Attention

Gautam Kamath

Sequence Modelling

- Suppose we have a sequence of length n, each element in the sequence is of dimension d
- Previous solution: RNNs
  - Computing each hidden state is $O(d^2)$, so overall $O(nd^2)$ computation
  - Long chains make it hard to deal with long-range dependencies
  - Optimization woes like vanishing and exploding gradients
  - Many training steps
  - Sequential nature makes it hard to parallelize
- The attention mechanism solves most of these
  - …though computation increases for long sequences

Attention Mechanism/Layer

- Takes three inputs: set of queries q_j, keys k_i, and values v_i
  - Same number of keys and values, number of queries may differ
- For a given query q_j, tries to mimic retrieval lookup of value v_i corresponding to key k_i which "matches" query
- $\text{attention}\left(q, \vec{k}, \vec{v}\right) = \sum_i \text{similarity}(q, k_i) \times v_i$
- (Draw: layer 1 is q and $k_i \rightarrow s_i$, layer 2 is softmax to get a_i's, layer 3 is multiplication and sum with a_i and v_i to produce output)

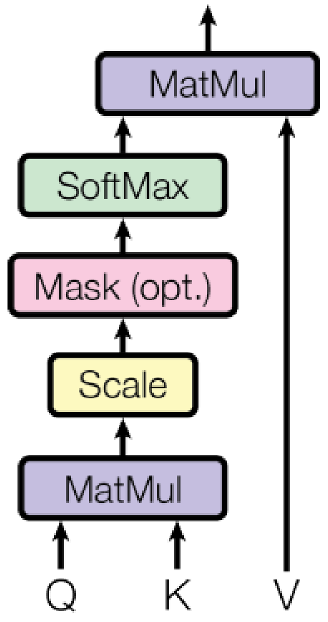
Similarity?

- What is $\mathrm{similarity}(q, k)$?
- Dot product: $\langle q, k \rangle$
- Scaled dot product: $\frac{\langle q,k \rangle}{\sqrt{d}}$
  - q and k are of dimension d
- General dot product: $\langle Aq, Bk \rangle$
  - A and B are matrices of learnable parameters

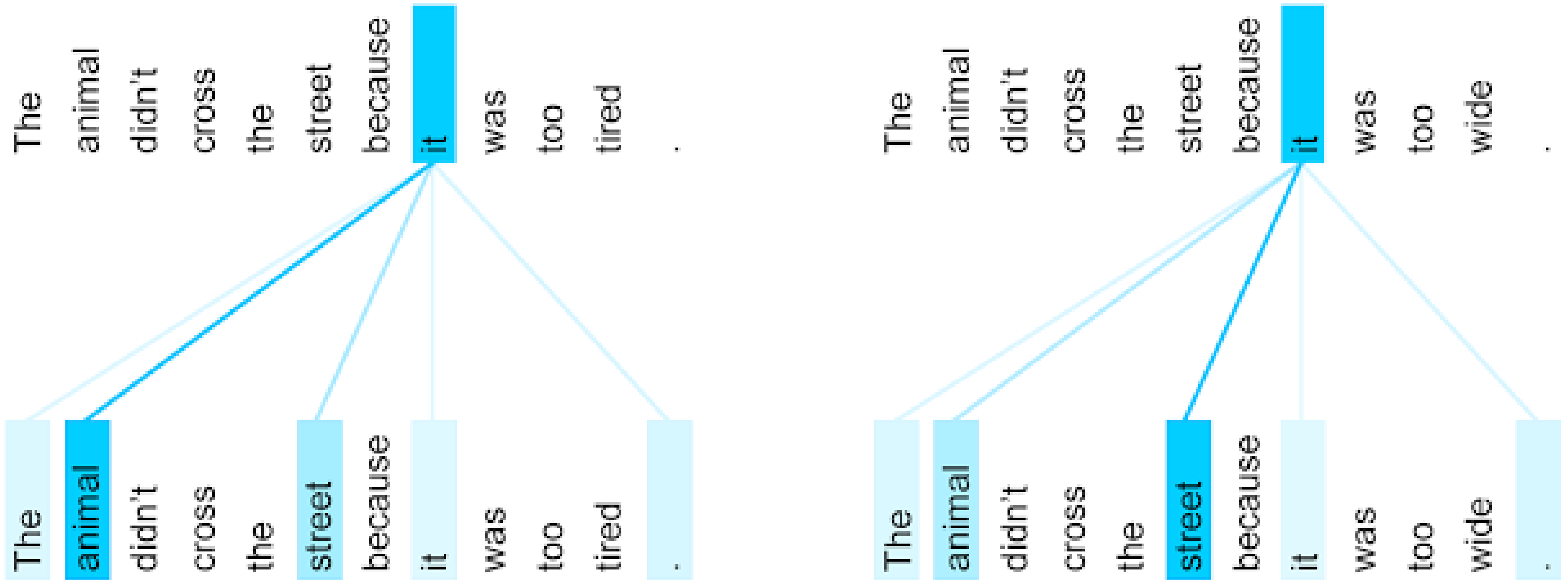
Attention for Sets

- Since similarity between vectors q and k is $\langle q, k \rangle$, how do we compute similarity between *sets* of vectors Q and K?
- Matrix multiplication: QK^T
- Attention mechanism looks like →
- Self-attention: when $Q = K = V$
- (Draw example with "word" matrices attn)
- $\mathrm{softmax}(VV^T)$: each row becomes distribution
- $\mathrm{softmax}(VV^T)V$: replace rows with weighted sums

Scaled Dot-Product Attention



Visualizing Attention

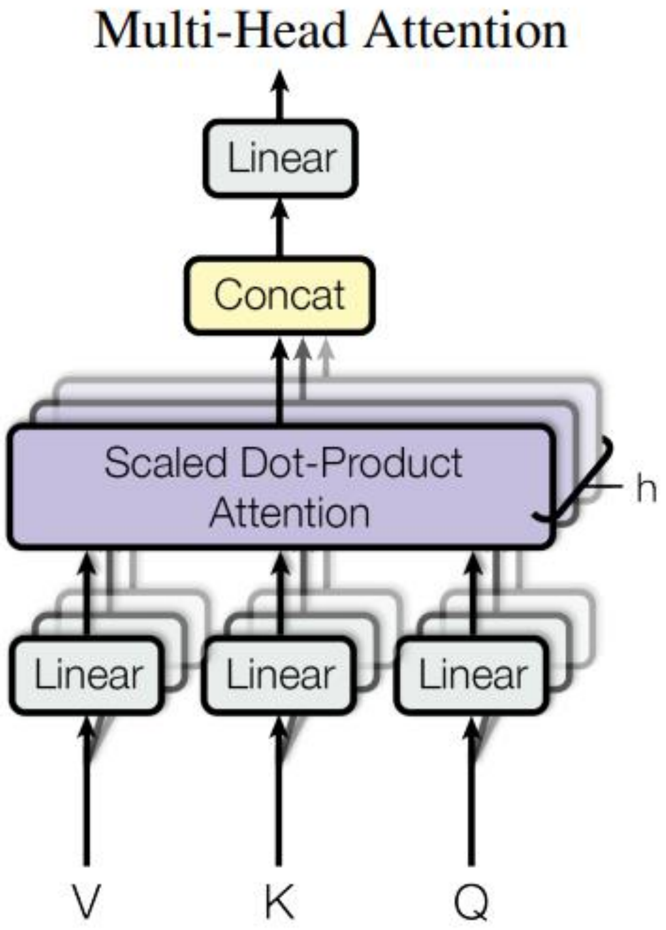


Attention vs RNN for sequences

- Sequence of n vectors, each d-dimensional
- Computation: $O(n^2 d)$
  - VV^T computation is $O(n^2 d)$
- Compare with RNNs: $O(nd^2)$
- Advantage of attention: maximum sequence length is $O(1)$
  - No long dependence chains for long sequences

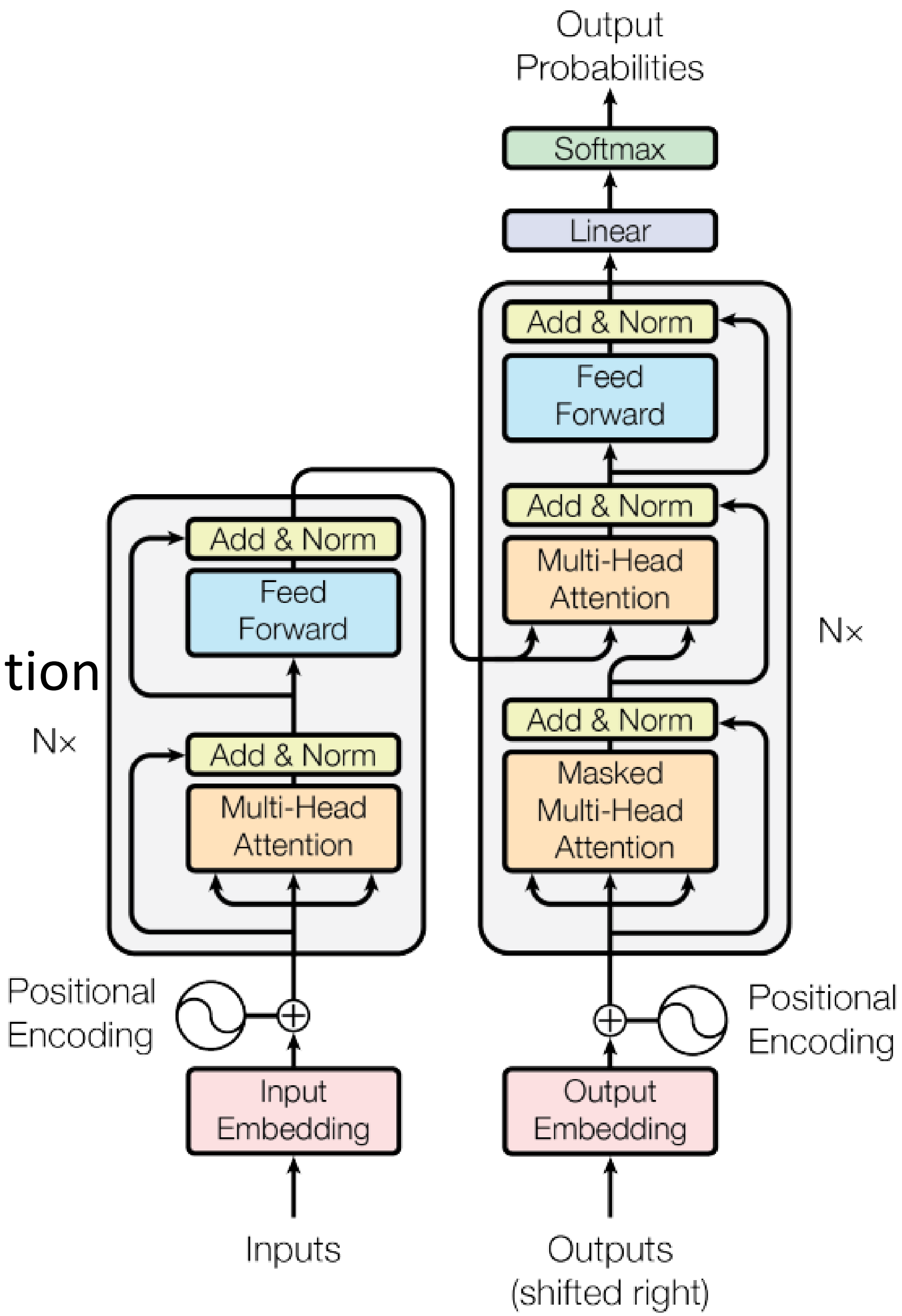
Multi-head attention

- Precede Attention with linear layer
  - General dot product
- Multiple “heads” in parallel
  - Similar to multiple filters in CNNs



Transformer Architecture

- Encoder-decoder structure
- Other layers:
- Input embedding
    - Convert one-hot vectors to a denser representation
- Positional encoding
    - Encode position of token in sequence
        - E.g., dog is behind the cat vs cat is behind the dog
- Layer normalization



Masked (Self-)Attention

- Say you're doing next word prediction… prevent "cheating" by looking at the next word!
- Self-attention: $\mathrm{softmax}(VV^T)V$
- Masked self-attention: $\mathrm{softmax}(\mathrm{mask}(VV^T))V$
- $mask(M)_{ij} = -\infty$ if $i < j$, M_{ij} otherwise

Results

Table 2: The Transformer achieves better BLEU scores than previous state-of-the-art models on the English-to-German and English-to-French newstest2014 tests at a fraction of the training cost.

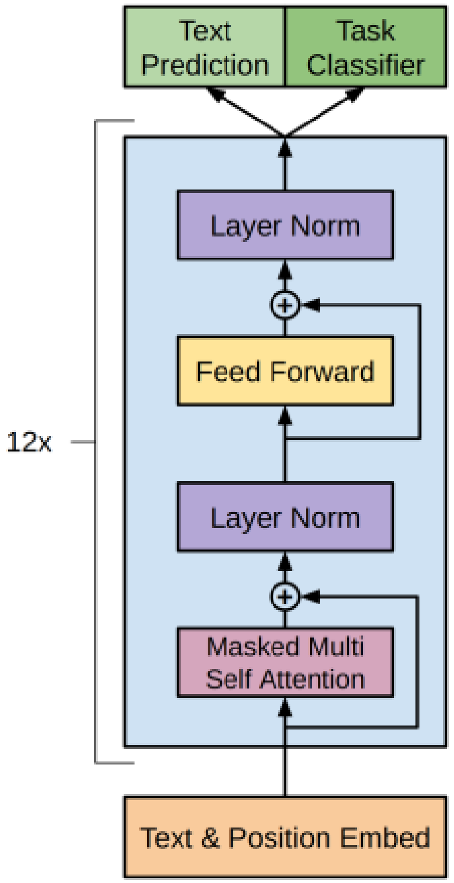
Model	BLEU		Training Cost (FLOPs)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet [18]	23.75			
Deep-Att + PosUnk [39]		39.2		$1.0 \cdot 10^{20}$
GNMT + RL [38]	24.6	39.92	$2.3 \cdot 10^{19}$	$1.4 \cdot 10^{20}$
ConvS2S [9]	25.16	40.46	$9.6 \cdot 10^{18}$	$1.5 \cdot 10^{20}$
MoE [32]	26.03	40.56	$2.0 \cdot 10^{19}$	$1.2 \cdot 10^{20}$
Deep-Att + PosUnk Ensemble [39]		40.4		$8.0 \cdot 10^{20}$
GNMT + RL Ensemble [38]	26.30	41.16	$1.8 \cdot 10^{20}$	$1.1 \cdot 10^{21}$
ConvS2S Ensemble [9]	26.36	**41.29**	$7.7 \cdot 10^{19}$	$1.2 \cdot 10^{21}$
Transformer (base model)	27.3	38.1	$\mathbf{3.3 \cdot 10^{18}}$	
Transformer (big)	**28.4**	**41.8**	$2.3 \cdot 10^{19}$	

GPT-1

- Transformer decoder arch
- Two stage training

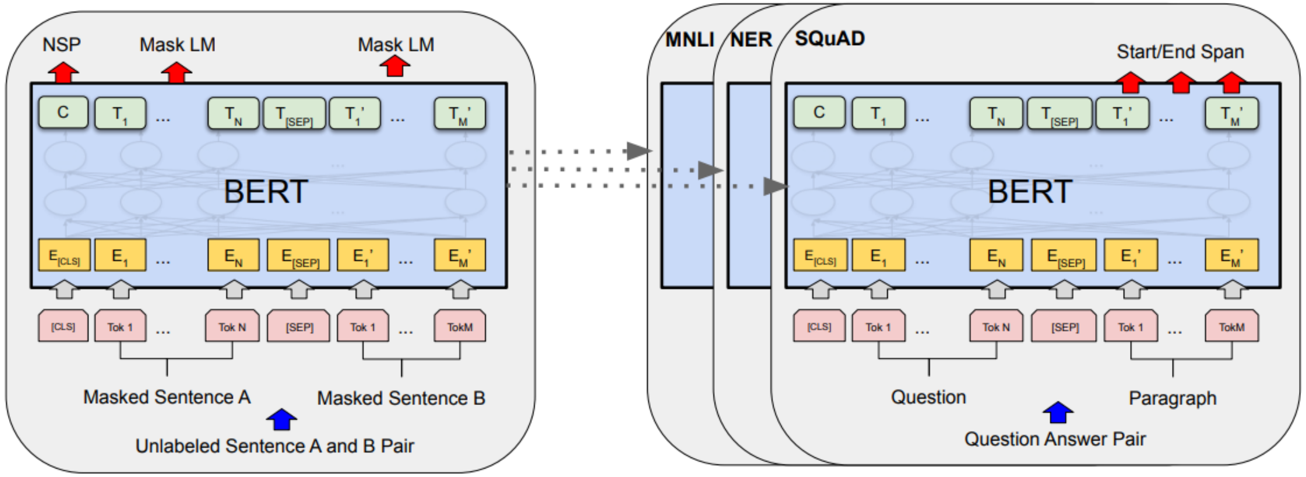
1. Unsupervised pre-training
    - Lots of (unlabeled) text used
    - Objective: next word prediction
2. Supervised fine-tuning
    - Smaller amount of labeled text

- Use a different output at end for each task
- Result of 1. can be downloaded and fine-tuned for multiple different tasks



BERT

- Transformer encoder arch
- Pre-training: masked word prediction
- Input: I took my [mask] for a walk
- Answer: dog



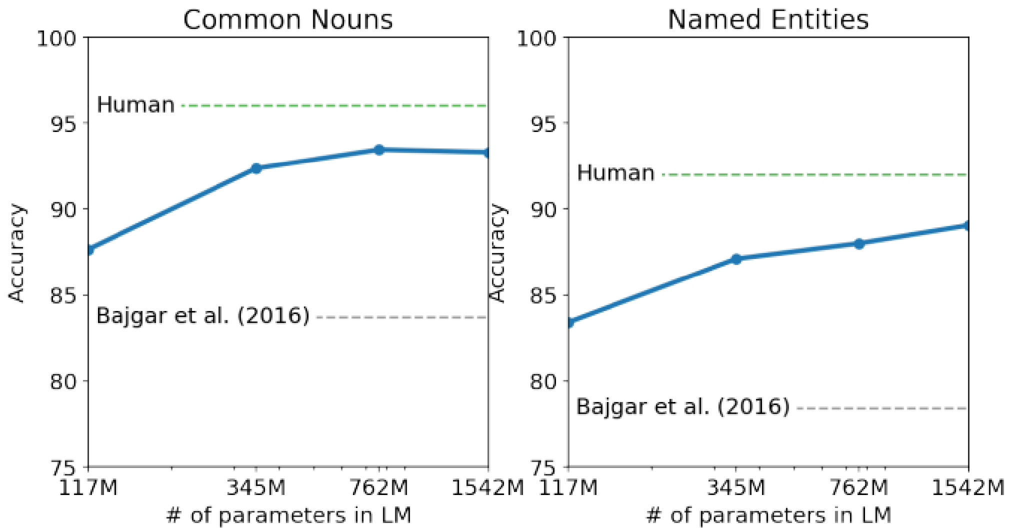
Pre-training

Fine-Tuning

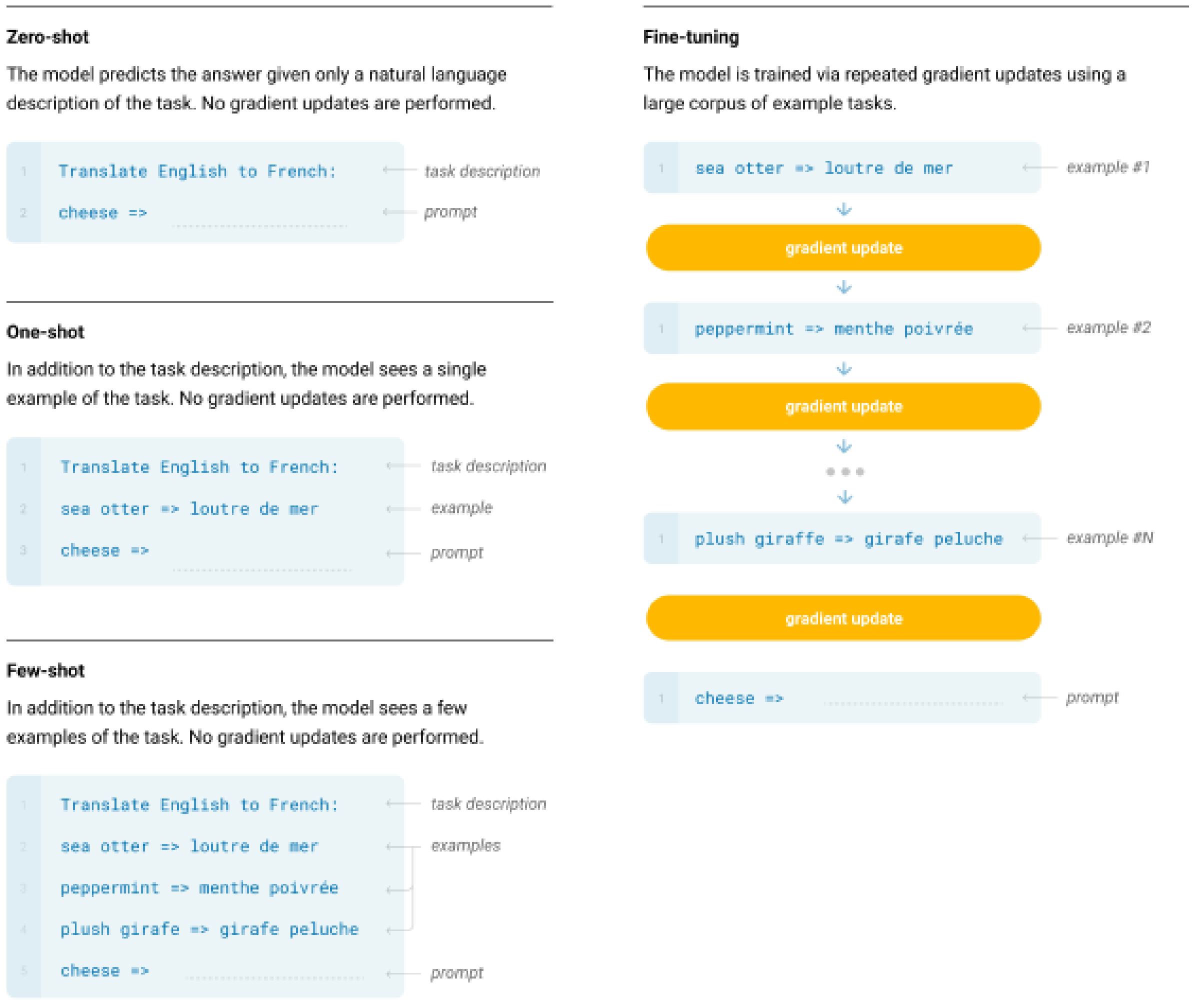
GPT-2

- 10x bigger than GPT-1
- Bigger models are more powerful

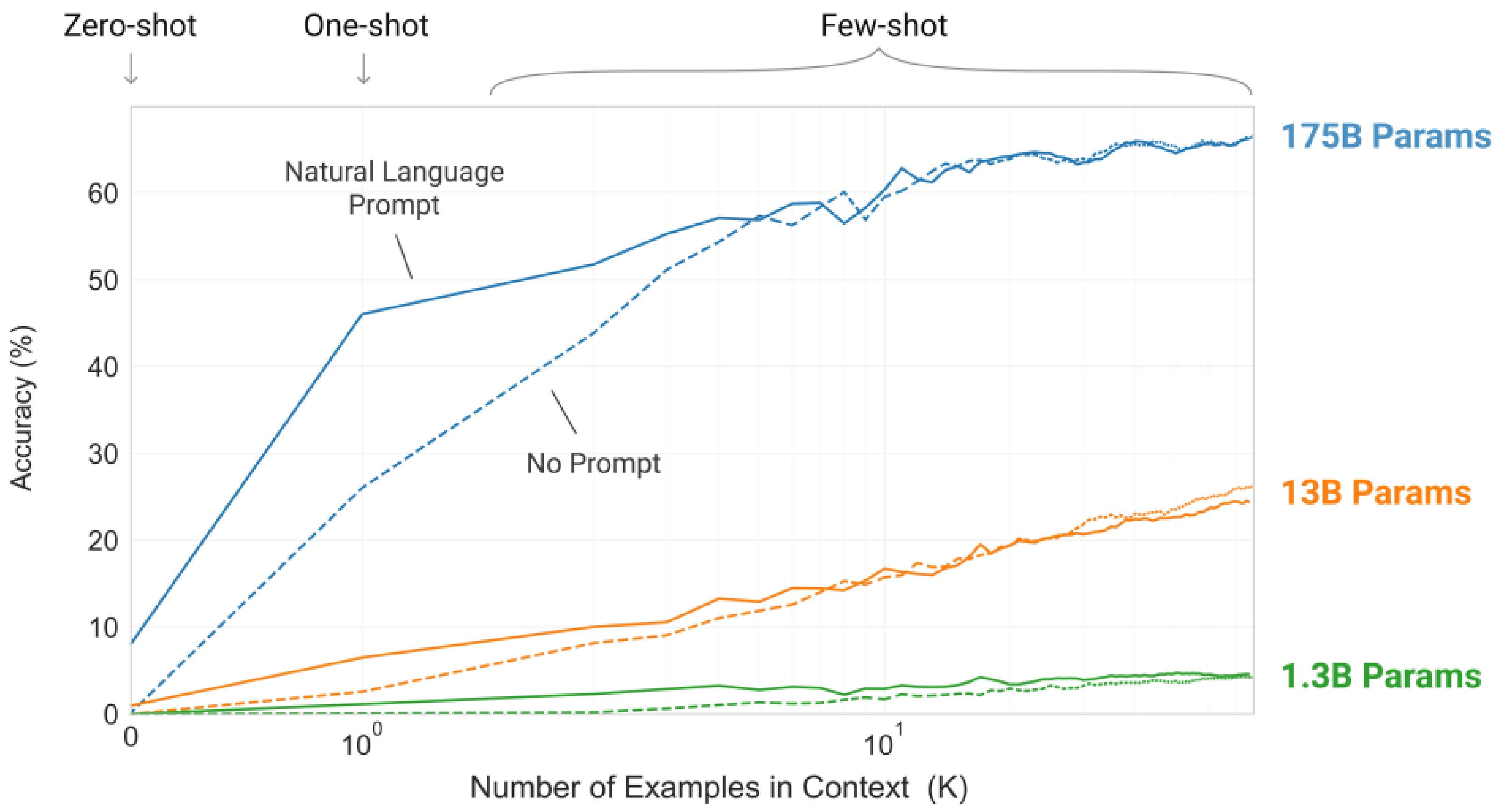
3.2. Children's Book Test



GPT-3 – even bigger



Scale Helps



Zero-shot
One-shot
Few-shot
175B Params
13B Params
1.3B Params
Natural Language Prompt
No Prompt
Accuracy (%)
0
10
20
30
40
50
60
0
10^0
10^1
Number of Examples in Context (K)

Try it yourself!

- https://beta.openai.com/playground