
forest $\vec{E}_8 \xrightarrow{W_K} \vec{K}_8$	$\vec{K}_8 \cdot \vec{Q}_1$	$\vec{K}_8 \cdot \vec{Q}_2$	$\vec{K}_8 \cdot \vec{Q}_3$	$\vec{K}_8 \cdot \vec{Q}_4$	$\vec{K}_8 \cdot \vec{Q}_5$	$\vec{K}_8 \cdot \vec{Q}_6$	$\vec{K}_8 \cdot \vec{Q}_7$	$\vec{K}_8 \cdot \vec{Q}_8$



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


a
fluffy
blue
creature
roamed
the
verdant
forest
$\vec{E}_1$ $\vec{E}_2$ $\vec{E}_3$ $\vec{E}_4$ $\vec{E}_5$ $\vec{E}_6$ $\vec{E}_7$ $\vec{E}_8$
a → $\vec{E}_1$ W_V → $\vec{V}_1$
fluffy → $\vec{E}_2$ W_V → $\vec{V}_2$
blue → $\vec{E}_3$ W_V → $\vec{V}_3$
creature → $\vec{E}_4$ W_V → $\vec{V}_4$
roamed → $\vec{E}_5$ W_V → $\vec{V}_5$
the → $\vec{E}_6$ W_V → $\vec{V}_6$
verdant → $\vec{E}_7$ W_V → $\vec{V}_7$
forest → $\vec{E}_8$ W_V → $\vec{V}_8$
0.00 $\vec{V}_1$
+
0.42 $\vec{V}_2$
+
0.58 $\vec{V}_3$
+
0.00 $\vec{V}_4$
+
0.00 $\vec{V}_5$
+
0.00 $\vec{V}_6$
+
0.00 $\vec{V}_7$
+
0.00 $\vec{V}_8$
||
$\Delta\vec{E}_4$
creature
$\vec{E}_4$
+
$\Delta\vec{E}_4$
||
$\vec{E}_4'$

One head of attention



One head attention

$x_1, x_2, \ldots, x_n \in R^{d_{in}}$ - token embeddings

embedding matrix: $X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \in R^{n \times d_{in}}$

d_k - dim of key & query vectors

$d_v = d_{in}$ - dim of value vectors

$W_Q, W_K \in R^{d_{in} \times d_k}, W_V \in R^{d_{in} \times d_v}$ - query, key and value transform matrices

Query, key, and value matrices:

$$Q = XW_Q \in R^{n \times d_k} \qquad K = XW_K \in R^{n \times d_k} \qquad V = XW_V \in R^{n \times d_v}$$

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

One head attention

$x_1, x_2, \ldots, x_n \in R^{d_{in}}$ - token embeddings

embedding matrix: $X = \begin{bmatrix} x_1^T \\ \vdots \\ x_n^T \end{bmatrix} \in R^{n \times d_{in}}$

d_k - dim of key & query vectors $d_v = d_{in}$ - dim of value vectors

$W_Q, W_K \in R^{d_{in} \times d_k}, W_V \in R^{d_{in} \times d_v}$ - query, key and value transform matrices

Query, key, and value matrices:

$$Q = XW_Q \in R^{n \times d_k} \quad K = XW_K \in R^{n \times d_k} \quad V = XW_V \in R^{n \times d_v}$$

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

Time complexity: $O(nd_{in}^2 + n^2 d_{in})$

Multihead attention

$$W_i^Q, W_i^K \in R^{d_{in} \times d_k}, W_i^V \in R^{d_{in} \times d_v}, i \in [h] \qquad W^O \in R^{h d_{in} \times d_v}$$

$$head_i = Attention(XW_i^Q, XW_i^K, XW_i^V)$$

$$MultiHead(\{W_i^Q, W_i^K, W_i^V\}_{i=1}^N) = Concat(head_1, \ldots, head_h)W^O$$

Multihead attention

$$W_i^Q, W_i^K \in R^{d_{in} \times d_k}, W_i^V \in R^{d_{in} \times d_v}, i \in [h] \qquad W^O \in R^{hd_{in} \times d_v}$$

$$head_i = Attention(XW_i^Q, XW_i^K, XW_i^V)$$

$$MultiHead(\{W_i^Q, W_i^K, W_i^V\}_{i=1}^N) = Concat(head_1, \ldots, head_h)W^O$$

Good practice: $d_k = d_v = \frac{d_{in}}{h} \Rightarrow$ Time complexity: $O(nd_{in}^2 + n^2 d_{in})$

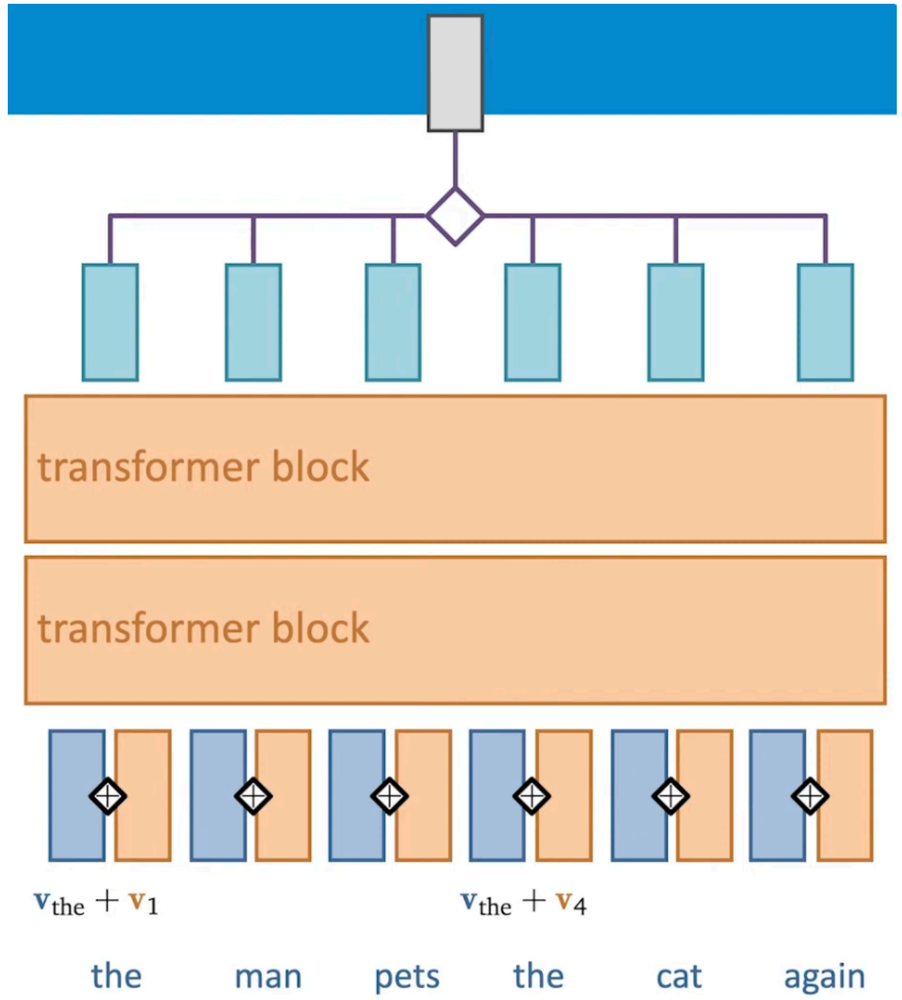
Position embeddings

word embeddings:

$\mathbf{v}_{\text{the}}, \mathbf{v}_{\text{man}}, \mathbf{v}_{\text{pets}}, \mathbf{v}_{\text{cat}}, \mathbf{v}_{\text{again}}$

position embeddings:

$\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_5, \ldots$



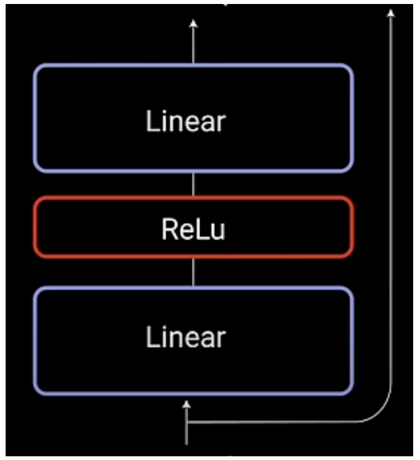
Transformer architecture

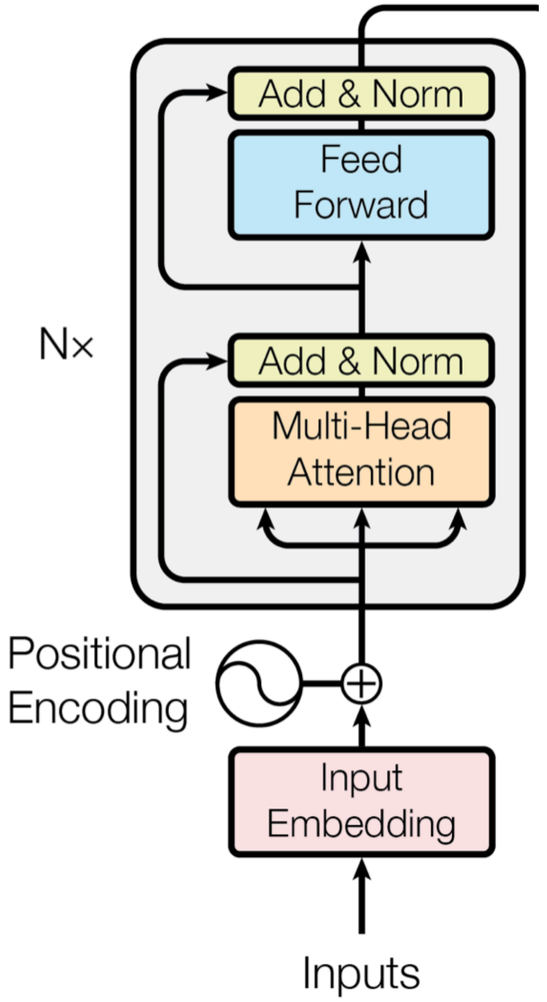
- **Layer Normalisation**
  - Normalize the outputs of different modules

$$y = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$

- **Residual Connections**
  - Add the input of a module to its output
  - $\mathrm{LayerNorm}(x + \mathrm{Sublayer}(x))$

Feed-Forward





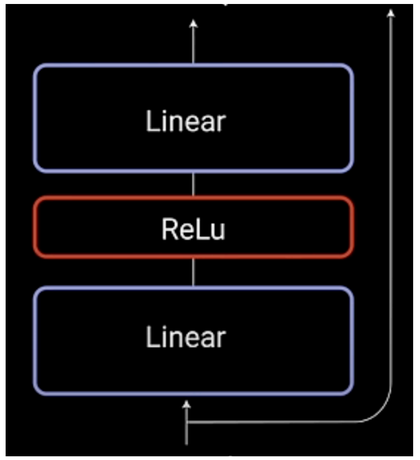
Transformer architecture

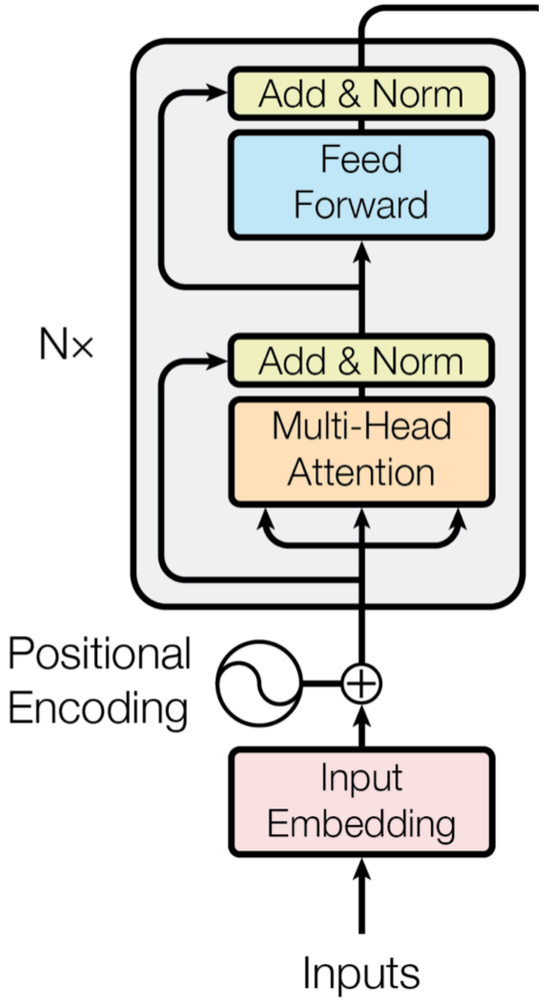
- **Layer Normalisation**
  - Normalize the outputs of different modules

$$y = \frac{x - \mathrm{E}[x]}{\sqrt{\mathrm{Var}[x] + \epsilon}} * \gamma + \beta$$

- **Residual Connections**
  - Add the input of a module to its output
  - $\mathrm{LayerNorm}(x + \mathrm{Sublayer}(x))$

Feed-Forward

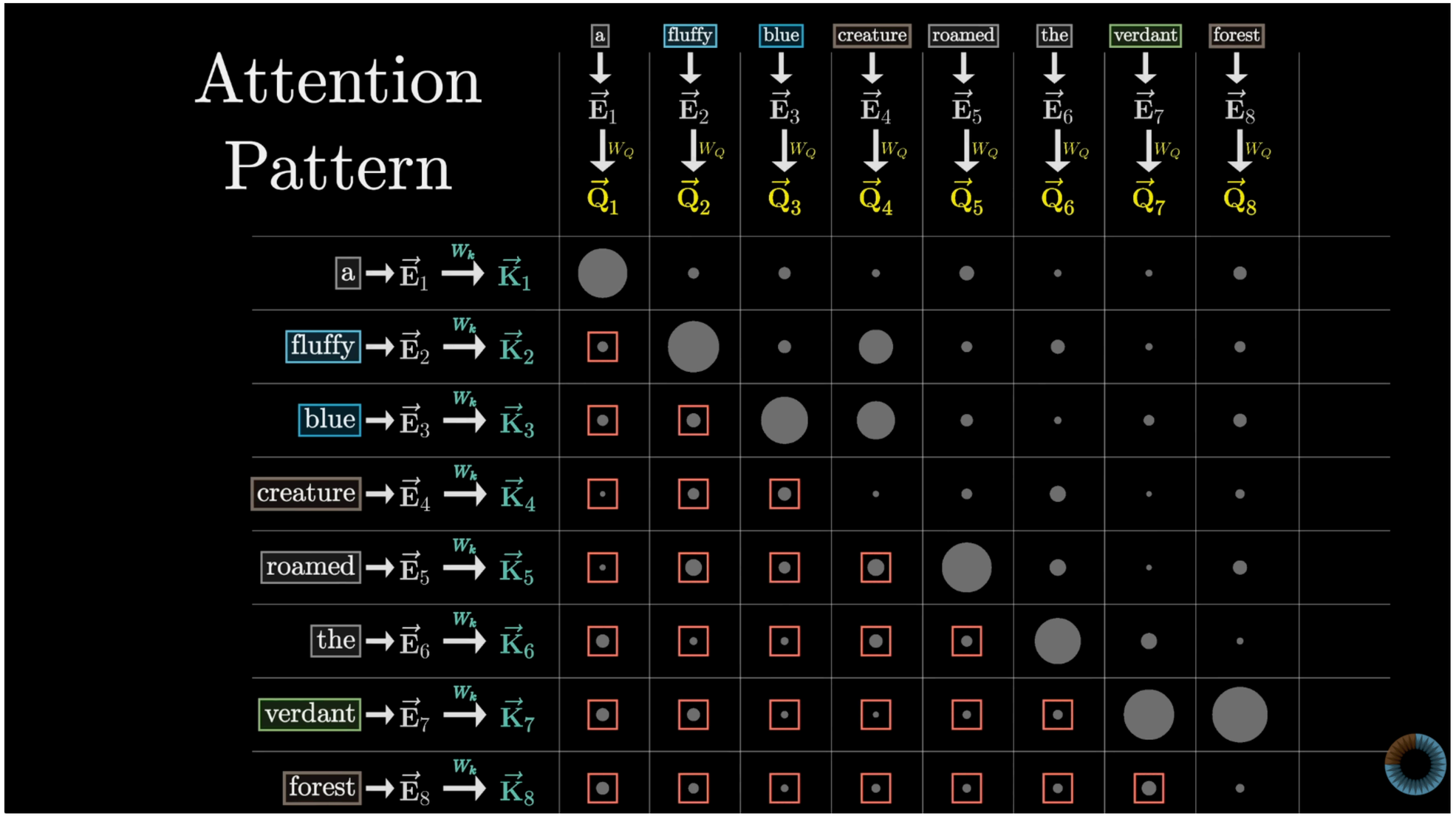




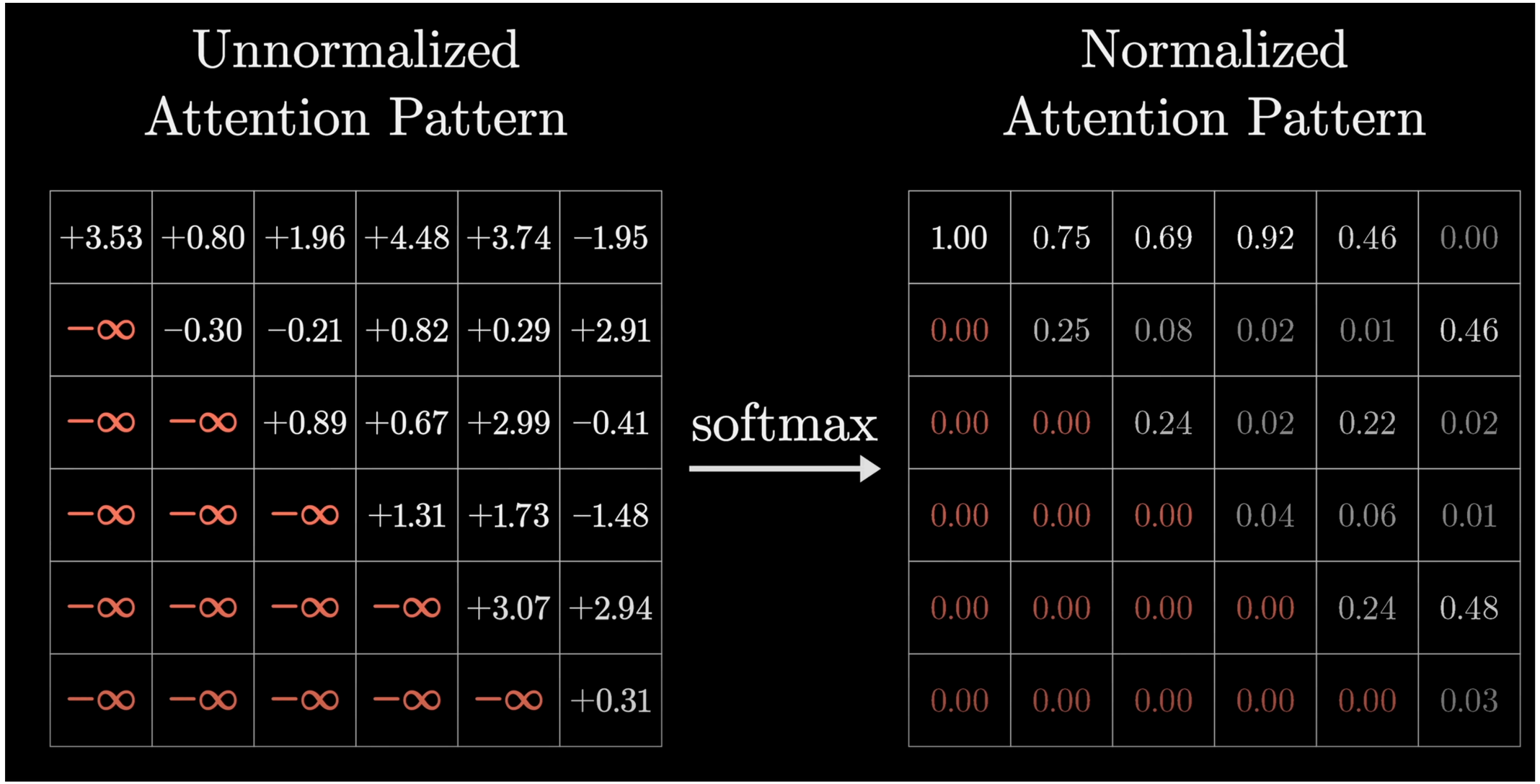
Feed-Forward layer is applied to tokens pointwise.

Tokens only interact in Attention layers!

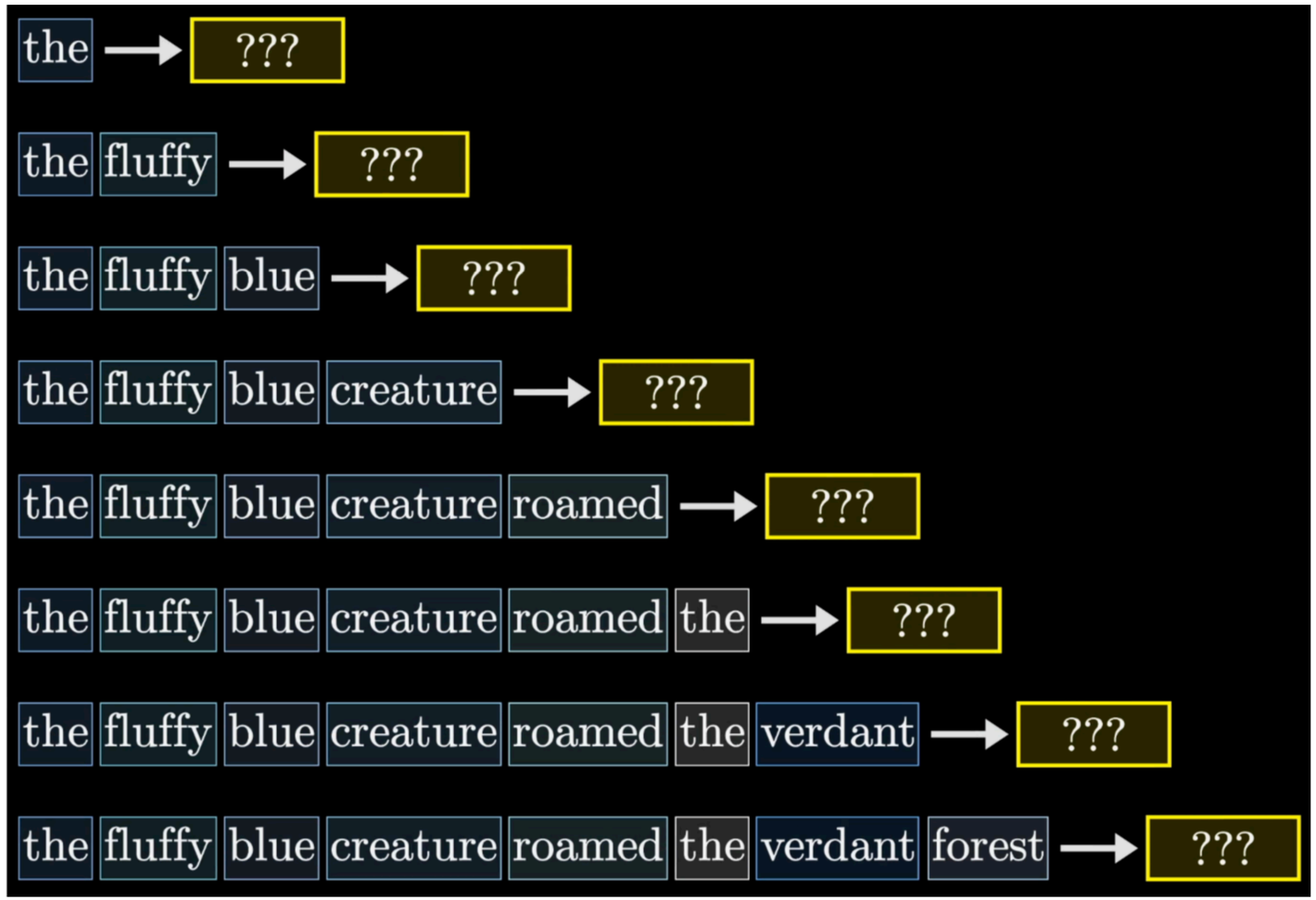
Attention Masking



Attention Masking



Transformer for next token prediction

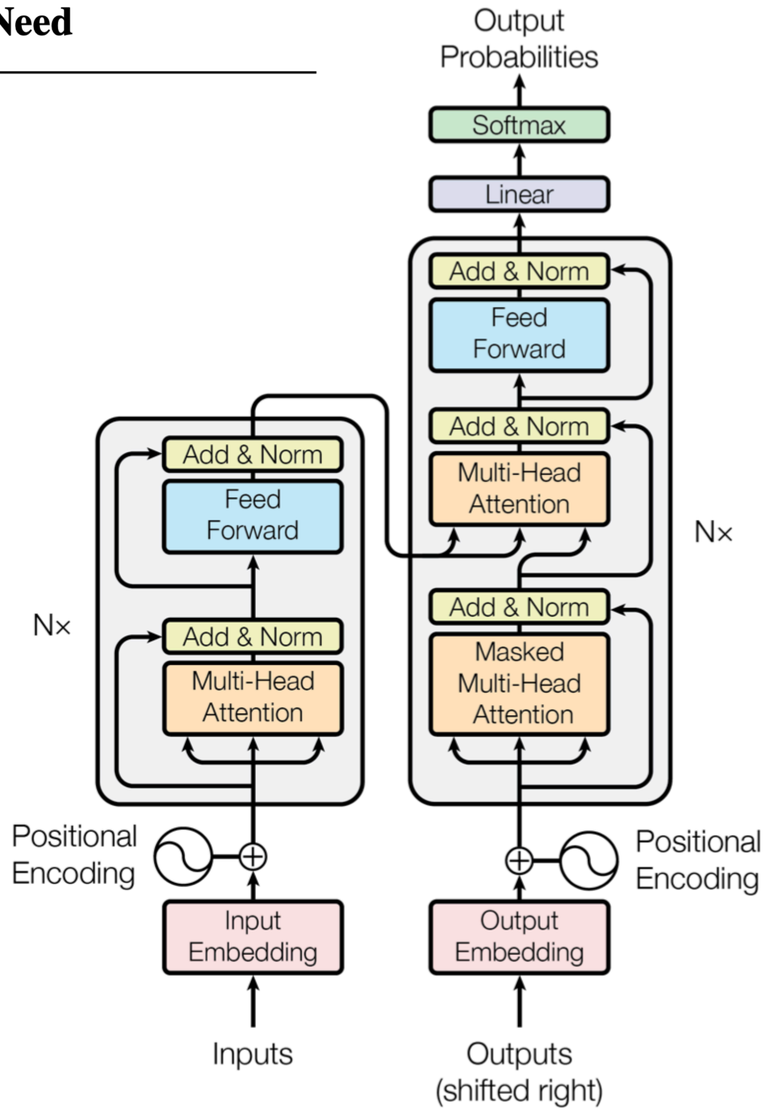


Terminology

- Attention without masking = bidirectional self-attention
- Attention with masking = autoregressive self-attention

Transformer: original encoder-decoder architecture (obsolete)

Attention Is All You Need



Types of Transformers

- Encoder-only Transformer (uses bidirectional self-attention)
- Decoder-only Transformer (uses autoregressive self-attention)
- Encoder-decoder Transformer (obsolete; uses bidirectional self-attention in encoder part, autoregressive self-attention in decoder part, and bidirectional cross-attention between them)

Transformers: summary

Pros:

- Attention mechanism: any token can attend to any other => good at long-term dependancies
- Architecture is designed to be highly parallelizable
- Universality: no inductive bias in architecture

Cons:

- Bottleneck: $O(n^2)$ computation and memory complexity for each attention
- More efficient attention mechanisms were proposed (active area of research)
- Requires a lot of data for training

LLMs

Language Models are Unsupervised Multitask Learners (GPT-2, 1.5B, 2019)

- Generative pre-training only
- Evaluating model by:
- - max-likelihood prediction (predict inserted word, MCQ)
- - simple prompting (e.g. TL;DR for summarisation)
- - few-shot learning (translation, question-answering dataset)
- Was pretty good on French-English translation, even though trained only on English corpora

---

”I’m not the cleverest man in the world, but like they say in French: **Je ne suis pas un imbecile [I’m not a fool].**

In a now-deleted post from Aug. 16, Soheil Eid, Tory candidate in the riding of Joliette, wrote in French: ”**Mentez mentez, il en restera toujours quelque chose**,” which translates as, ”**Lie lie and something will always remain.**”

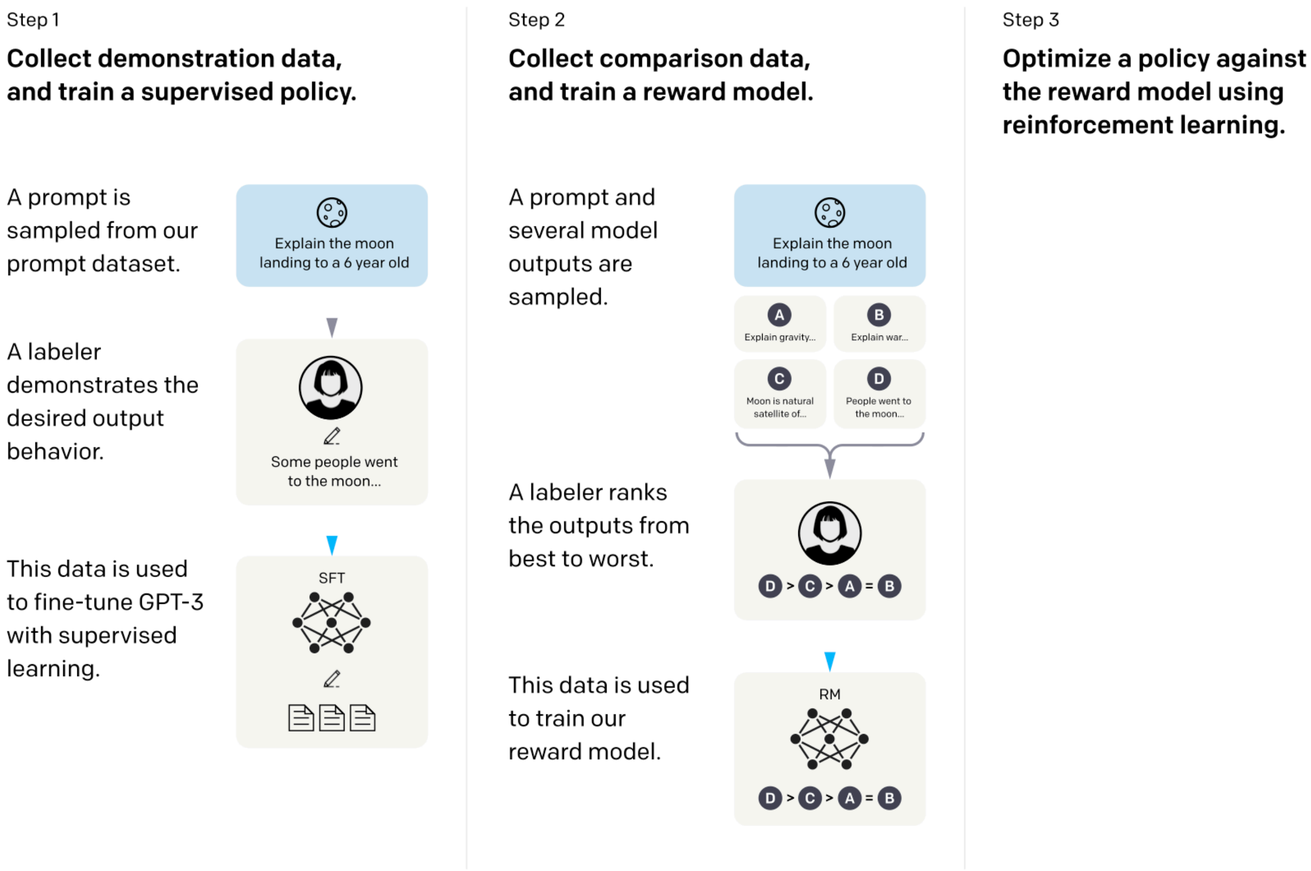
“I hate the word ‘**perfume**,”’ Burr says. ‘It’s somewhat better in French: ‘**parfum**.’

If listened carefully at 29:55, a conversation can be heard between two guys in French: “**-Comment on fait pour aller de l’autre coté? -Quel autre coté?**”, which means “**- How do you get to the other side? - What side?**”.

If this sounds like a bit of a stretch, consider this question in French: **As-tu aller au cinéma?**, or **Did you go to the movies?**, which literally translates as Have-you to go to movies/theater?

“**Brevet Sans Garantie Du Gouvernement**”, translated to English: “**Patented without government warranty**”.

Instruction tuning, RLHF

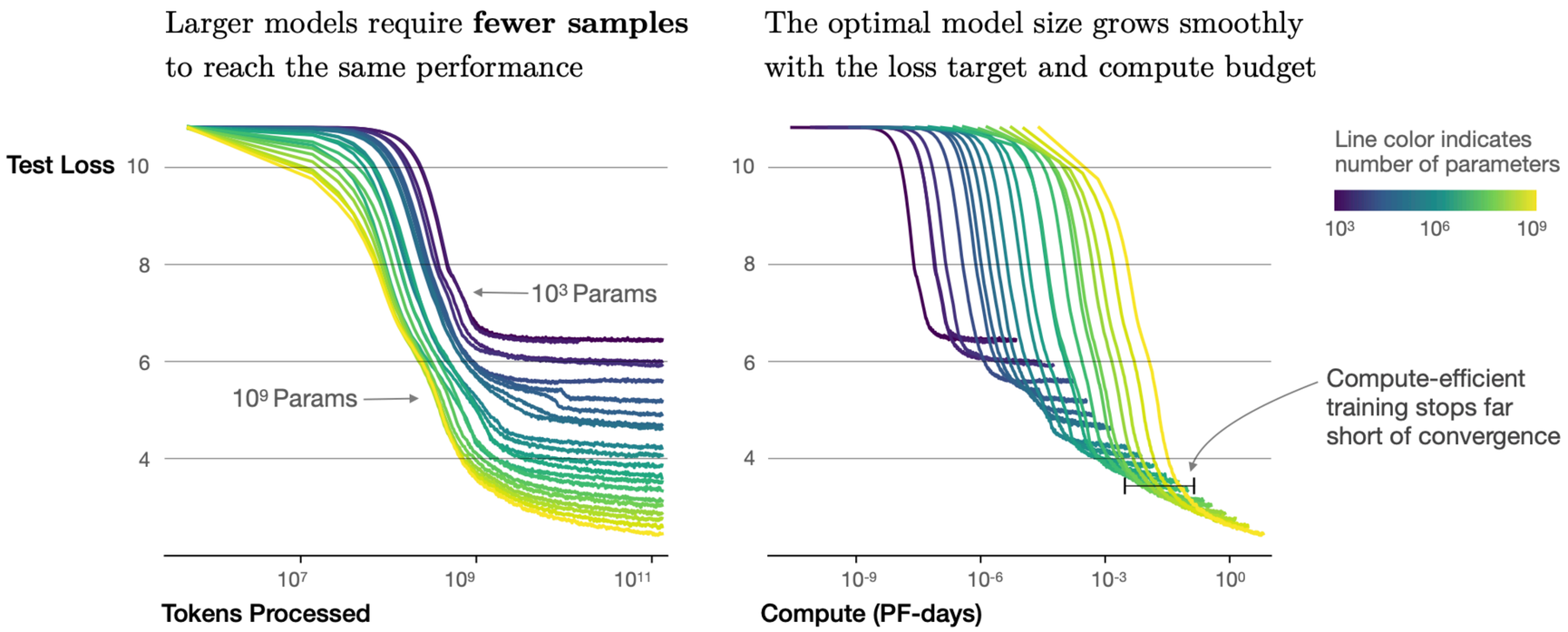


Training language models to follow instructions with human feedback (GPT-Instruct, 2022)

Instruction tuning

- Forgetting: applying instruction tuning naively can reduce the model performance on standard NL benchmarks
- Remedy: mixing gradients from instruction tuning objective with the gradients from the pre-training task objective

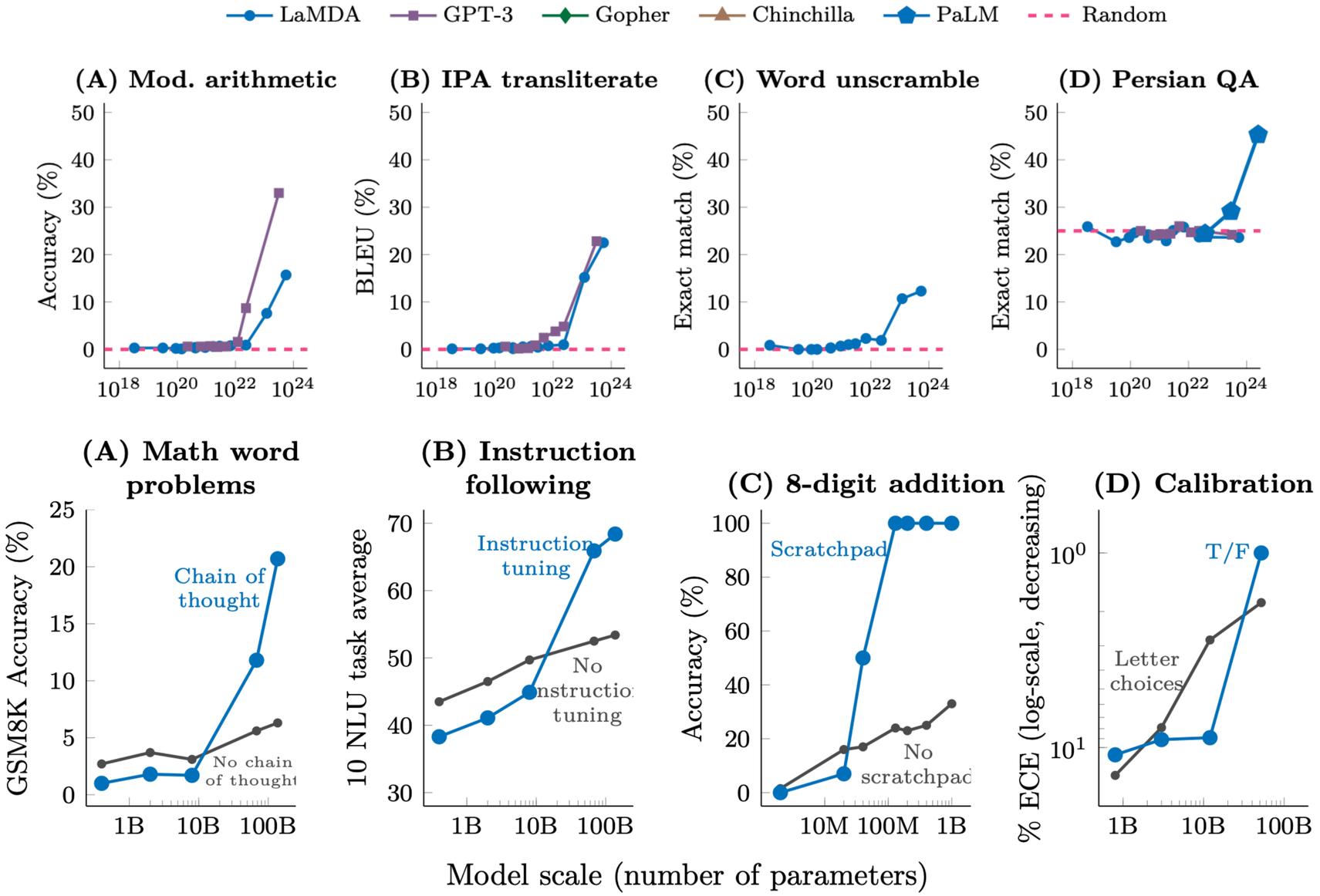
Scaling Laws for LLMs



- When working within a fixed compute budget, we obtain the optimal performance by training very large models and stopping significantly short of convergence

from “Scaling Laws for Neural Language Models” (Kaplan et. al., 2020)

Emergent abilities of LLMs



Emergent Abilities of Large Language Models (Wei et. al., 2022)

Thank you!

Sources:

- 3Blue1Brown
- Course by Boaz Barak
- DLVU