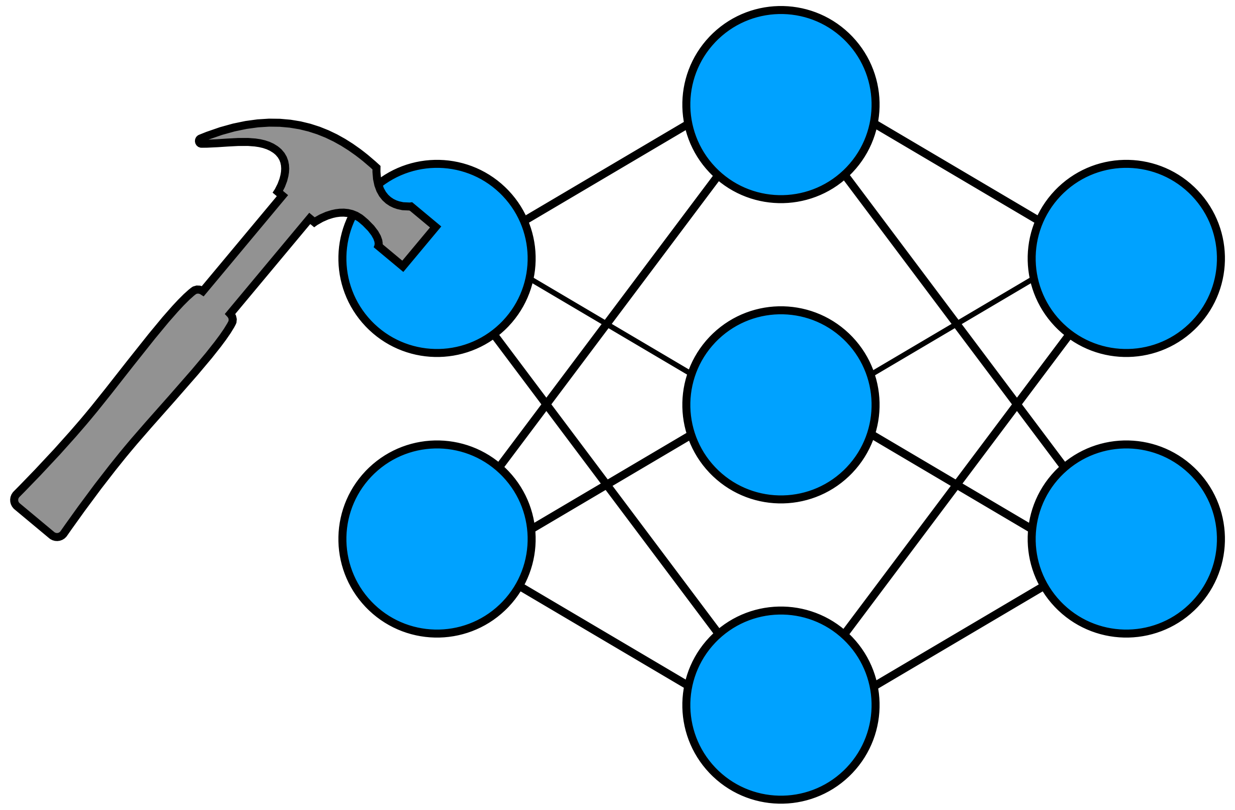
How to Build an LLM from Scratch

Shaw Talebi



Should Enterprises Consider A Large Language Model Strategy?

Forbes

Vibhore Kumar Forbes Councils Member
Forbes Technology Council COUNCIL POST | Membership (Fee-Based)

May 19, 2023, 09:45am EDT

Should You Purchase an LLM or Train Your Own?

An excerpt from our Training LLMs from Scratch piece to help you decide if you should purchase a large language model or train your own

Justin Tenuto

Share 1 comment 2 stars

Last Updated: Sep 25, 2023

Weights & Biases

Introducing BloombergGPT, Bloomberg's 50-billion parameter large language model, purpose-built from scratch for finance

[1]

March 30, 2023

Technology And Analytics

How to Train Generative AI Using Your Company's Data

by Tom Davenport and Maryam Alavi

Harvard Business Review

July 06, 2023

How much does it cost?

Llama 2 (7b)	~180,000 GPU hours [2]	→	**~10b model**	~100,000 GPU hours
Llama 2 (70b)	~1,700,000 GPU hours		**~100b model**	~1,000,000 GPU hours

Renting

Invidia A100: $1-2 per GPU per hour

⟹ **10b model: $150,000**
100b model: $1,500,000

Buying

Invidia A100: ~$10,000

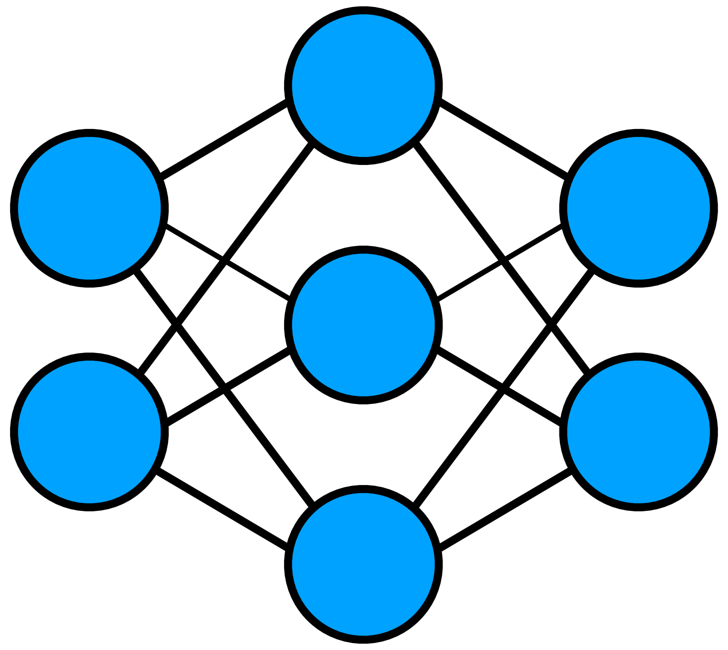
⟹ **GPU Cluster**: ~$10,000 x 1000 = **$10,000,000**

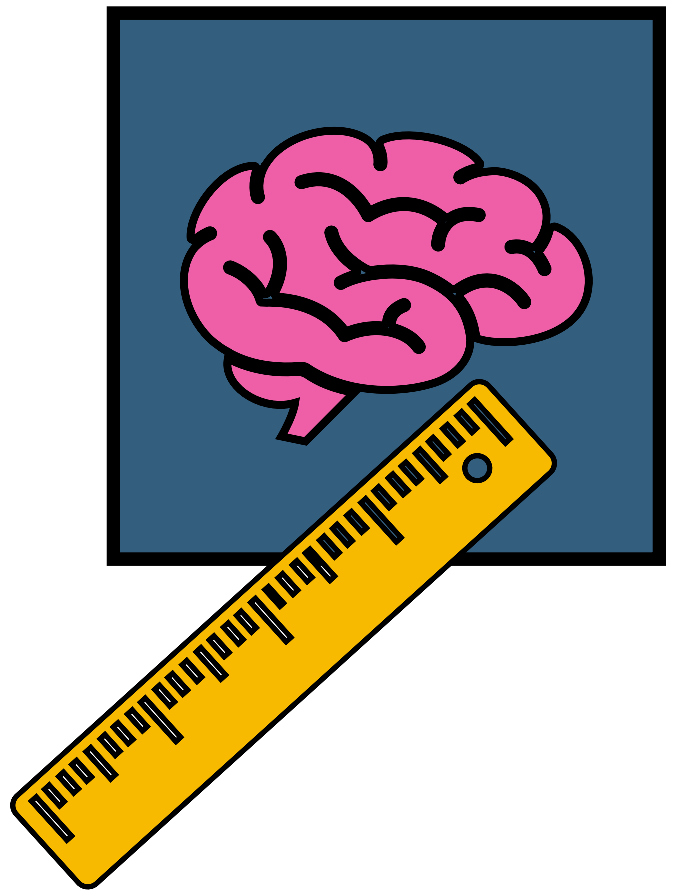
Training Energy Cost (100b model): ~1,000 megawatt hour [3]
Energy Price: ~$100 per megawatt hour

⟹ **Marginal Energy cost (100b model)**:
1,000 x $100 = **$100,000**

4 Key Steps

1. **Data Curation**
2. **Model Architecture**
3. **Training at Scale**
4. **Evaluation**





Step 1: Data Curation

The quality of your model is driven by the quality of your data

0.5T tokens

GPT-3 175b[4]

2T tokens

Llama 2 70b[2]

3.5T tokens

Falcon 180b[5]

Trillion words ≈ 1,000,000 novels ≈ 1,000,000,000 news articles

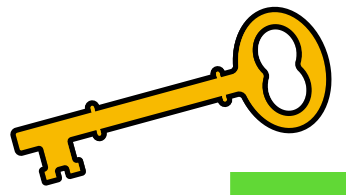
Step 1: Data Curation

Where do we get all these data?

The internet e.g. web pages, wikipedia, forums, books, scientific articles, code bases, etc.

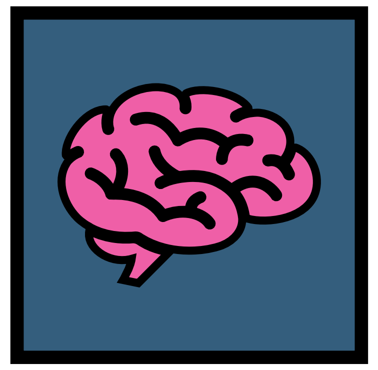
Public datasets

- Common Crawl (Colossal Clean Crawled Corpus i.e. C4, Falcon RefinedWeb)
- The Pile[6]
- Hugging Face Datasets



Private data sources e.g. FinPile (BloombergGPT) drawn from Bloomberg archives[1]

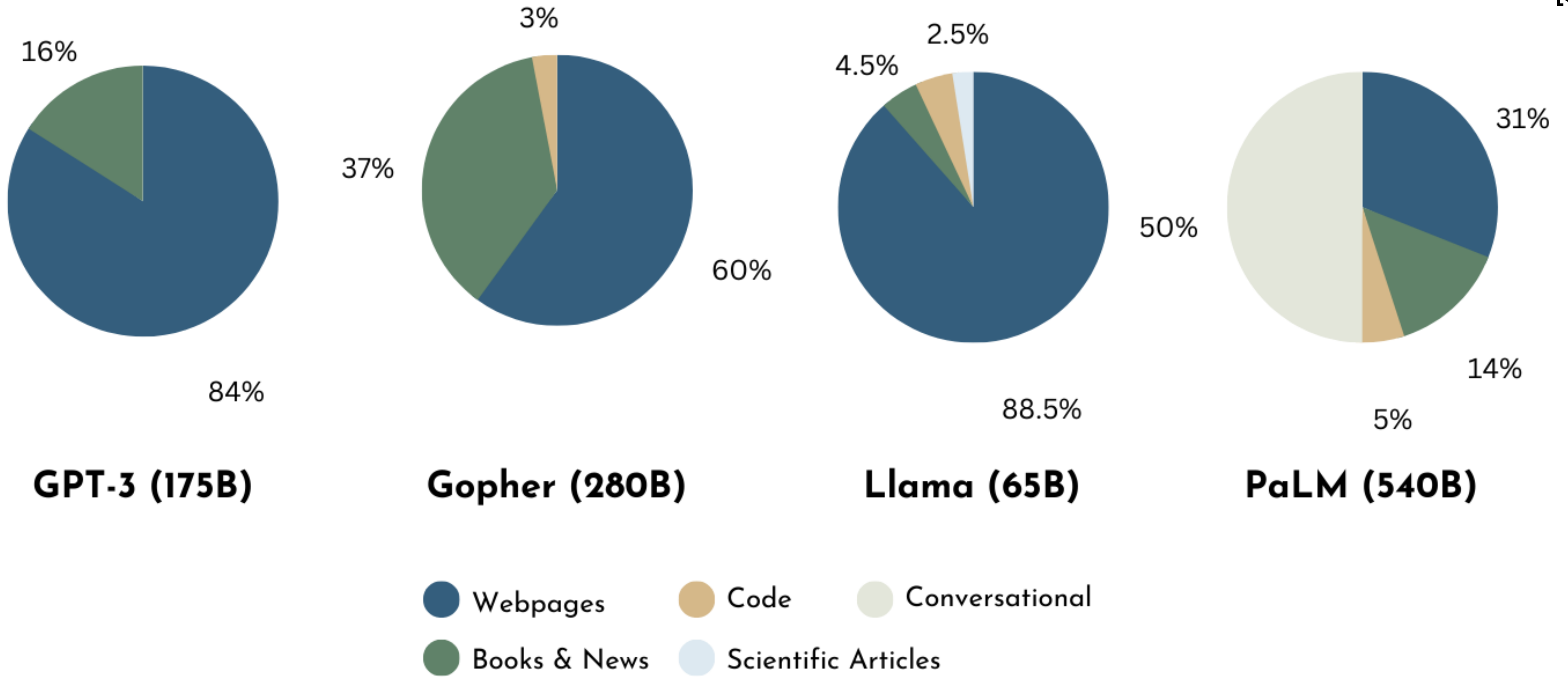
Use an LLM e.g. Alpaca - an LLM trained on structured text generated by GPT-3[7]

Make me training data →  → **Okay.**

Step 1: Data Curation

Dataset Diversity

[8]

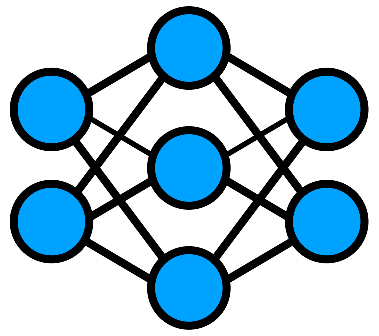


Step 1: Data Curation

How do we prepare the data?

Quality Filtering - remove “low-quality” text from dataset [8]

Classifier-based



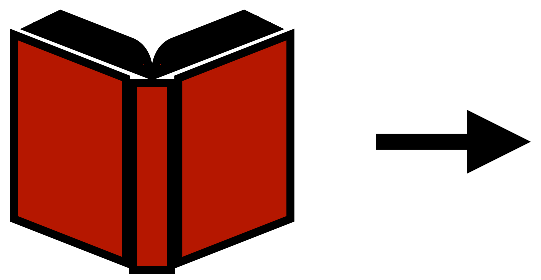
Heuristic-based

De-duplication - several instances of same (or very similar) text can bias model and disrupt training [8, 9]

Privacy Redaction - removal of sensitive and confidential information

Tokenization - translate text into numbers [8]

Bytepair Encoding Algorithm [10]



Efficient sub-word Vocabulary

Libraries: SentencePiece, Tokenizers [11, 12]

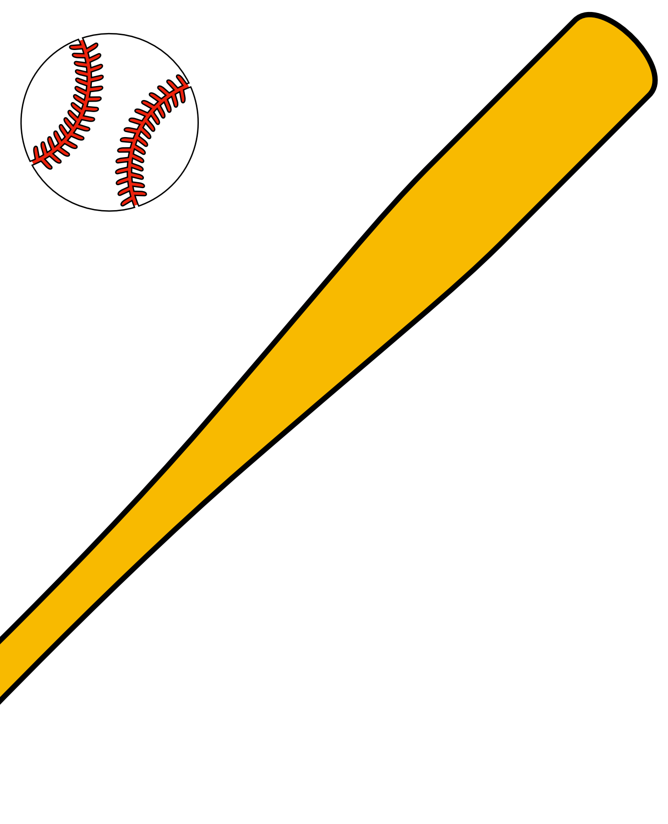
Step 2: Model Architecture

Transformers

Neural network architecture that uses **attention** mechanisms

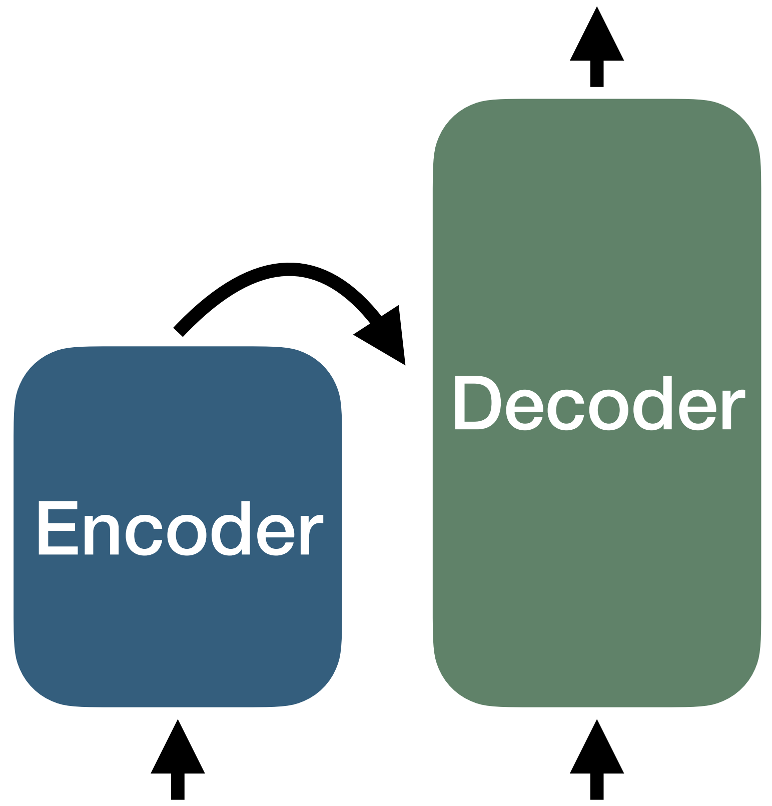
Attention mechanism - learns dependencies between different elements of a sequence based on position and content [13]

“I hit the baseball with a bat”



“I hit the bat with a baseball”

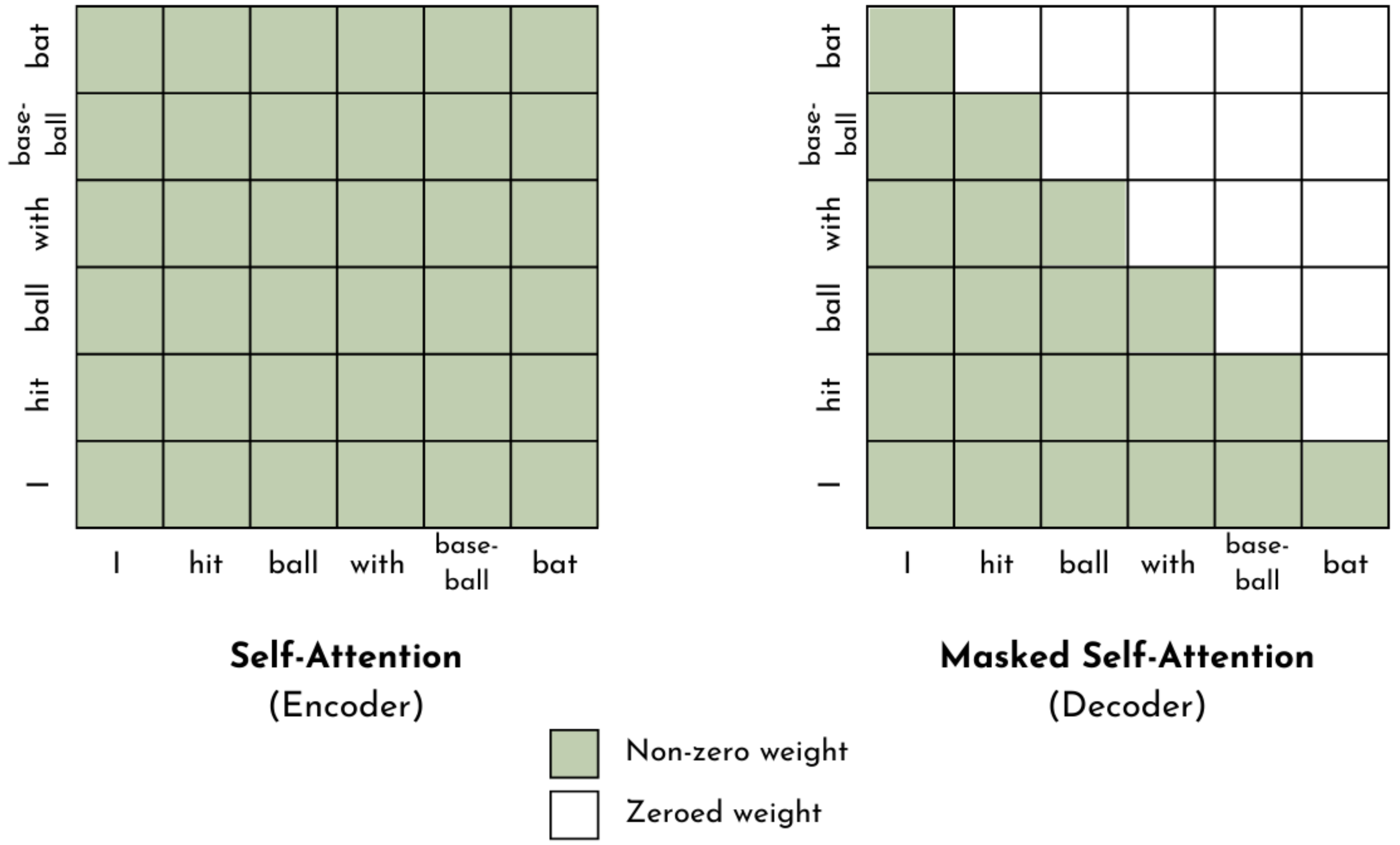
Step 2: Model Architecture



3 Types of Transformers [14, 15]

Encoder-only - encoder translates tokens into a semantically meaningful representation | *tasks: text classification* [15]

Decoder-only - similar to encoder but does not allow self-attention with future elements | *tasks: text generation* [8, 15]



Encoder-decoder - combines both and allows cross-attention | *tasks: translation* [13, 15]

Step 2: Model Architecture

Other Design Choices

Residual Connections - allow intermediate training values to bypass hidden layers[14]

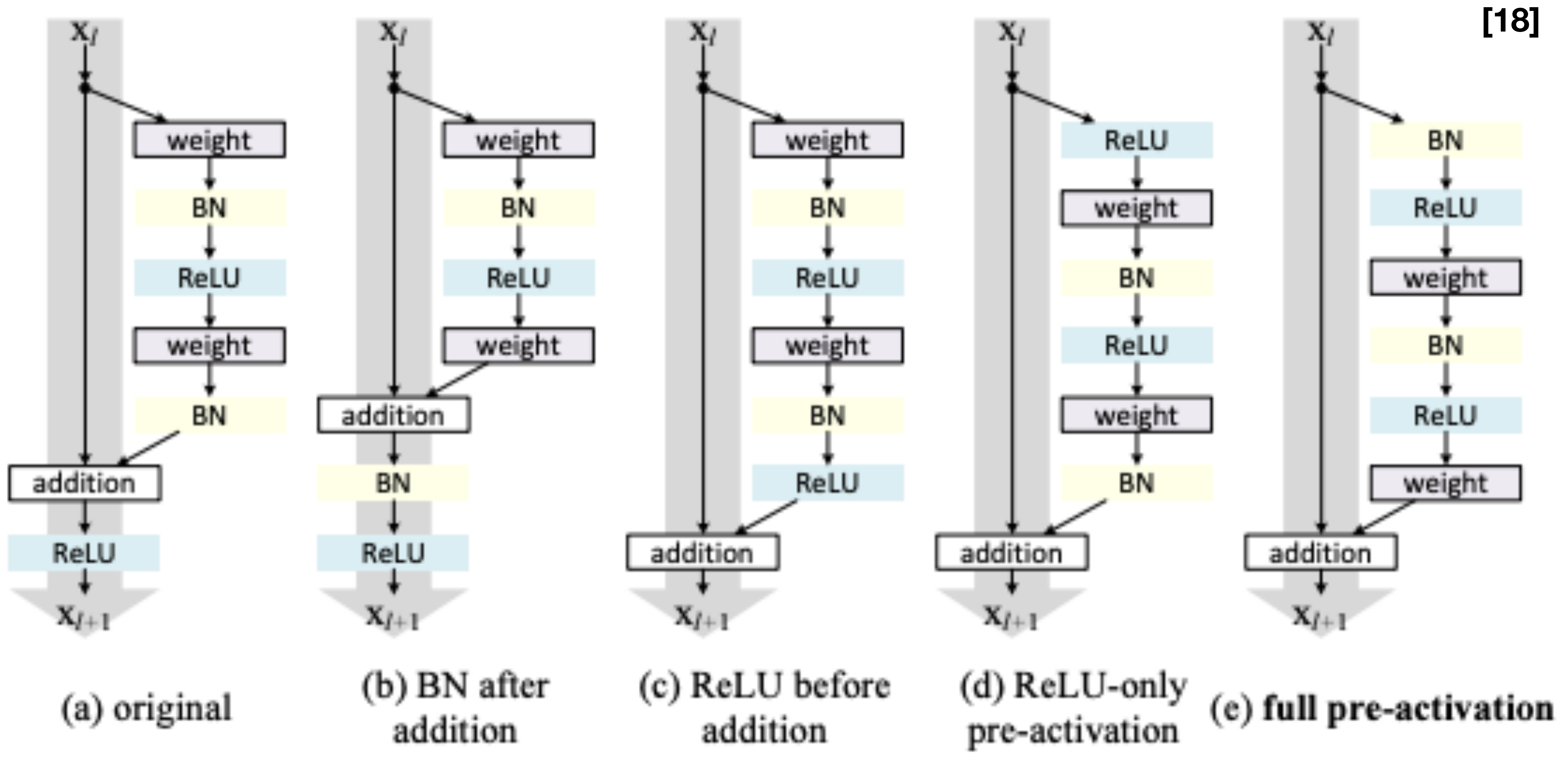
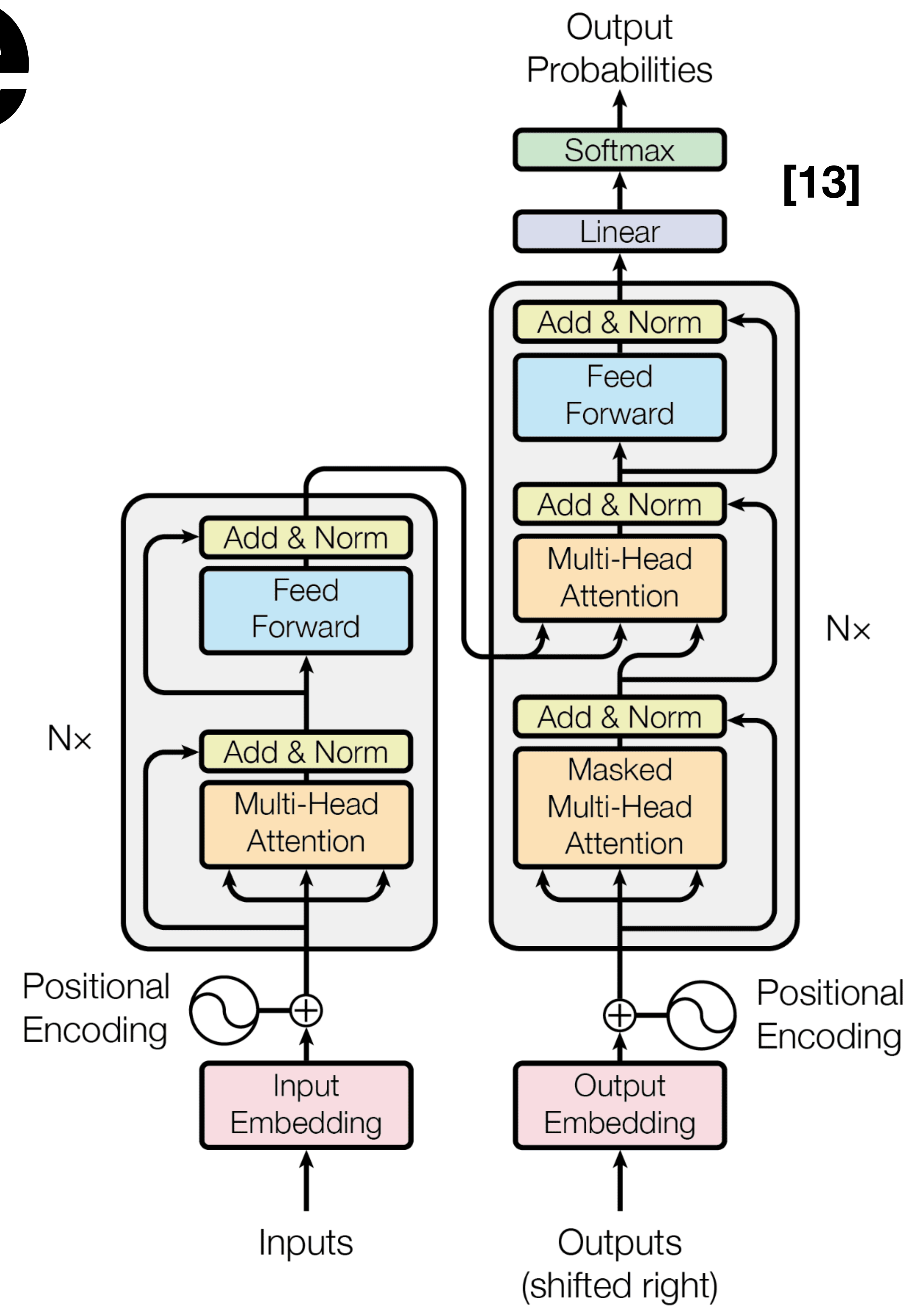


Figure 4. Various usages of activation in Table 2. All these units consist of the same components — only the orders are different.



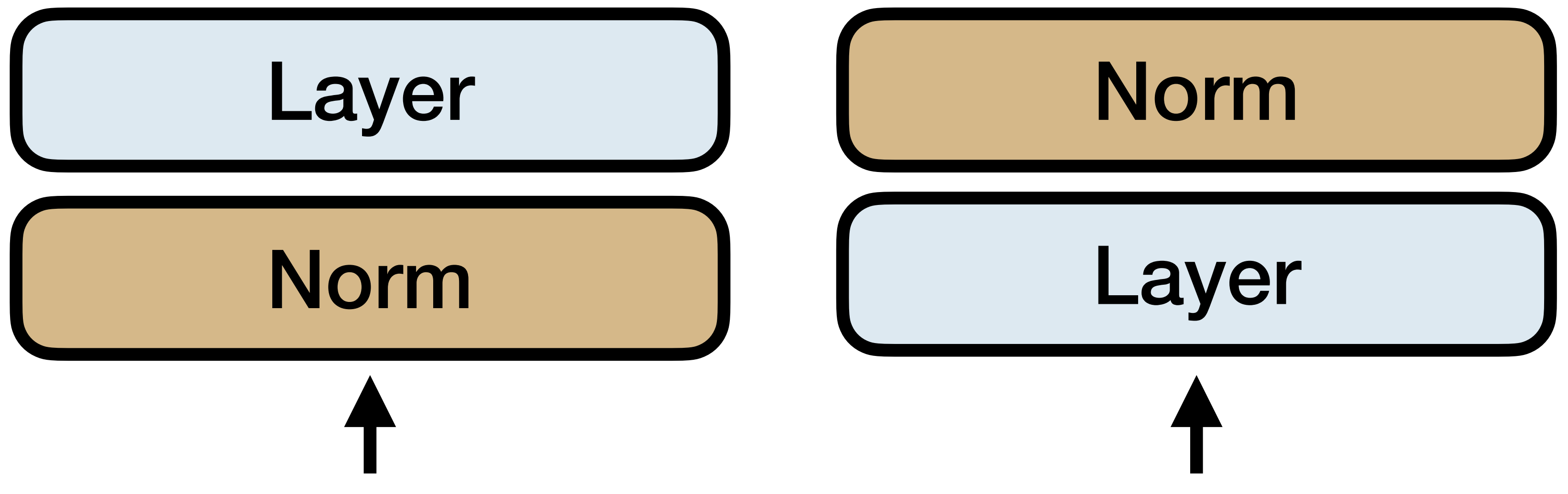
Step 2: Model Architecture

Other Design Choices

Residual Connections - allow intermediate training values to bypass hidden layers

Layer Normalization - re-scaling values between layers based on their mean and standard deviation

Where you normalize



Pre- [8] Post-

How you normalize

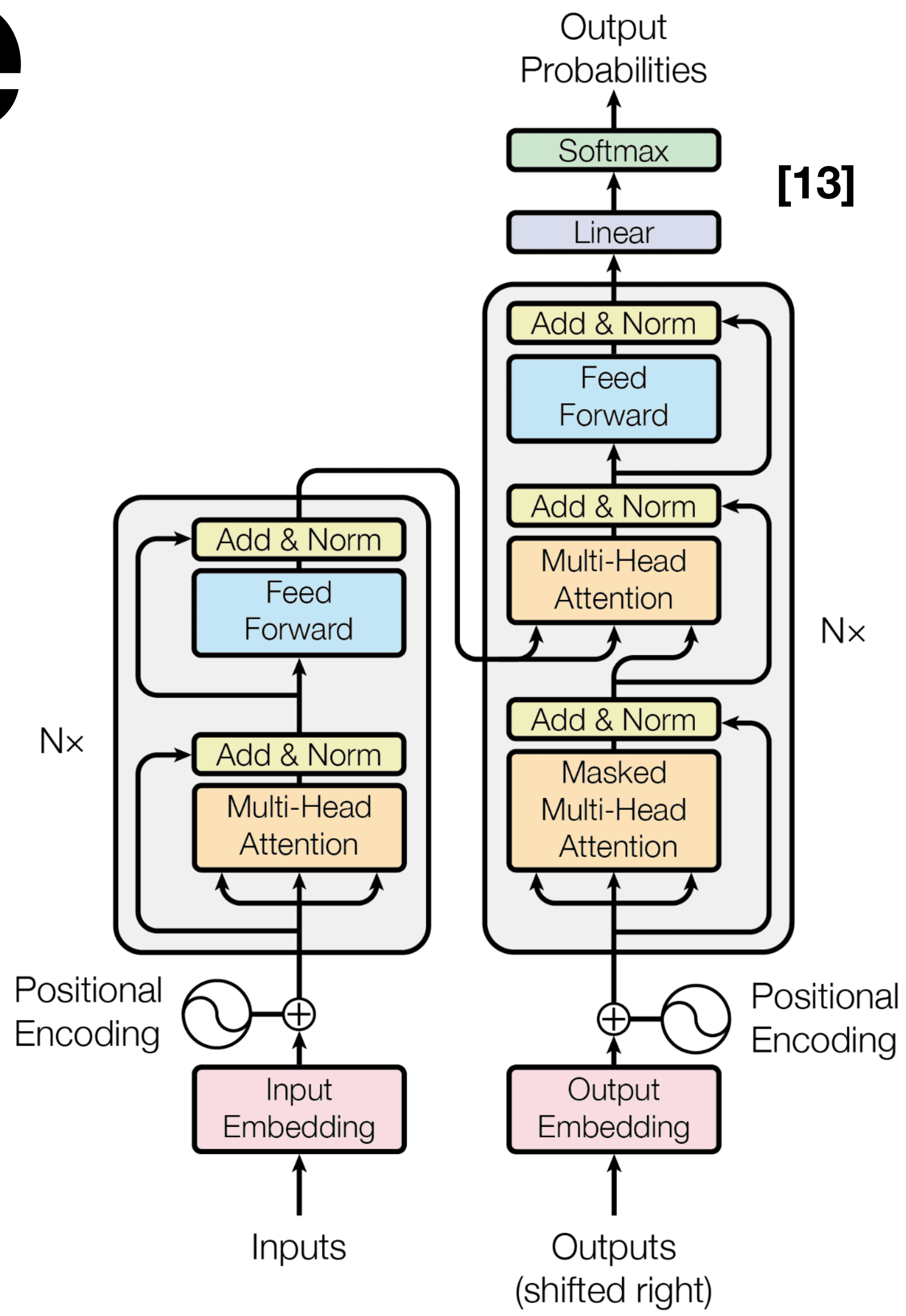
$$y = \frac{x - \bar{x}}{\sqrt{Var(x) + \epsilon}} \times \gamma + \beta$$

Layer Norm [8]

$$y = \frac{x}{RMS(x)} \times \gamma + \beta$$

RMS Norm

Step 2: Model Architecture



[13]

Other Design Choices

Residual Connections - allow intermediate training values to bypass hidden layers

Layer Normalization - re-scaling values between layers based on their mean and star

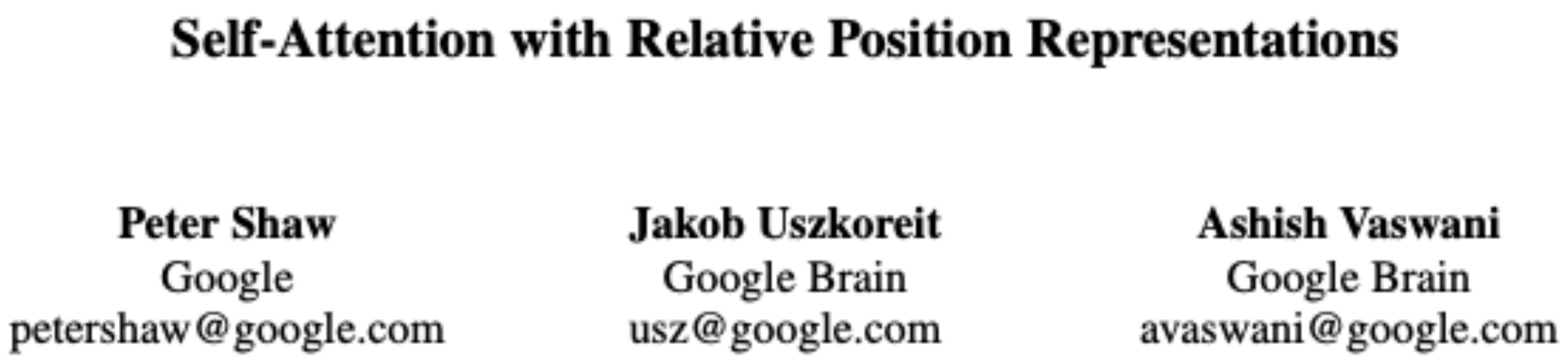
[8]

Activation Functions - introduce non-linearities into model *e.g. GeLU, ReLU, Swish, Sw*

Position Embedding - captures information about token positions

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

Positional encodings [13]



Self-Attention with Relative Position Representations

Peter Shaw
Google
petershaw@google.com

Jakob Uszkoreit
Google Brain
usz@google.com

Ashish Vaswani
Google Brain
avaswani@google.com

Relative positional encodings [20]

Step 2: Model Architecture

How big do I make it?

If model is too big or trained too long, it can overfit

If model is too small or not trained long enough, it can underperform

[21]

Parameters	FLOPs	Tokens
400 Million	1.92e+19	8.0 Billion
1 Billion	1.21e+20	20.2 Billion
10 Billion	1.23e+22	205.1 Billion
67 Billion	5.76e+23	1.5 Trillion
175 Billion	3.85e+24	3.7 Trillion
280 Billion	9.90e+24	5.9 Trillion
520 Billion	3.43e+25	11.0 Trillion
1 Trillion	1.27e+26	21.2 Trillion
10 Trillion	1.30e+28	216.2 Trillion

~20 tokens per model parameter

~100x increase in FLOPs for each 10x increase in model parameters

Step 3: Training at Scale

3 Training Techniques

Mixed Precision Training - uses both 32-bit and 16-bit floating point data types [8, 22]

3D Parallelism - combination of pipeline, model, and data parallelism[8]

- **Pipeline Parallelism** - distributes transformer layers across multiple GPUs
- **Model Parallelism** - decomposes parameter matrix operation into multiple matrix multiplies distributed across multiple GPUs
- **Data Parallelism** - distributes training data across multiple GPUs

Zero Redundancy Optimizer (ZeRO) - reduces data redundancy regarding the optimizer state, gradient, or parameter partitioning [8]

DeepSpeed [23]

Step 3: Training at Scale

Training Stability

Checkpointing - takes a snapshot of model artifacts so training can resume from that point [8]

Weight Decay - regularization strategy that penalizes large parameter values by adding a term (e.g. L2 norm of weights) to the loss function or changing the parameter update rule [8, 24]

Gradient Clipping - rescales the gradient of the objective function if its norm exceeds a pre-specified value [8, 25]

Step 3: Training at Scale

Hyperparameters

Batch Size: (*Static*) typically ~16M tokens. (*Dynamic*) GPT-3 increased from 32K to 3.2M [8]

Learning Rate: (*Dynamic*) learning rate increases linearly until reaching a maximum value and then reduces via a cosine decay until the learning rate is about 10% of its max value [8]

Optimizer: Adam-based optimizers are most commonly used for LLMs[8]

Dropout: typical values between 0.2 and 0.5 [32]

[32] https://www.cs.toronto.edu/~hinton/absps/JMLRdropout.pdf

Step 4: Evaluation

Benchmark Dataset (Open LLM Leaderboard)

LLM Benchmark | About | Submit here!

Search for your model and press ENTER...

Select columns to show

Average, ARC, HellaSwag, MMLU, TruthfulQA, Type, Precision, Hub License, #Params (B), Hub, Model sha

Show gated/private/deleted models

Model types: pretrained, fine-tuned, instruction-tuned, RL-tuned, ? Unknown

Precision: torch.float16, torch.bfloat16, torch.float32, 8bit, 4bit, GPTQ

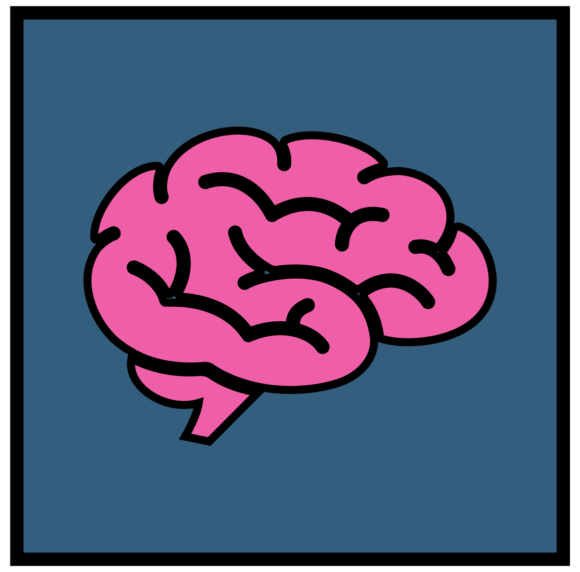
Model sizes: Unknown, < 1.5B, ~3B, ~7B, ~13B, ~35B, 60B+

Multiple-choice Tasks

T	Model	Average	ARC	HellaSwag	MMLU	TruthfulQA
🔶	AIDC-ai-business/Marcoroni-70B-v1	74.06	73.55	87.62	70.67	64.41
🔶	ICBU-NPU/FashionGPT-70B-V1.1	74.05	71.76	88.2	70.99	65.26
🔶	adonlee/LLaMA_2_70B_LoRA	73.9	72.7	87.55	70.84	64.52
🔶	uni-tianyan/Uni-TianYan	73.81	72.1	87.4	69.91	65.81
🔶	Riiid/sheep-duck-llama-2	73.69	72.35	87.78	70.82	63.8
			[27]	[28]	[29]	[30]

Step 4: Evaluation

Multiple-choice Tasks e.g. ARC, Hellaswag, MMLU

Listen to your →  → **heart.**

[31]

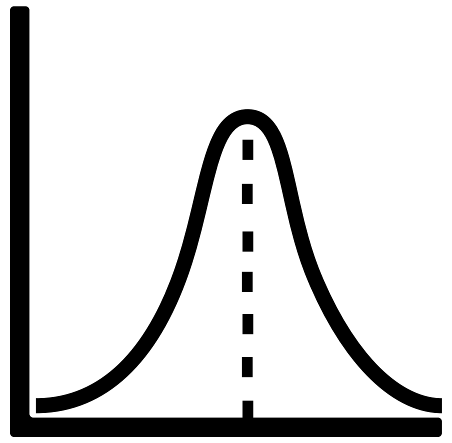
```
“””Question: Which technology was developed most recently?

Choices:
A. Cellular Phone
B. Television
C. Refrigerator
D. Airplane

Answer:”””
```

Step 4: Evaluation

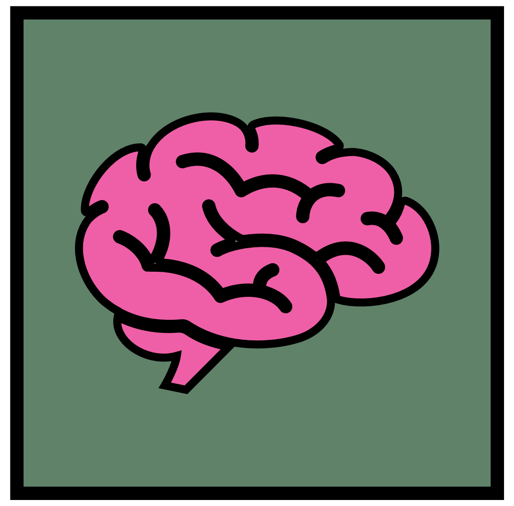
Open-ended Tasks e.g. TruthfulQA

Human Evaluation - a person scores completion based on ground truth, guidelines, or both

NLP Metrics - quantify completion quality via metrics such as Perplexity, BLEU, or ROGUE scores



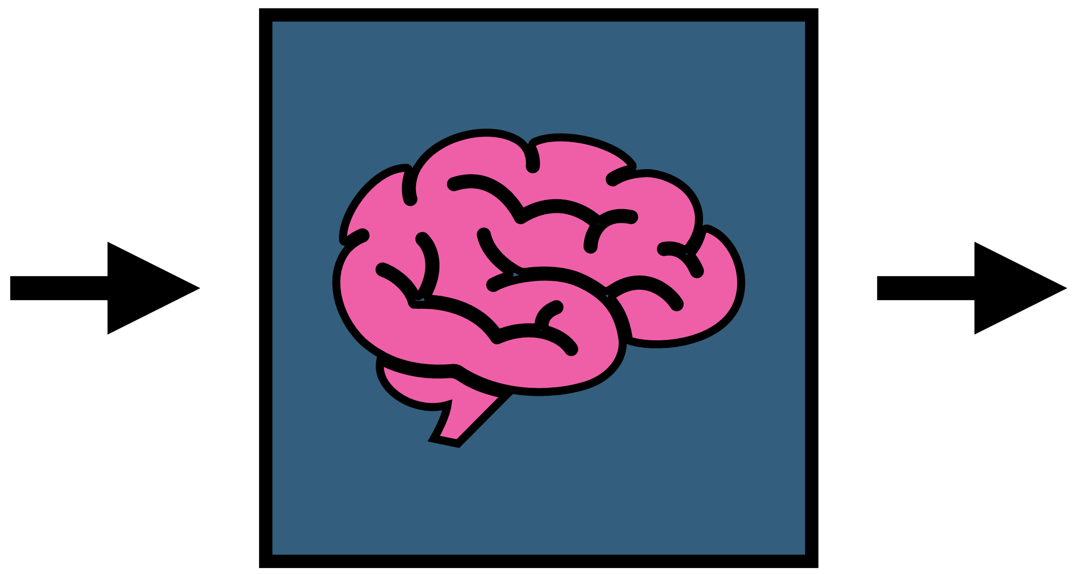
Auxiliary Fine-tuned LLM - use LLM to compare completions to ground truth[30]



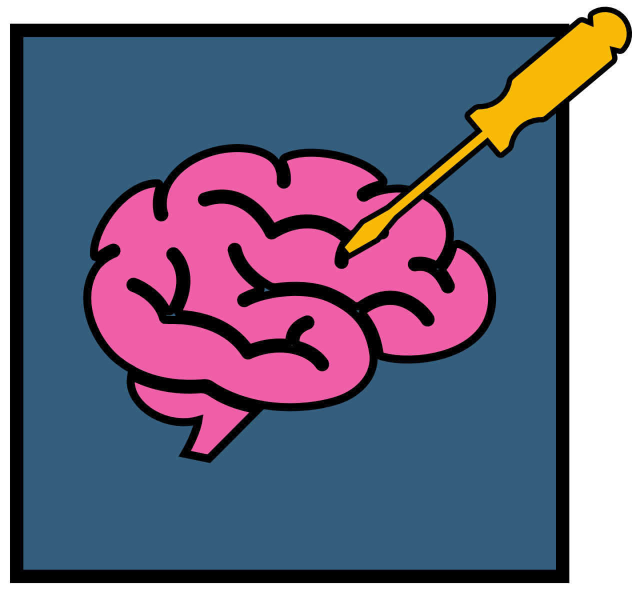
What's Next?

Base models are typically a starting point, not final solution

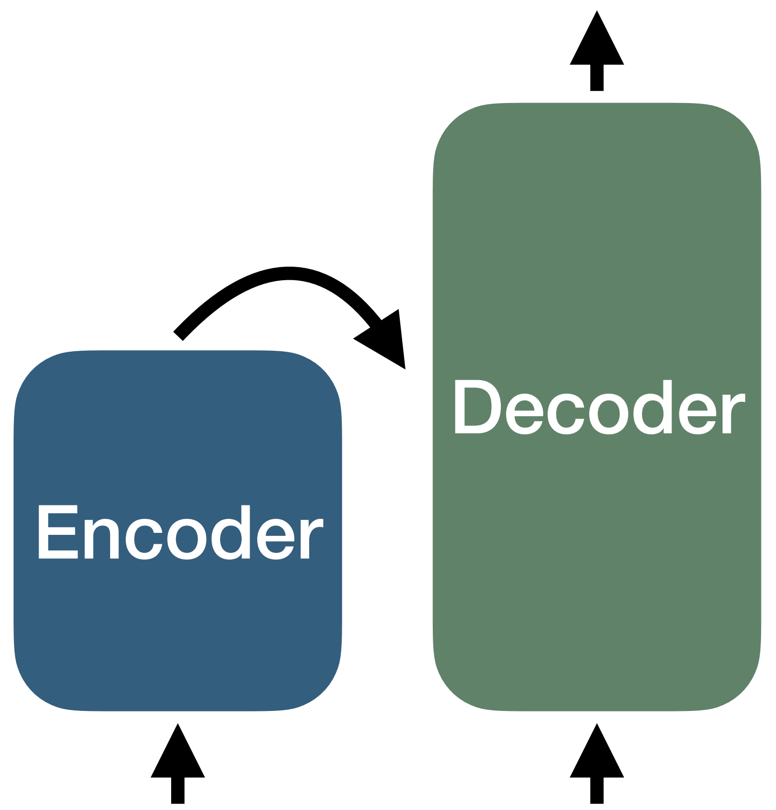
Prompt Engineering



Model fine-tuning



Step 2: Model Architecture



[15]
: text classification

[8, 15]
: text generation