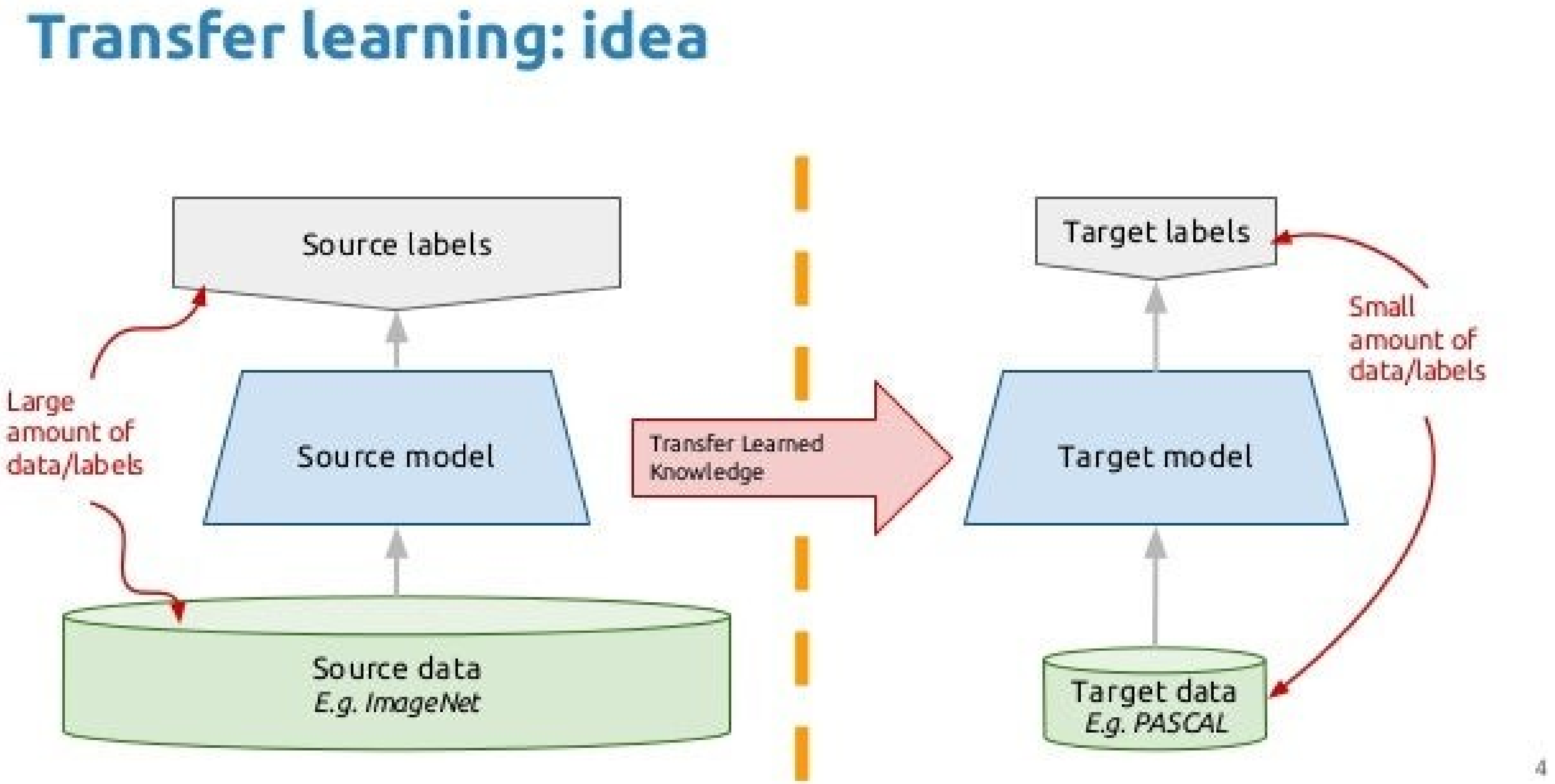
AI Implications for Business Strategy

7. Large Language Models

Transfer Learning

- Once we have models trained in a domain, we have a knowledge base
  - We can then transfer that knowledge to similar domains
  - For example, if you know how to drive a car, it’s easy to learn how to drive a truck
- In ML, transfer learning takes place by
  - Starting with a pre-trained model on a domain
  - Then fine tune the model by training it data from the new domain
  - Learning new patterns and tasks relies on referring to previously learned patterns and tasks
  - In production transfer learning training is faster than regular learning, more accurate and requires less training data
- Derived from cognitive learning theory

Transfer Learning



Transfer learning: idea

Source labels

Source model

Source data
E.g. ImageNet

Large amount of data/labels

Transfer Learned Knowledge

Target labels

Target model

Target data
E.g. PASCAL

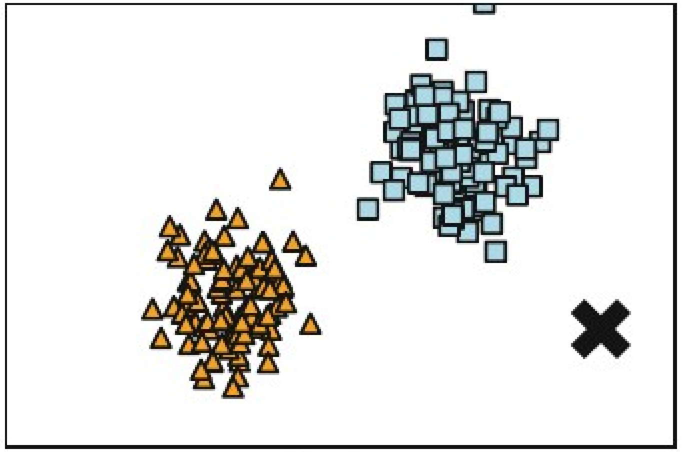
Small amount of data/labels

Discriminative Probability Models

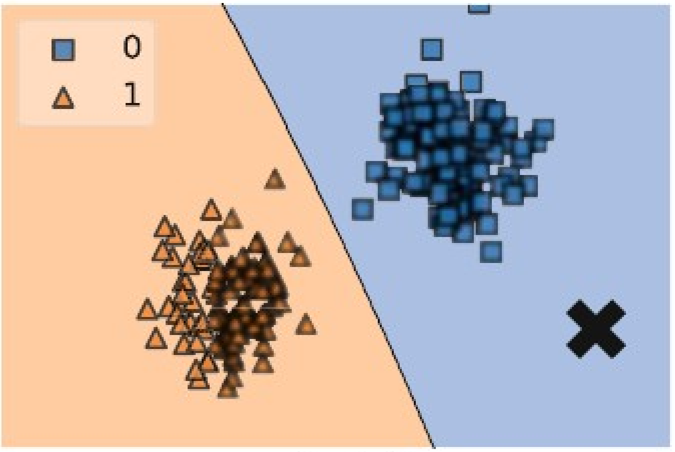
- Traditional ML produce probability distributions
  - However they lack context
  - Probability of classification is taken as a measure of certainty but this is limited to the probability distribution produces on the basis of the training data
  - It doesn’t take into account the center of mass of the probability distribution in context
  - A data point may be in one class based on the model but be quite different from the other members of the class (ie. the classification is spurious)
  - We might catch this mistake if we understood the probability distribution of the whole space across all the features.
- Example
  - Chairs might be classified in a distribution of furniture features based on appearance, size, the presence of a seat and other factors
  - A hologram of chair would be misclassified as a chair because it differs on other features

Discriminative Probability Models

- In the image below the X would be classified as blue
  - But it is distant from both of the centers of mass of classifications
  - This fact should increase the uncertainty of classifying it as blue
  - Can be we sure that there is not a third class that we didn’t detect in the training set?



Data

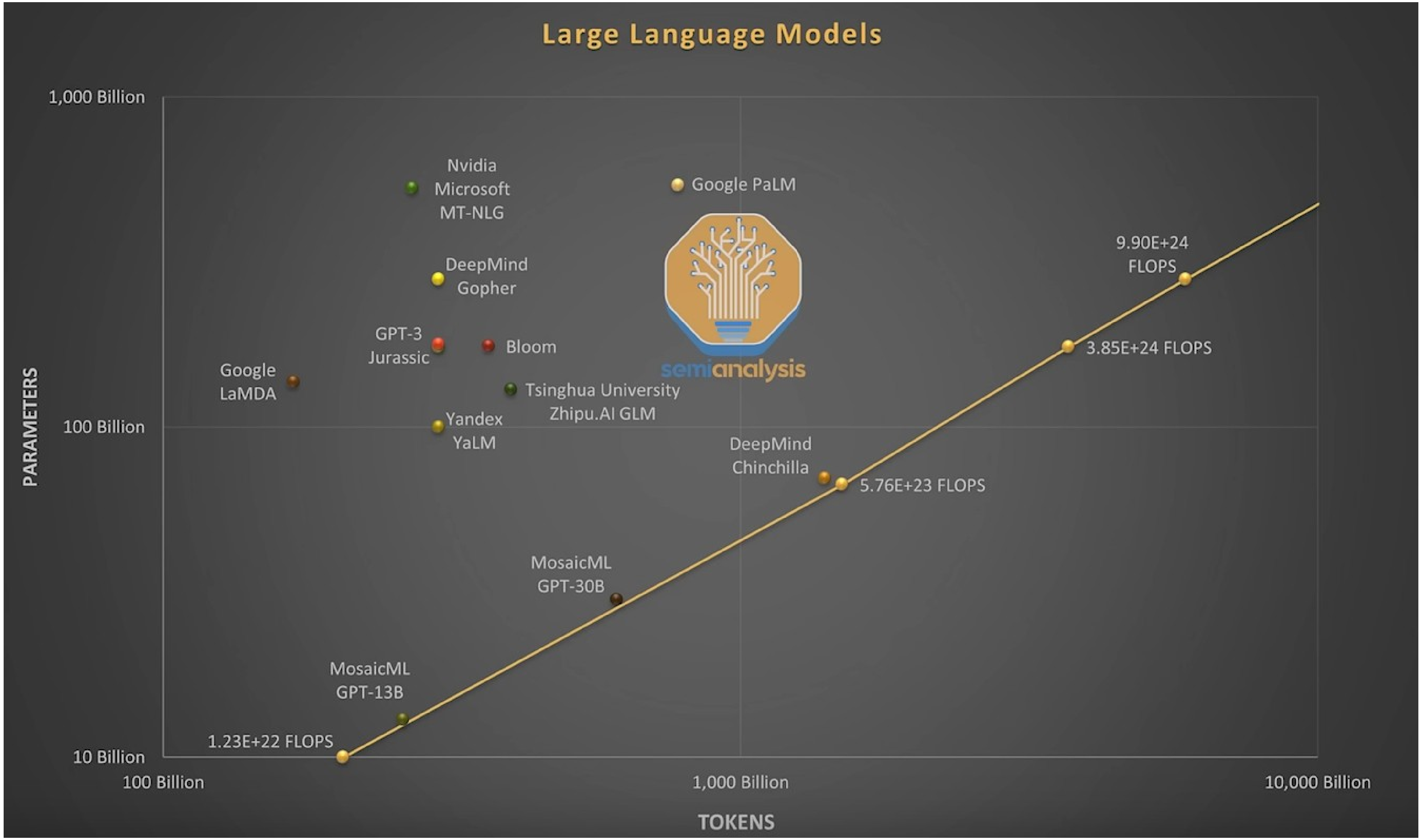


$p(y|\mathbf{x})$

Foundation Models

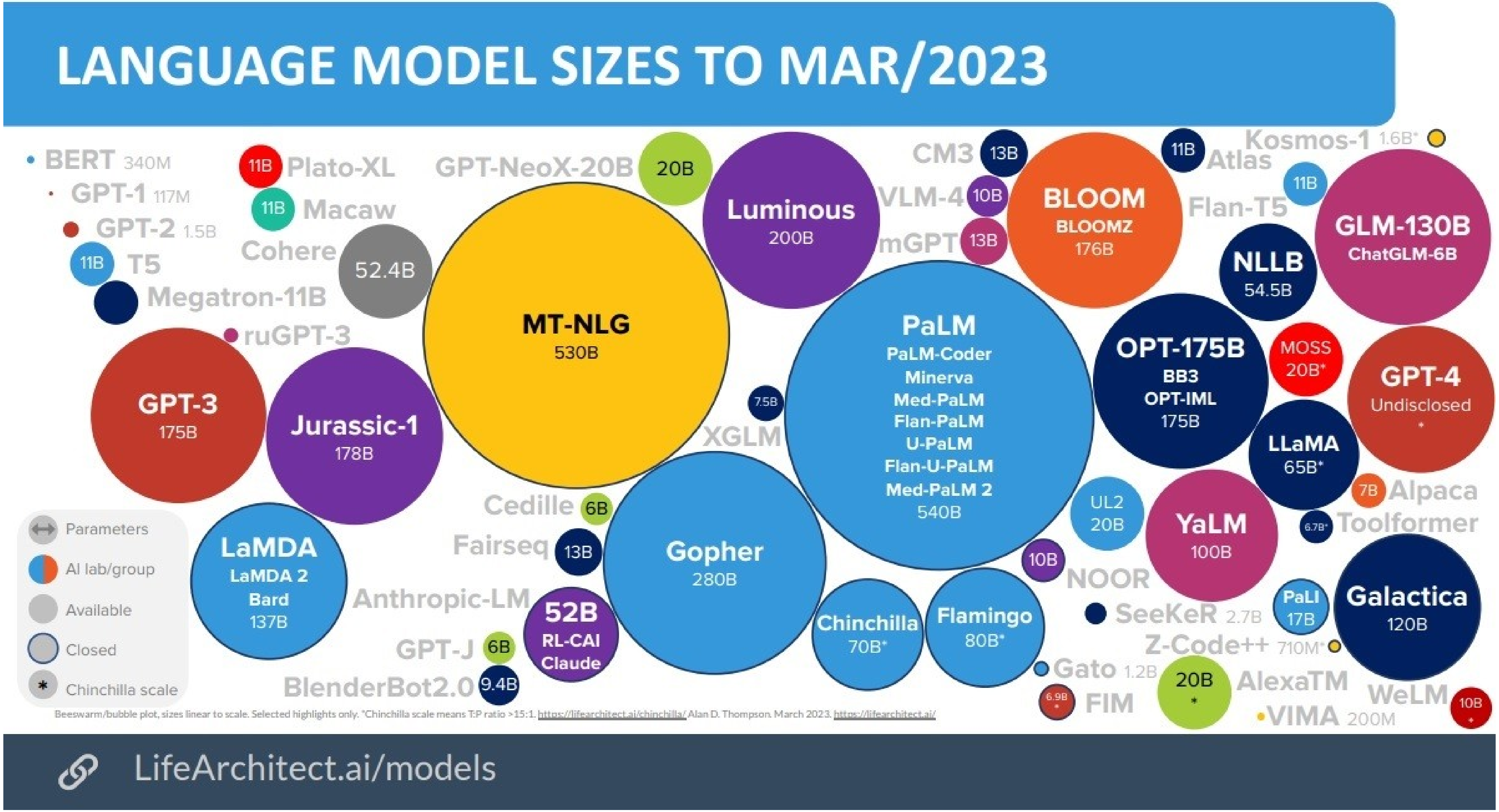
- Considered to be the revolution that enabled GAI
- Defined as:
  - *"any model that is trained on broad data (generally using self-supervision at scale) that can be adapted (e.g., fine-tuned) to a wide range of downstream tasks"*
- Prior to foundation models
  - AI models were trained on specific data sets for specific tasks, eg. facial recognition
  - Foundation models can replace all of the specialized models with adaptation to specialized domains
- Why now? Three enablers
  - Compute power available – massive numbers of dimensions and volume of input can be processed
  - Transformer architectures
  - In-context learning

Scale

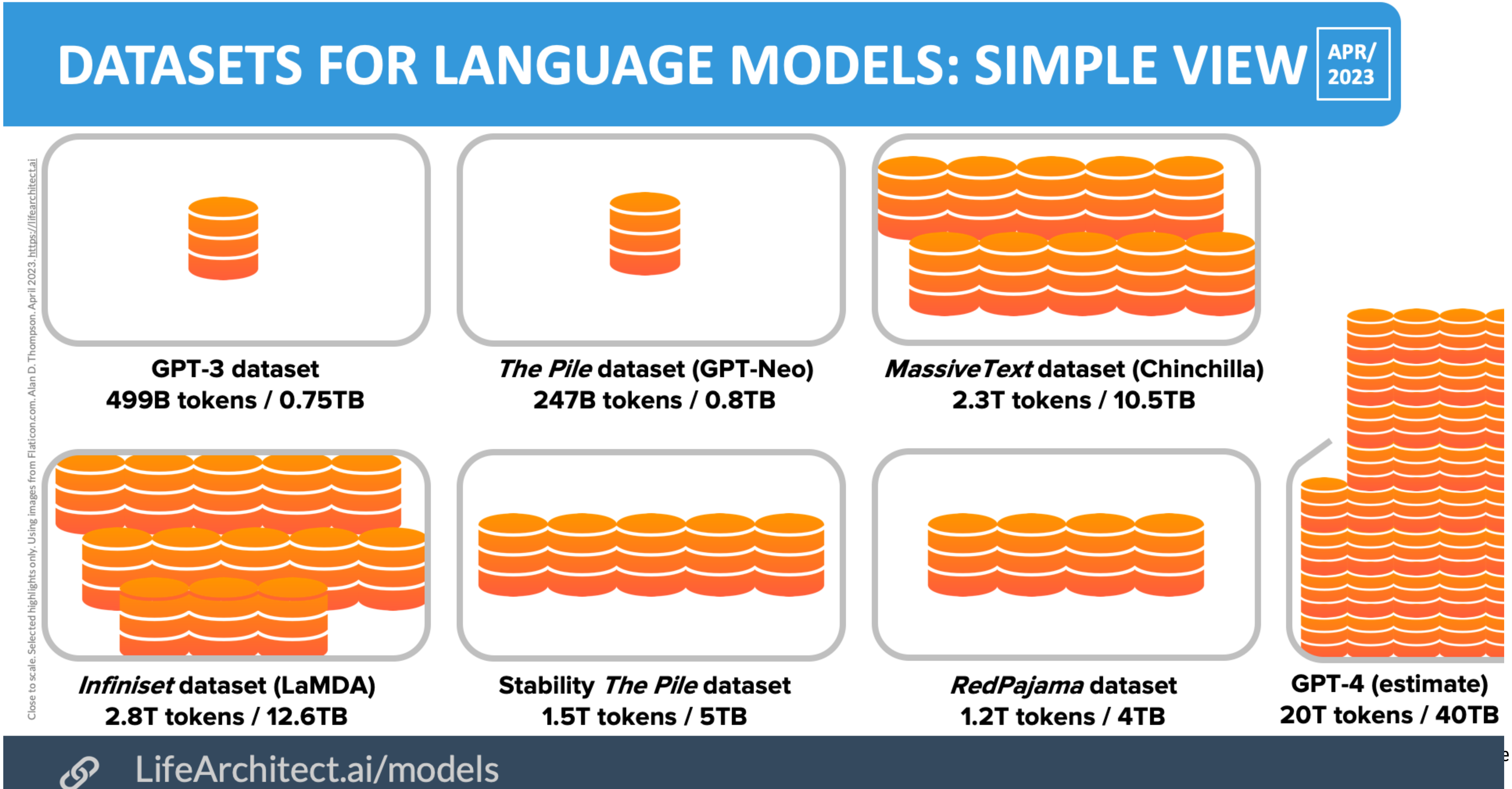


Large Language Models
1,000 Billion
100 Billion
10 Billion
PARAMETERS
100 Billion
1,000 Billion
10,000 Billion
TOKENS
Nvidia Microsoft MT-NLG
Google PaLM
DeepMind Gopher
GPT-3 Jurassic
Bloom
Google LaMDA
Tsinghua University Zhipu.AI GLM
Yandex YaLM
DeepMind Chinchilla
MosaicML GPT-30B
MosaicML GPT-13B
1.23E+22 FLOPS
5.76E+23 FLOPS
3.85E+24 FLOPS
9.90E+24 FLOPS
semianalysis

Scale


LANGUAGE MODEL SIZES TO MAR/2023
BERT 340M
GPT-1 117M
GPT-2 1.5B
11B T5
Megatron-11B
11B Plato-XL
11B Macaw
Cohere 52.4B
ruGPT-3
GPT-NeoX-20B 20B
MT-NLG 530B
Luminous 200B
CM3 13B
VLM-4 10B
mGPT 13B
BLOOM BLOOMZ 176B
11B Atlas
Kosmos-1 1.6B*
11B Flan-T5
GLM-130B ChatGLM-6B
NLLB 54.5B
GPT-3 175B
Jurassic-1 178B
7.5B XGLM
PaLM PaLM-Coder Minerva Med-PaLM Flan-PaLM U-PaLM Flan-U-PaLM Med-PaLM 2 540B
OPT-175B BB3 OPT-IML 175B
MOSS 20B*
GPT-4 Undisclosed *
LLaMA 65B*
7B Alpaca
6.7B* Toolformer
Cedille 6B
Fairseq 13B
Gopher 280B
LaMDA LaMDA 2 Bard 137B
Anthropic-LM 52B RL-CAI Claude
GPT-J 6B
BlenderBot2.0 9.4B
Chinchilla 70B*
Flamingo 80B*
10B NOOR
UL2 20B
YaLM 100B
SeeKeR 2.7B
PaLI 17B
Galactica 120B
Z-Code++ 710M*
Gato 1.2B
6.9B * FIM
20B * AlexaTM
WeLM 10B *
VIMA 200M
Parameters
AI lab/group
Available
Closed
* Chinchilla scale
Beeswarm/bubble plot, sizes linear to scale. Selected highlights only. *Chinchilla scale means T:P ratio >15:1. https://lifearchitect.ai/chinchilla/ Alan D. Thompson. March 2023. https://lifearchitect.ai/
LifeArchitect.ai/models

Scale



DATASETS FOR LANGUAGE MODELS: SIMPLE VIEW

APR/
2023

GPT-3 dataset
499B tokens / 0.75TB

***The Pile* dataset (GPT-Neo)**
247B tokens / 0.8TB

***MassiveText* dataset (Chinchilla)**
2.3T tokens / 10.5TB

***Infiniset* dataset (LaMDA)**
2.8T tokens / 12.6TB

Stability *The Pile* dataset
1.5T tokens / 5TB

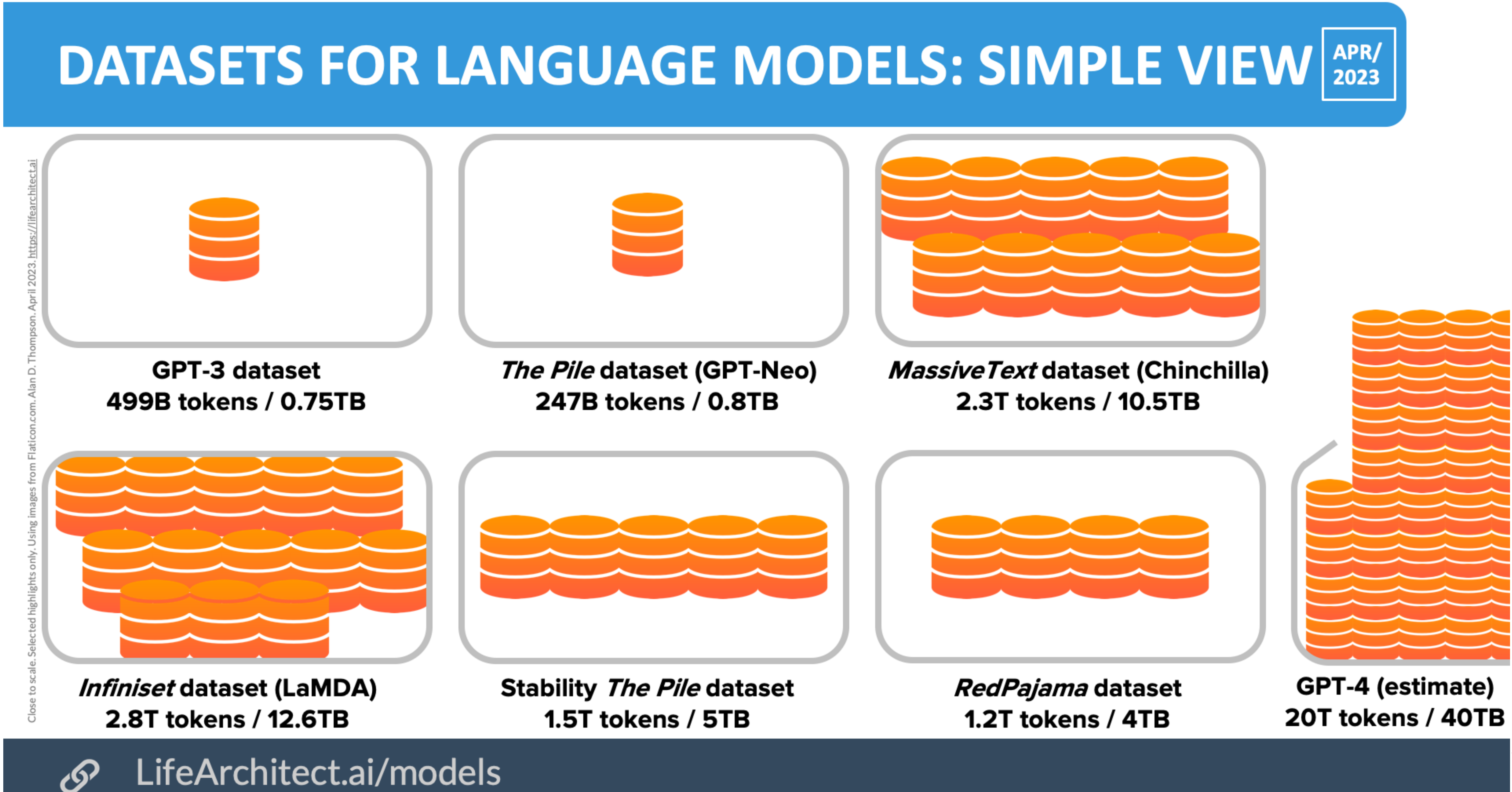
***RedPajama* dataset**
1.2T tokens / 4TB

GPT-4 (estimate)
20T tokens / 40TB

Close to scale. Selected highlights only. Using images from Flaticon.com. Alan D. Thompson. April 2023. https://lifearchitect.ai

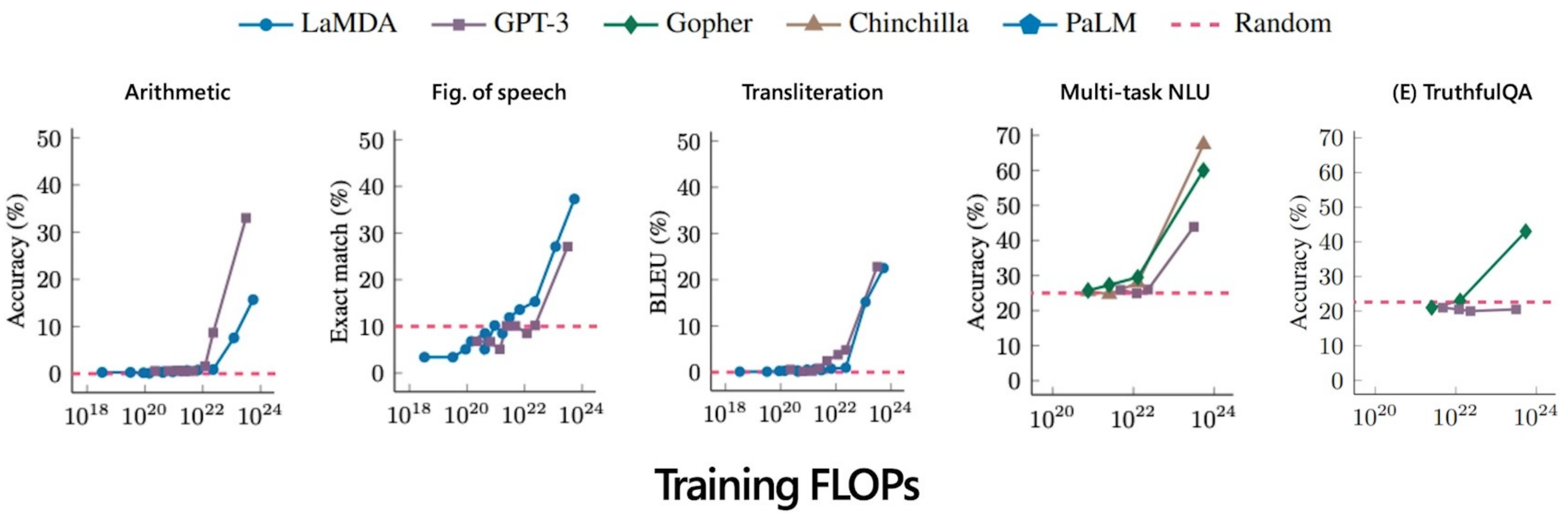
LifeArchitect.ai/models

Scale

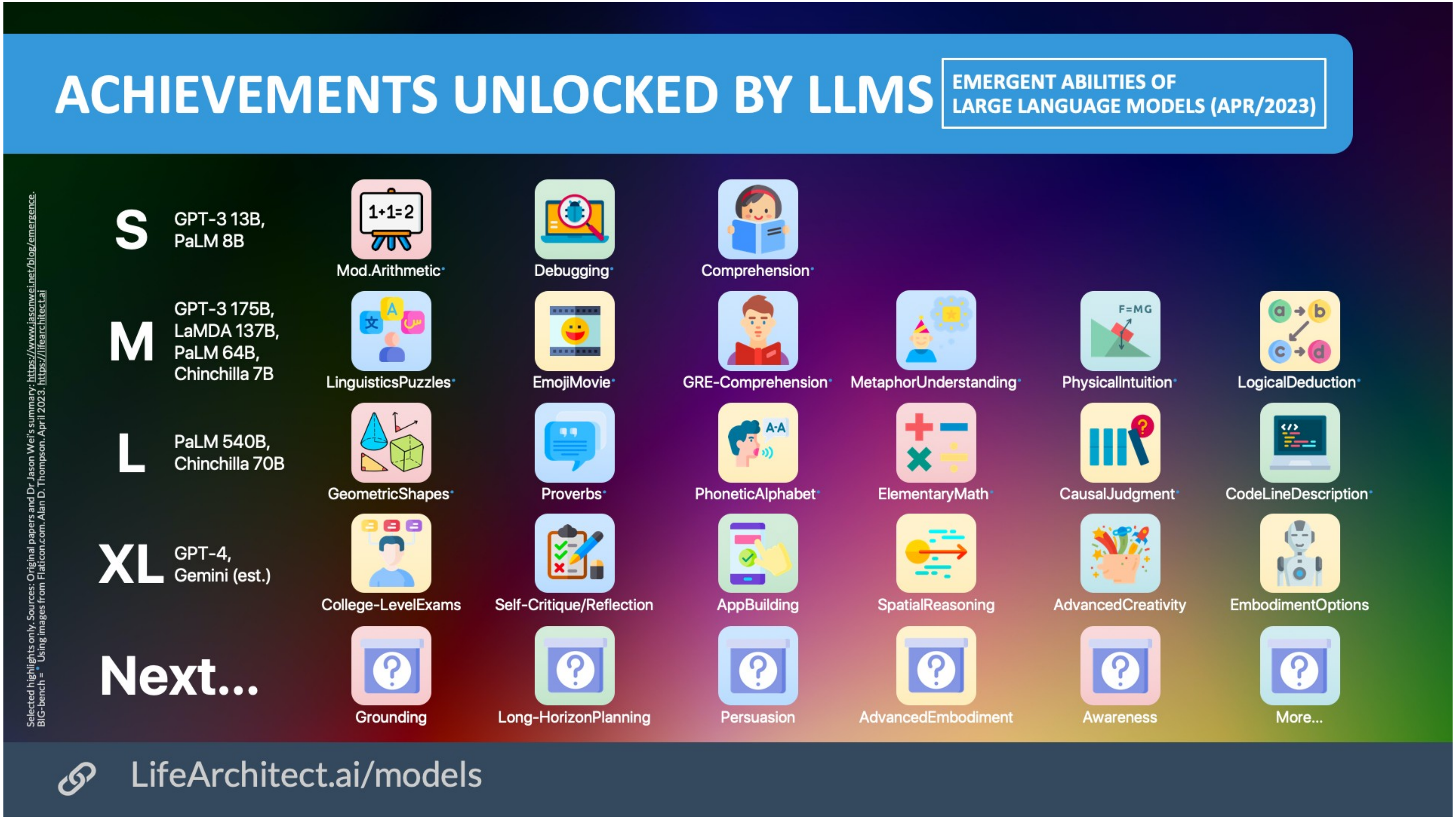

DATASETS FOR LANGUAGE MODELS: SIMPLE VIEW
APR/ 2023
GPT-3 dataset
499B tokens / 0.75TB
The Pile dataset (GPT-Neo)
247B tokens / 0.8TB
MassiveText dataset (Chinchilla)
2.3T tokens / 10.5TB
Infiniset dataset (LaMDA)
2.8T tokens / 12.6TB
Stability *The Pile* dataset
1.5T tokens / 5TB
RedPajama dataset
1.2T tokens / 4TB
GPT-4 (estimate)
20T tokens / 40TB
Close to scale. Selected highlights only. Using images from Flaticon.com. Alan D. Thompson. April 2023. https://lifearchitect.ai
LifeArchitect.ai/models

Impact of Scale

- Models show emergent behavior or unexpected capabilities but only when a critical threshold of size and complexity is reached because of increased compute (MFLOPS)



Impact of Scale



ACHIEVEMENTS UNLOCKED BY LLMS

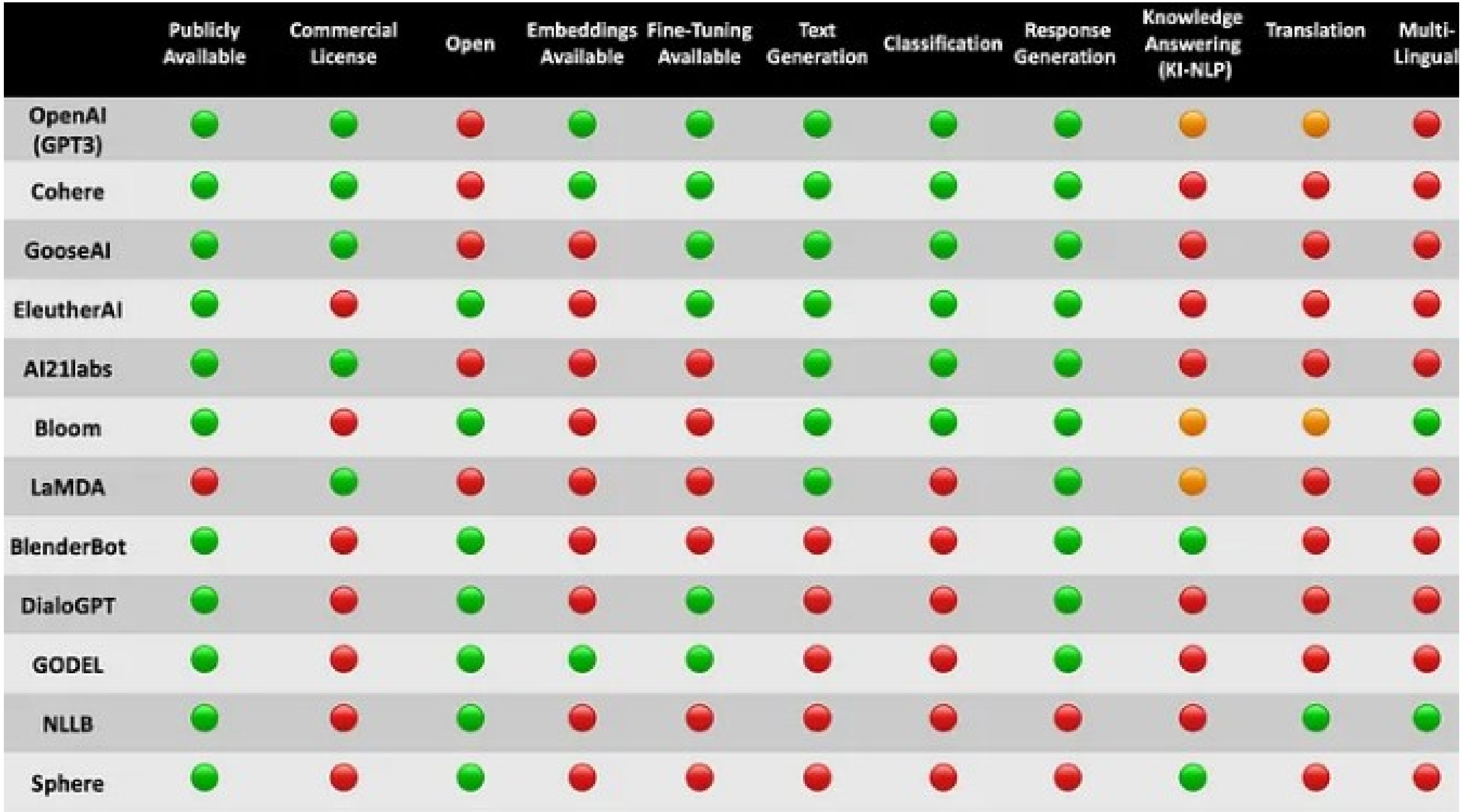
EMERGENT ABILITIES OF LARGE LANGUAGE MODELS (APR/2023)

S	GPT-3 13B, PaLM 8B	Mod.Arithmetic•	Debugging•	Comprehension•			
M	GPT-3 175B, LaMDA 137B, PaLM 64B, Chinchilla 7B	LinguisticsPuzzles•	EmojiMovie•	GRE-Comprehension•	MetaphorUnderstanding•	PhysicalIntuition•	LogicalDeduction•
L	PaLM 540B, Chinchilla 70B	GeometricShapes•	Proverbs•	PhoneticAlphabet•	ElementaryMath•	CausalJudgment•	CodeLineDescription•
XL	GPT-4, Gemini (est.)	College-LevelExams	Self-Critique/Reflection	AppBuilding	SpatialReasoning	AdvancedCreativity	EmbodimentOptions
Next...		Grounding	Long-HorizonPlanning	Persuasion	AdvancedEmbodiment	Awareness	More...

Selected highlights only. Sources: Original papers and Dr Jason Wei's summary: https://www.jasonwei.net/blog/emergence.
BIG-bench = • Using images from Flaticon.com. Alan D. Thompson. April 2023. https://lifearchitect.ai

LifeArchitect.ai/models

Large Language Models



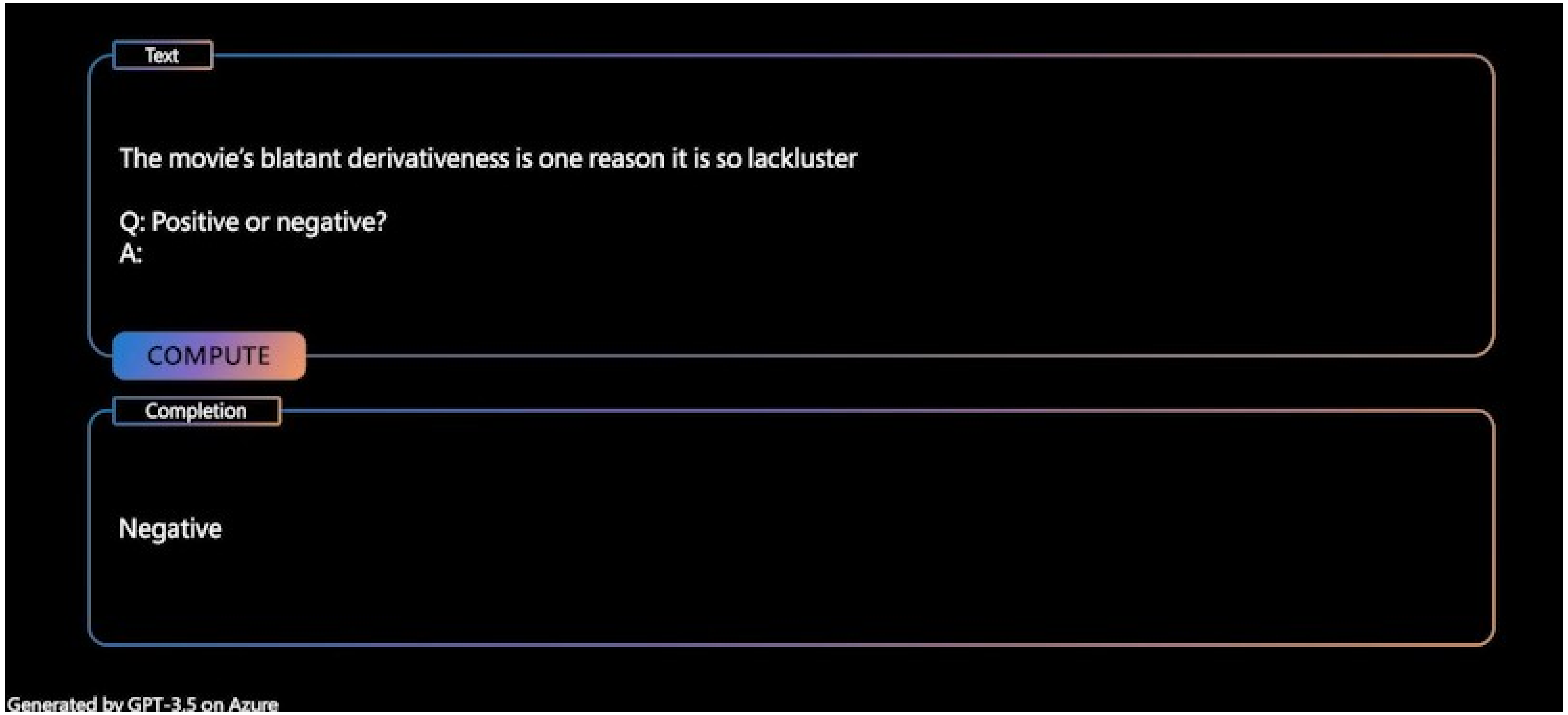
	Publicly Available	Commercial License	Open	Embeddings Available	Fine-Tuning Available	Text Generation	Classification	Response Generation	Knowledge Answering (KI-NLP)	Translation	Multi-Lingual
OpenAI (GPT3)	🟢	🟢	🔴	🟢	🟢	🟢	🟢	🟢	🟠	🟠	🔴
Cohere	🟢	🟢	🔴	🟢	🟢	🟢	🟢	🟢	🔴	🔴	🔴
GooseAI	🟢	🟢	🔴	🔴	🟢	🟢	🟢	🟢	🔴	🔴	🔴
EleutherAI	🟢	🔴	🟢	🔴	🟢	🟢	🟢	🟢	🔴	🔴	🔴
AI21labs	🟢	🟢	🔴	🔴	🔴	🟢	🟢	🟢	🔴	🔴	🔴
Bloom	🟢	🔴	🟢	🔴	🔴	🟢	🟢	🟢	🟠	🟠	🟢
LaMDA	🔴	🟢	🔴	🔴	🔴	🟢	🔴	🟢	🟠	🔴	🔴
BlenderBot	🟢	🔴	🟢	🔴	🔴	🔴	🔴	🟢	🟢	🔴	🔴
DialoGPT	🟢	🔴	🟢	🔴	🟢	🔴	🔴	🟢	🔴	🔴	🔴
GODEL	🟢	🔴	🟢	🟢	🟢	🔴	🔴	🟢	🔴	🔴	🔴
NLLB	🟢	🔴	🟢	🔴	🔴	🔴	🔴	🔴	🔴	🟢	🟢
Sphere	🟢	🔴	🟢	🔴	🔴	🔴	🔴	🔴	🟢	🔴	🔴

In-Context Learning

- Evolution of Machine Learning
  - Representational Learning – Use supervised training and specialized architecture design
  - Transfer Learning – Pre-trained large models that are tuned for downstream application
  - In-Context Learning – Foundation model used “out of the box” with few-shot learning
- Few-shot learning
  - We can improve performance on a query by providing a number of different examples of the output that we want

In-Context Learning

- Zero-shot
  - The model is generating answers from existing learning – sort of like unsupervised learning
  - The image below is a zero-shot sentiment analysis prompt



The movie's blatant derivativeness is one reason it is so lackluster

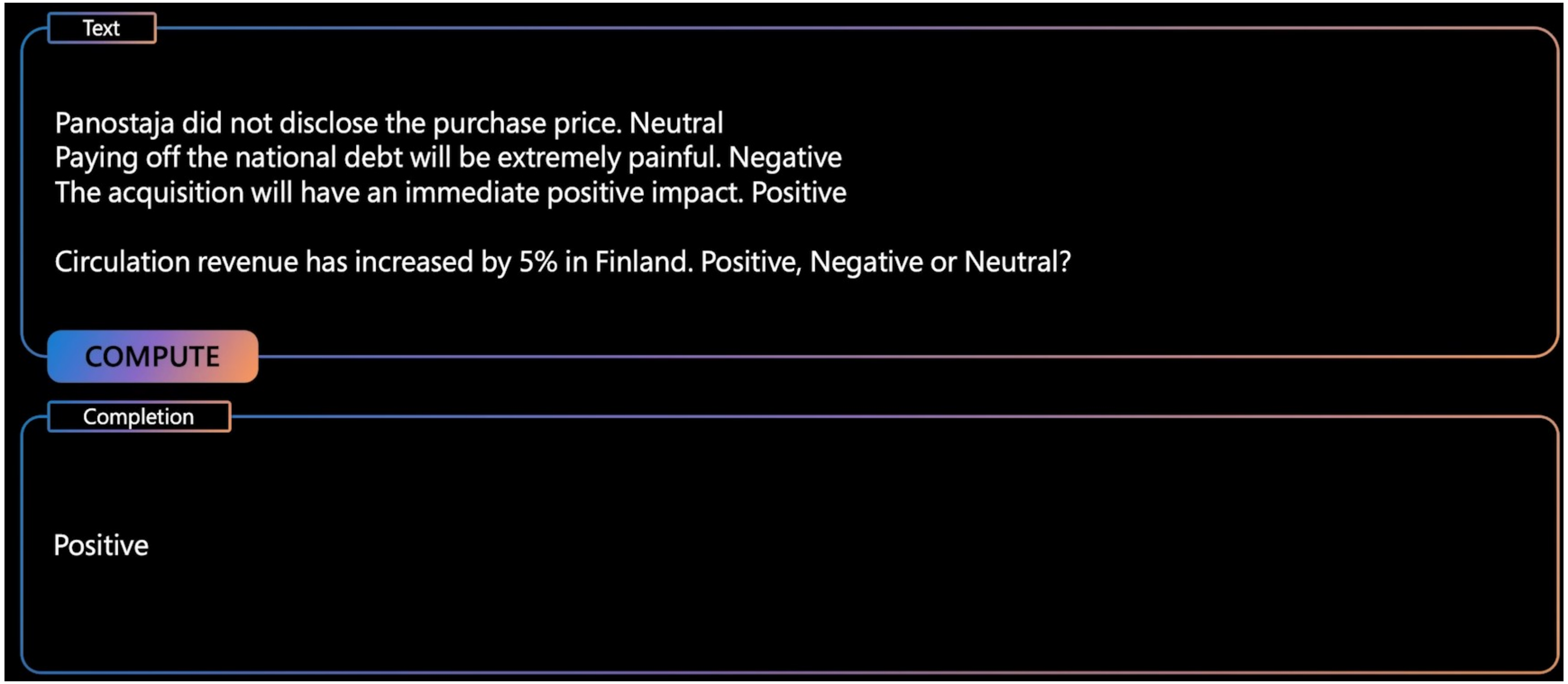
Q: Positive or negative?
A:

Negative

Generated by GPT-3.5 on Azure

In-Context Learning

- Few-shot
  - We provide a couple of examples of the task to be performed



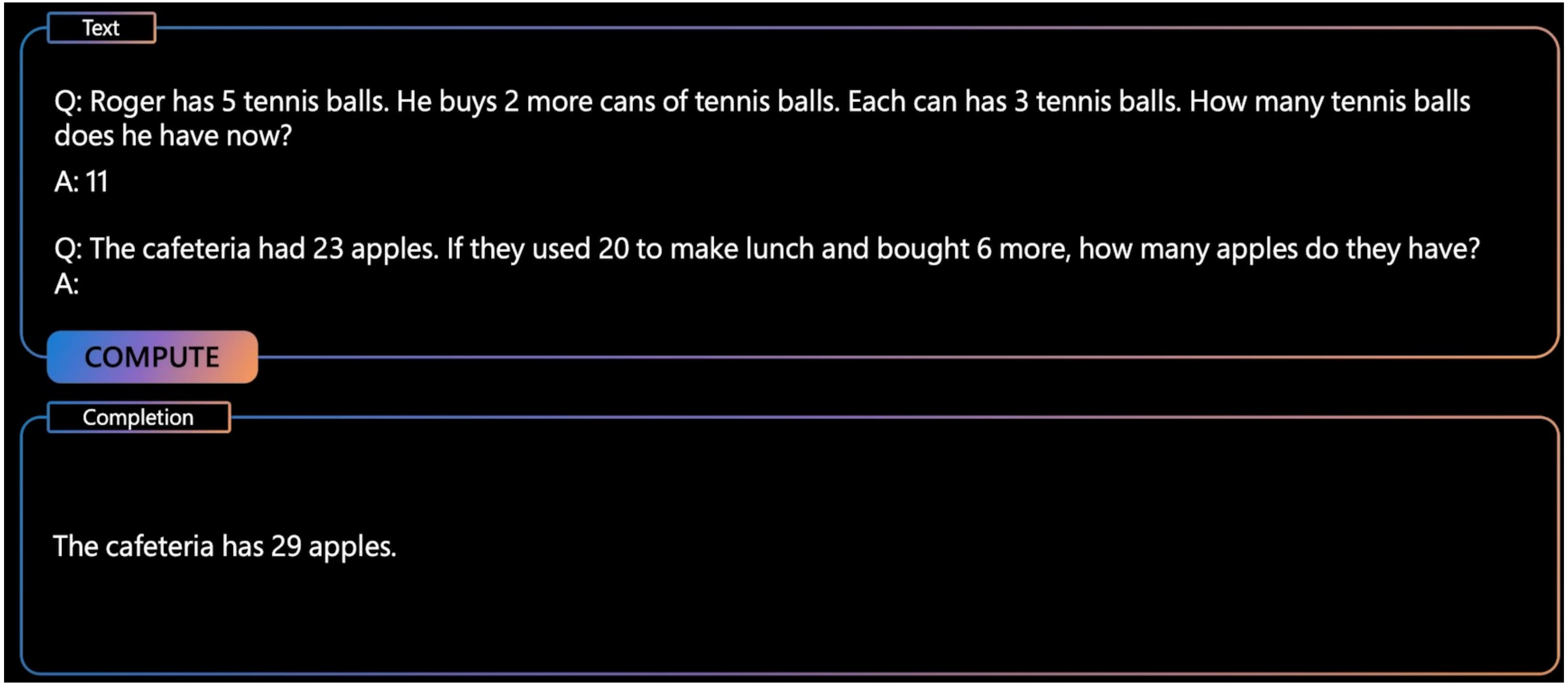
Panostaja did not disclose the purchase price. Neutral
Paying off the national debt will be extremely painful. Negative
The acquisition will have an immediate positive impact. Positive

Circulation revenue has increased by 5% in Finland. Positive, Negative or Neutral?

Positive

In-Context Learning

- Does it actually take place?
  - Consider the following example where the model does not get an arithmetic question right – one of the problems models often have are doing math “word problems”



Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

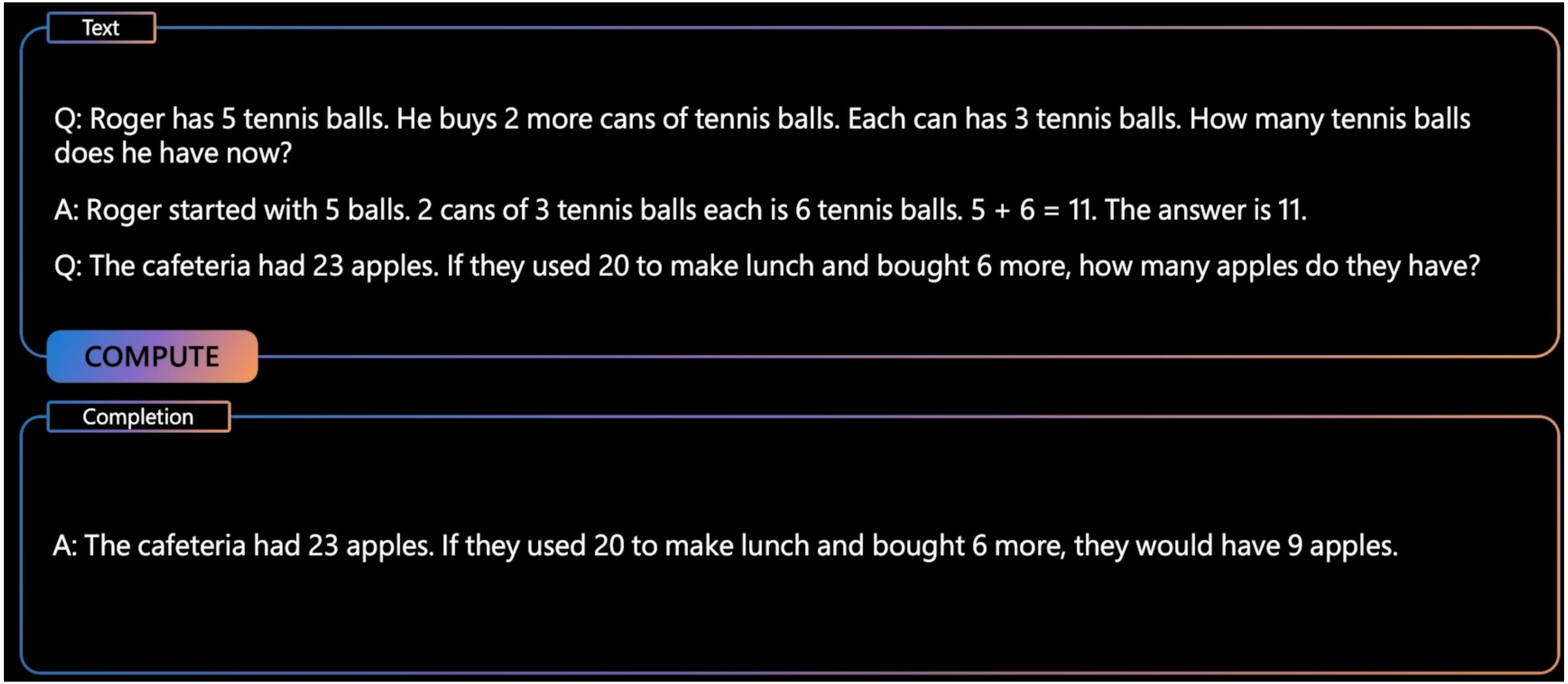
A: 11

Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?
A:

The cafeteria has 29 apples.

In-Context Learning

- Using a chain-of-though prompt improves performance
  - The prompt demonstrates the way the problem should be solved



Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

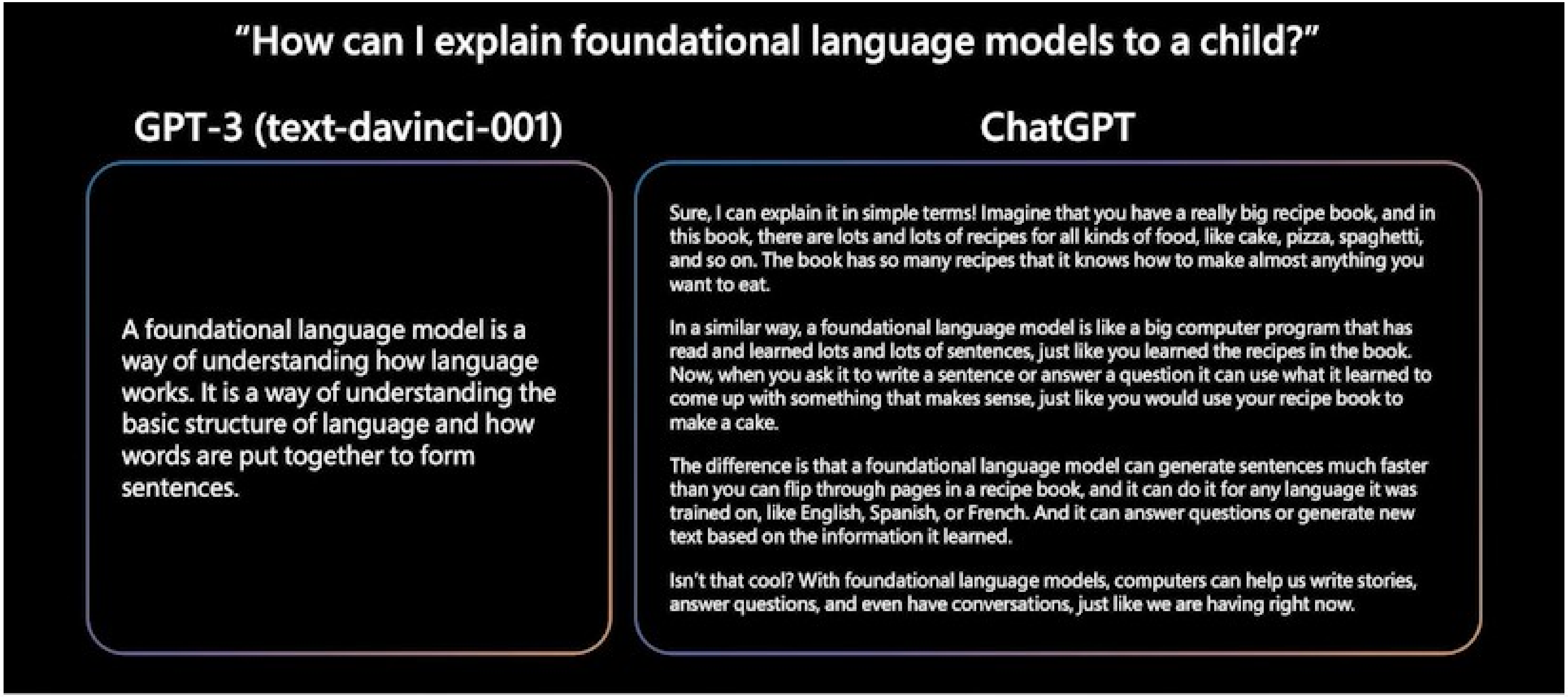
Q: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, how many apples do they have?

A: The cafeteria had 23 apples. If they used 20 to make lunch and bought 6 more, they would have 9 apples.

Improvements in Text

- The model no longer has to be tuned to work in a specific domain
- The model adapts as it responds to the prompts
- The larger the model, the more features it has, the more natural the generated text
  - This means that the model can be used in a wider range of applications
  - It becomes easier to use for general user
  - This produces interactive and conversational interfaces

Improvements in Text



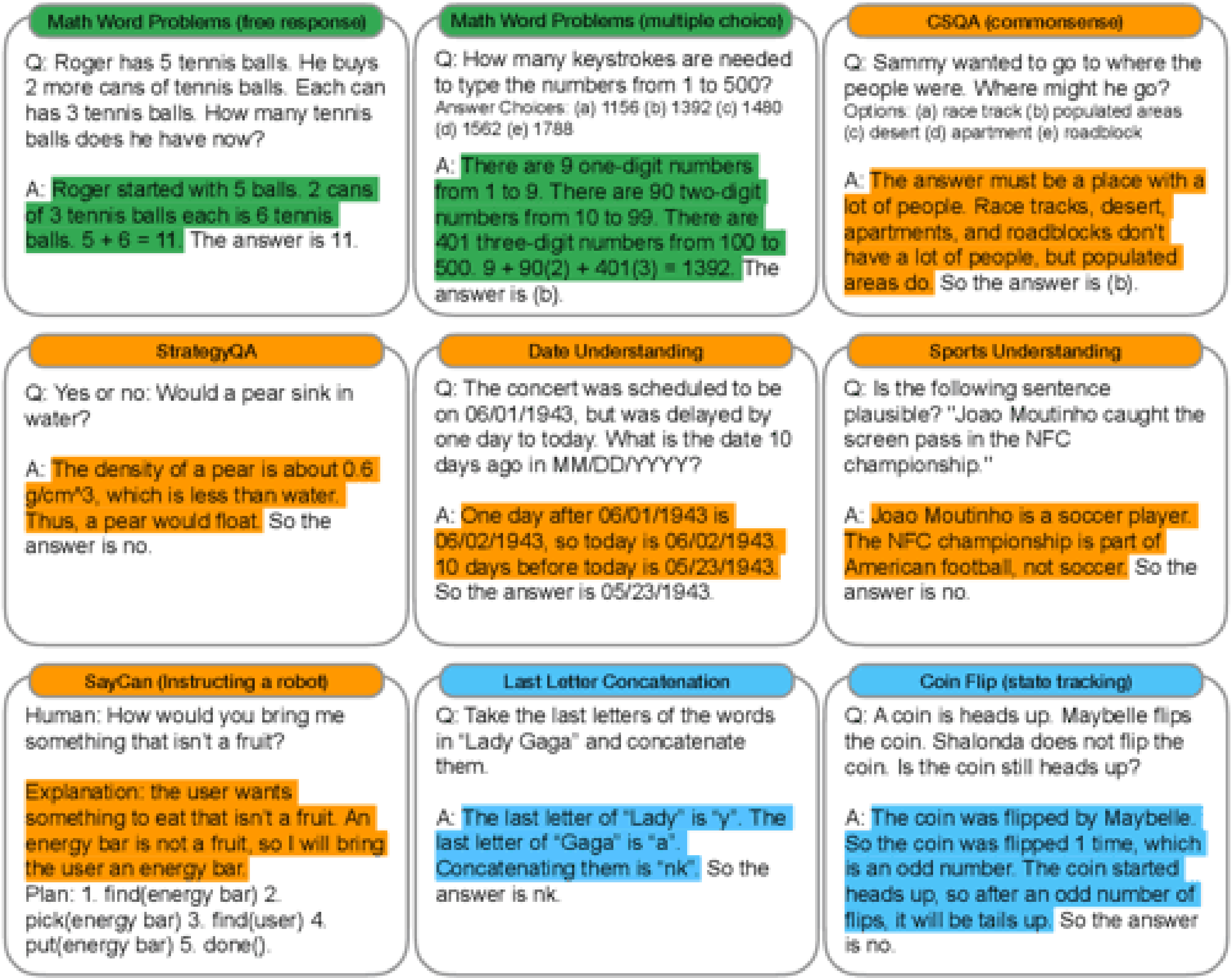
"How can I explain foundational language models to a child?"

GPT-3 (text-davinci-001)	ChatGPT
A foundational language model is a way of understanding how language works. It is a way of understanding the basic structure of language and how words are put together to form sentences.	Sure, I can explain it in simple terms! Imagine that you have a really big recipe book, and in this book, there are lots and lots of recipes for all kinds of food, like cake, pizza, spaghetti, and so on. The book has so many recipes that it knows how to make almost anything you want to eat. In a similar way, a foundational language model is like a big computer program that has read and learned lots and lots of sentences, just like you learned the recipes in the book. Now, when you ask it to write a sentence or answer a question it can use what it learned to come up with something that makes sense, just like you would use your recipe book to make a cake. The difference is that a foundational language model can generate sentences much faster than you can flip through pages in a recipe book, and it can do it for any language it was trained on, like English, Spanish, or French. And it can answer questions or generate new text based on the information it learned. Isn't that cool? With foundational language models, computers can help us write stories, answer questions, and even have conversations, just like we are having right now.

Improvements with Cross Training

- Cross training refers to training as model on multiple domains
- For example, models can be trained on images, text or code, etc
- Cross training refers to training a model on more than one domain
- When we train models on code and language, then language and reasoning performance
- Leverages the idea that there are structural similarities in the underlying domain that can be used to enrich the model

In-Context Learning



Math Word Problems (free response)

Q: Roger has 5 tennis balls. He buys 2 more cans of tennis balls. Each can has 3 tennis balls. How many tennis balls does he have now?

A: Roger started with 5 balls. 2 cans of 3 tennis balls each is 6 tennis balls. 5 + 6 = 11. The answer is 11.

Math Word Problems (multiple choice)

Q: How many keystrokes are needed to type the numbers from 1 to 500? Answer Choices: (a) 1156 (b) 1392 (c) 1480 (d) 1562 (e) 1788

A: There are 9 one-digit numbers from 1 to 9. There are 90 two-digit numbers from 10 to 99. There are 401 three-digit numbers from 100 to 500. 9 + 90(2) + 401(3) = 1392. The answer is (b).

CSQA (commonsense)

Q: Sammy wanted to go to where the people were. Where might he go? Options: (a) race track (b) populated areas (c) desert (d) apartment (e) roadblock

A: The answer must be a place with a lot of people. Race tracks, desert, apartments, and roadblocks don't have a lot of people, but populated areas do. So the answer is (b).

StrategyQA

Q: Yes or no: Would a pear sink in water?

A: The density of a pear is about 0.6 g/cm^3, which is less than water. Thus, a pear would float. So the answer is no.

Date Understanding

Q: The concert was scheduled to be on 06/01/1943, but was delayed by one day to today. What is the date 10 days ago in MM/DD/YYYY?

A: One day after 06/01/1943 is 06/02/1943, so today is 06/02/1943. 10 days before today is 05/23/1943. So the answer is 05/23/1943.

Sports Understanding

Q: Is the following sentence plausible? "Joao Moutinho caught the screen pass in the NFC championship."

A: Joao Moutinho is a soccer player. The NFC championship is part of American football, not soccer. So the answer is no.

SayCan (Instructing a robot)

Human: How would you bring me something that isn't a fruit?

Explanation: the user wants something to eat that isn't a fruit. An energy bar is not a fruit, so I will bring the user an energy bar.
Plan: 1. find(energy bar) 2. pick(energy bar) 3. find(user) 4. put(energy bar) 5. done().

Last Letter Concatenation

Q: Take the last letters of the words in "Lady Gaga" and concatenate them.

A: The last letter of "Lady" is "y". The last letter of "Gaga" is "a". Concatenating them is "nk". So the answer is nk.

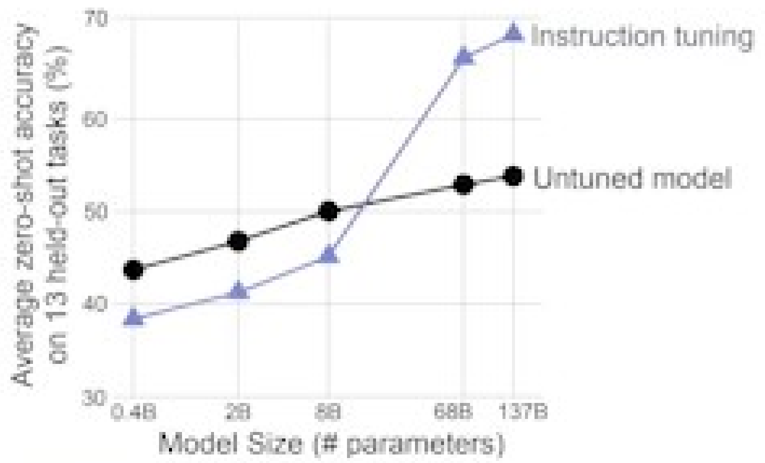
Coin Flip (state tracking)

Q: A coin is heads up. Maybelle flips the coin. Shalonda does not flip the coin. Is the coin still heads up?

A: The coin was flipped by Maybelle. So the coin was flipped 1 time, which is an odd number. The coin started heads up, so after an odd number of flips, it will be tails up. So the answer is no.

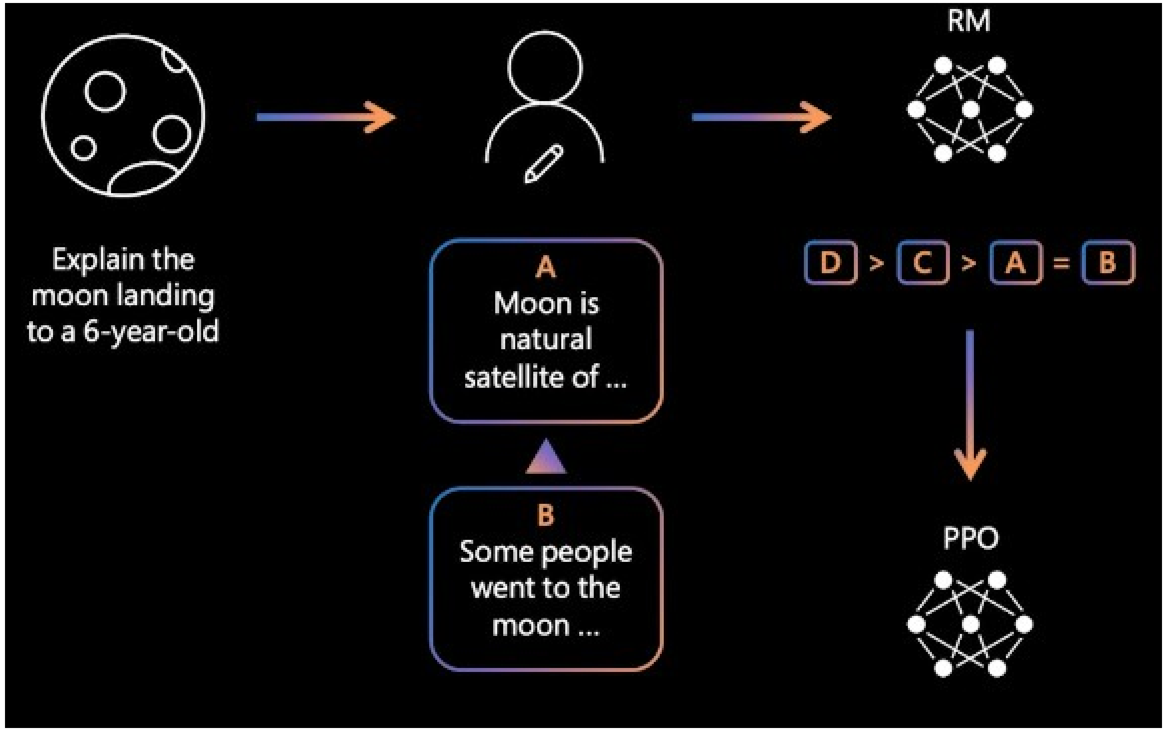
Instruction Tuning

- A form of supervised interactive learning for a trained model
  - A collection of people select a prompt from a selection of prompts
  - Then provide examples of how the prompt should be answered
- Similar to fine tuning a model
  - Instruction tuning improves the performance of zero-shot queries
  - Improves the model’s ability to display reasoning

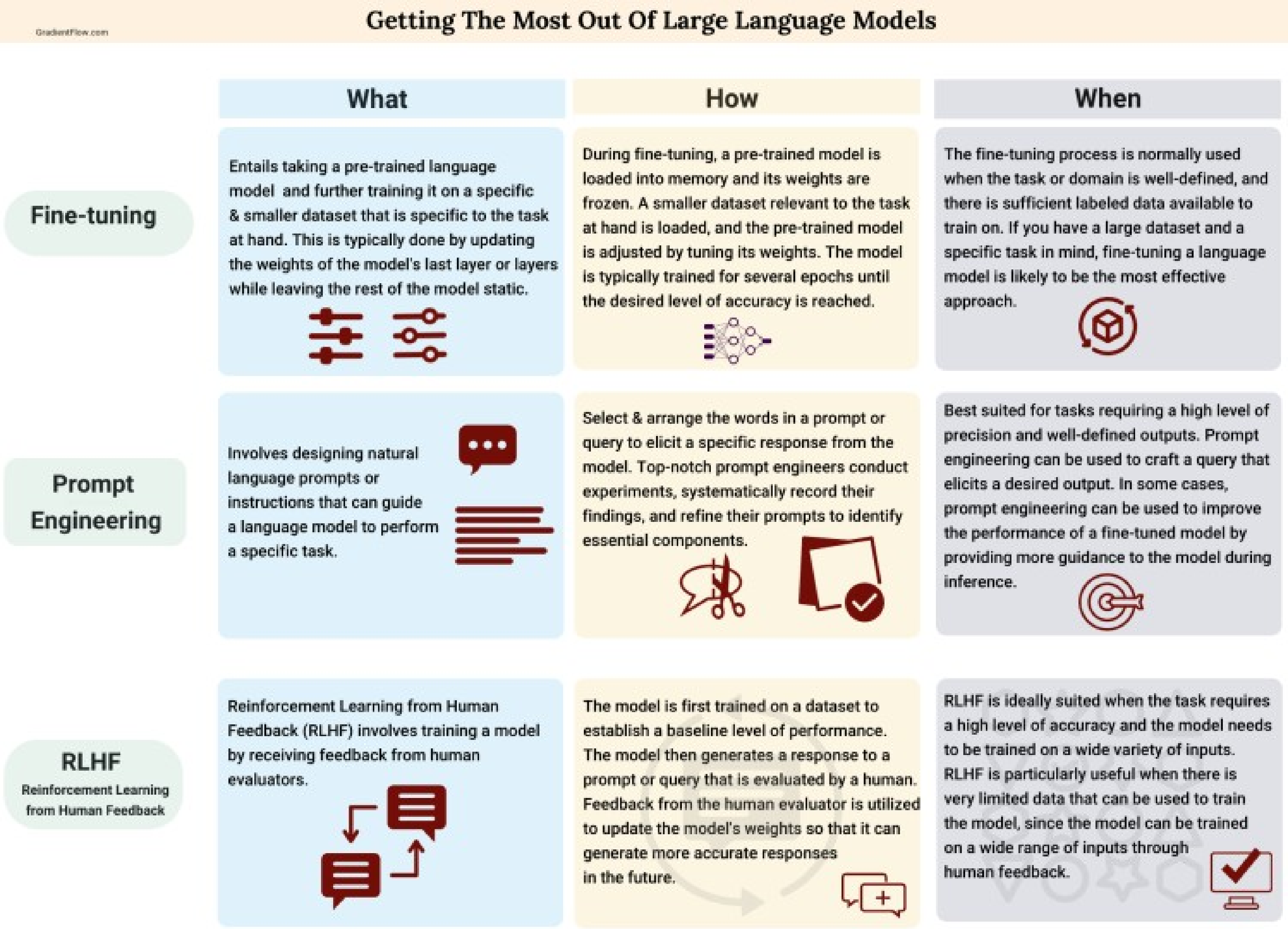


Feedback Tuning

- A form of reinforcement learning
  - Different outputs from a prompt are ranked and the rankings (rewards) are used to prioritize how the model generates responses
  - Note PPO = proximal policy optimization, a reinforcement learning algorithm


Explain the moon landing to a 6-year-old
RM
A
Moon is natural satellite of ...
B
Some people went to the moon ...
D > C > A = B
PPO

Tuning LLMs



GradientFlow.com

Getting The Most Out Of Large Language Models

	What	How	When
Fine-tuning	Entails taking a pre-trained language model and further training it on a specific & smaller dataset that is specific to the task at hand. This is typically done by updating the weights of the model's last layer or layers while leaving the rest of the model static.	During fine-tuning, a pre-trained model is loaded into memory and its weights are frozen. A smaller dataset relevant to the task at hand is loaded, and the pre-trained model is adjusted by tuning its weights. The model is typically trained for several epochs until the desired level of accuracy is reached.	The fine-tuning process is normally used when the task or domain is well-defined, and there is sufficient labeled data available to train on. If you have a large dataset and a specific task in mind, fine-tuning a language model is likely to be the most effective approach.
Prompt Engineering	Involves designing natural language prompts or instructions that can guide a language model to perform a specific task.	Select & arrange the words in a prompt or query to elicit a specific response from the model. Top-notch prompt engineers conduct experiments, systematically record their findings, and refine their prompts to identify essential components.	Best suited for tasks requiring a high level of precision and well-defined outputs. Prompt engineering can be used to craft a query that elicits a desired output. In some cases, prompt engineering can be used to improve the performance of a fine-tuned model by providing more guidance to the model during inference.
RLHF Reinforcement Learning from Human Feedback	Reinforcement Learning from Human Feedback (RLHF) involves training a model by receiving feedback from human evaluators.	The model is first trained on a dataset to establish a baseline level of performance. The model then generates a response to a prompt or query that is evaluated by a human. Feedback from the human evaluator is utilized to update the model's weights so that it can generate more accurate responses in the future.	RLHF is ideally suited when the task requires a high level of accuracy and the model needs to be trained on a wide variety of inputs. RLHF is particularly useful when there is very limited data that can be used to train the model, since the model can be trained on a wide range of inputs through human feedback.

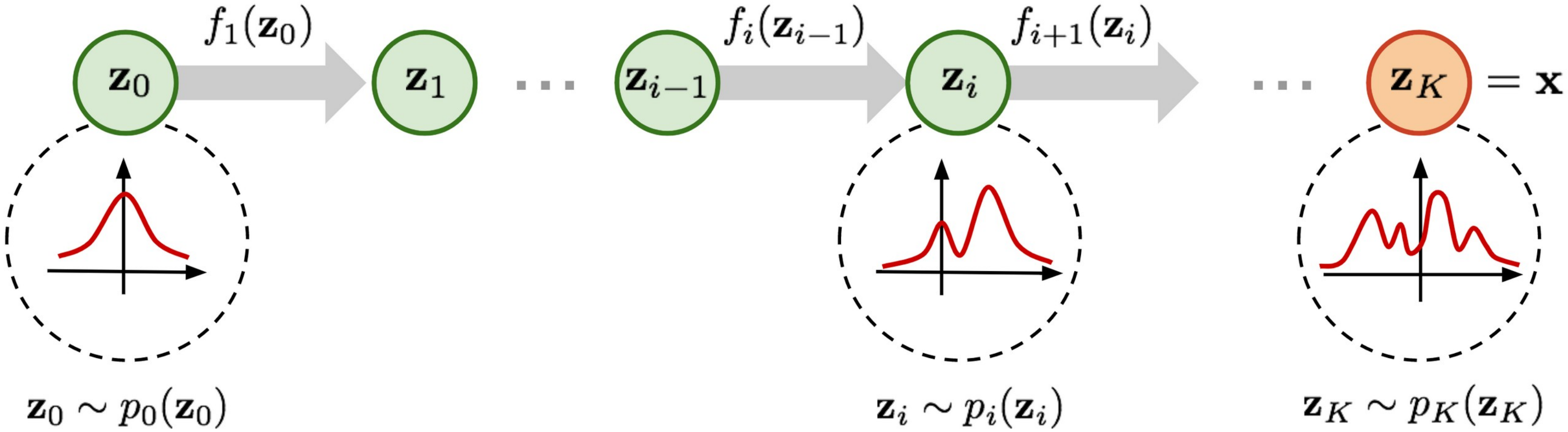
Image Generation Models

- Trained on a large set of images to produce a foundation model for images
  - We have seen an example of this with variational autoencoders
  - We will look at a different approach using GANs in the next module
- Most common models are used to convert text to images
  - Images are generated in response to a text input
  - Trained like a LLM but with data made up of a pair consisting of an image and caption
  - Usually scraped from the web
  - We are still using the same self-supervised learning approach as before
- Image to text applications are generally used to caption images

Flow Models

- Based on the idea of normalizing flows
  - The model flows from a one type of data (random numbers) to an image
- Training a flow model
  - A large training set of images is compiled and the parameters initialized to random variables
  - For each image:
    - The image goes through a forward pass which produces some numeric representation of the image
    - The representation is put through a backward pass to reconstruct the image
  - The reconstructed loss is calculated and the parameters updated by back propagation
- The result of the training is a transformation and reverse transformation
  - New images are generated by running random numbers through the reverse transformation to construct the image

Flow Models



A normalizing flow transforms a simple distribution into a complex one by applying a sequence of invertible transformation functions. Flowing through a chain of transformations, we repeatedly substitute the variable for the new one according to the change of variables theorem and eventually obtain a probability distribution of the final target variable.

Flow Models

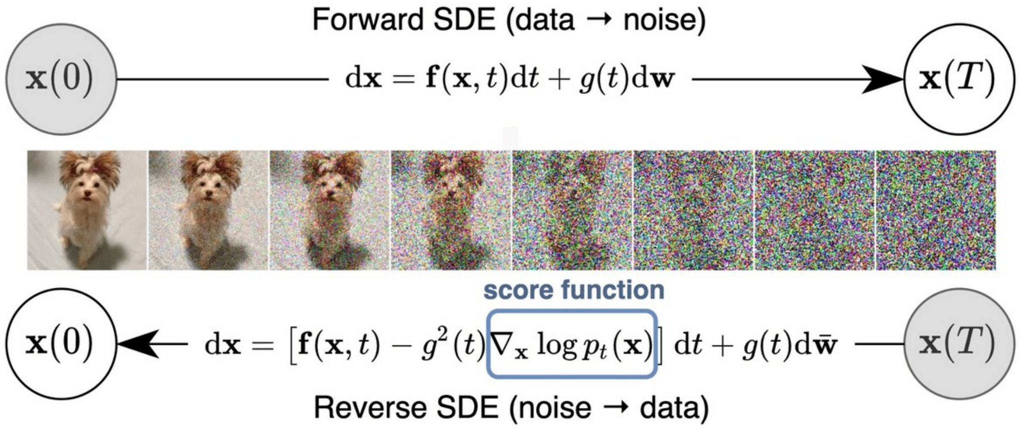
- Advantages
  - Provide a tractable likelihood, which means that you can compute the exact probability of an image occurring
  - Allow for exact and efficient sampling so that latent code (the representation) can quickly generate corresponding images
  - Are inherently invertible, which means images can be mapped back to their latent representations which is useful for image to image translation
  - Stable training using back propagation and gradient descent
  - Multiple layers of invertable transformation can be used to represent intricate patterns

Flow Models

- Disadvantages
  - Struggle to represent extremely high-dimensional and complicated data distributions found in natural images because of the limitations of the architecture
  - Computational complexity grows with the dimensionality of the data, making them less efficient for high-resolution images
  - Designing appropriate transformations and architectures requires careful consideration to ensure efficient and expressive learning
  - Designing an effective and compact latent space is challenging which affects the quality of the generated images
  - Struggle with generating diverse samples in scenarios where the data distribution is highly multimodal which can lead to mode collapse, where the model only generates a limited set of similar images

Diffusion Models

- Similar to other models in reducing an image to a representation and having a reconstruction process
  - Starts with a target image
  - Gradually adds noise effectively 'diffusing' the image into noise
  - Learns to reverse the diffusion process
  - Generative goal is to start with a simple random noise image and then progressively refine it until it becomes a sample from the same distribution as the training images


Forward SDE (data → noise)
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = \mathbf{f}(\mathbf{x}, t)\mathrm{d}t + g(t)\mathrm{d}\mathbf{w}$
$\mathbf{x}(T)$
score function
$\mathbf{x}(0)$
$\mathrm{d}\mathbf{x} = \left[\mathbf{f}(\mathbf{x}, t) - g^2(t)\boxed{\nabla_{\mathbf{x}} \log p_t(\mathbf{x})}\right] \mathrm{d}t + g(t)\mathrm{d}\bar{\mathbf{w}}$
$\mathbf{x}(T)$
Reverse SDE (noise → data)

Diffusion Models

- Training Process
  - The forward process is done in steps
  - Each steps add some noise to the image via a diffusion equation
  - Continues until the image is just noise
  - The backward process starts with a random noise image and reverses the noising steps to recover the target image
  - A neural net is used at each step of the backward process to learn the de-noising function
- To generate images
  - Start with random noise and denoise to generate a new image
  - The new image should wind up belonging to the original image probability distribution

Diffusion Models

- Advantages
  - Ability to generate high-resolution, visually appealing images with fine details and coherent structures by iteratively refining the generated images
  - Can be used for image completion and inpainting tasks, where missing or corrupted parts of an image are filled in accurately based on context - image restoration and editing applications
  - Understand contextual information about images which allows generation of images that are more contextually relevant and semantically meaningful
  - Tend to exhibit fewer mode collapse issues compared to other generative models

Diffusion Models

- Disadvantages
  - Can be computationally expensive and time-consuming
  - Lack interpretability making it difficult to diagnose and fix potential issues
  - Complex optimization techniques can make training and fine-tuning the models more challenging
  - Memory requirements of diffusion models increase with the size of the model and the resolution of the images being generated
  - Struggle with understanding very long-range dependencies in complex scenes
  - Involves significant data and computational resources raising concerns about privacy and environmental impact

Transformers

- The development of transformers was one of the three drivers of large foundation model
  - A specific architectural model that is based on using encoders, decoders and other features of neural nets
  - Originally had a revolutionary impact on language models
  - Key innovation "attention mechanism"
    - Weighs the importance of different parts of the input data
    - Lets the model decide which parts of the data to focus on at each step of the computation (what to pay attention to)
    - Relies on contextual information during processing
  - Originally described in the paper *All you need is attention* by Vaswani et al.

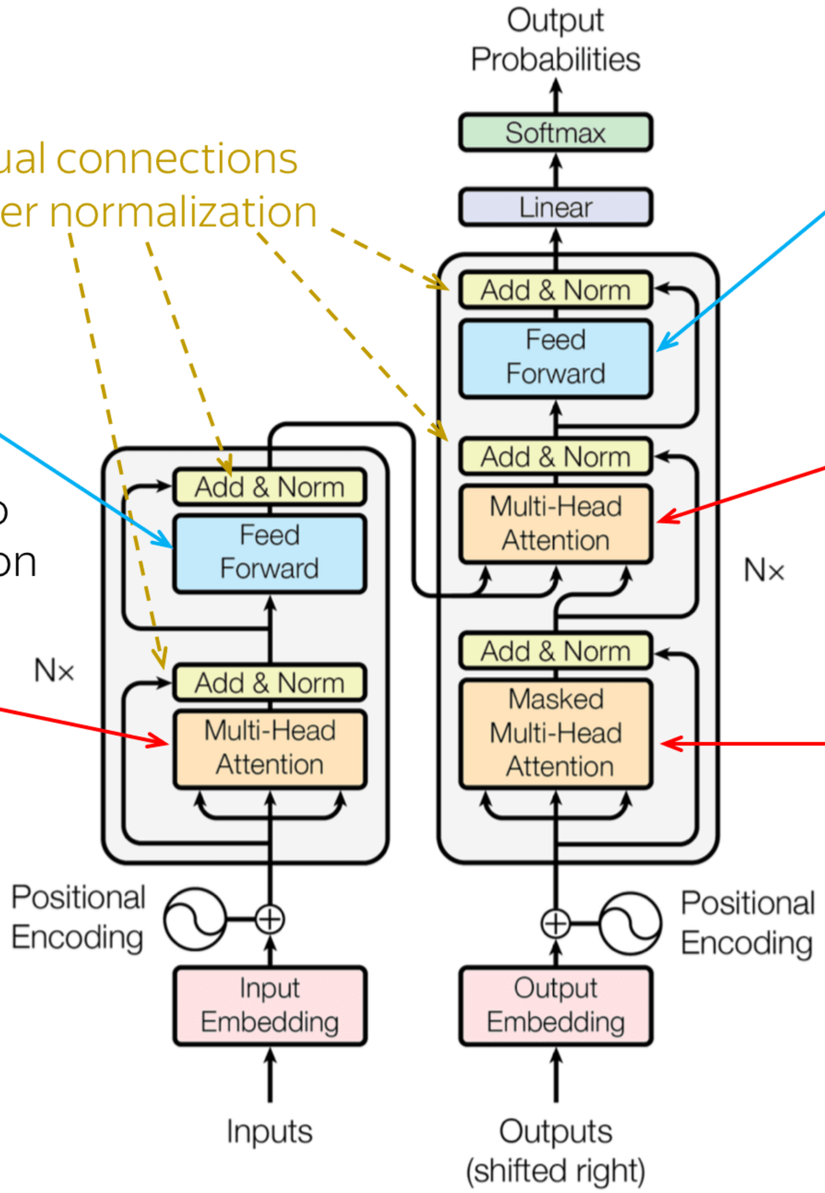
Transformers

Residual connections
and layer normalization

Feed-forward network:
after taking information from
other tokens, take a moment to
think and process this information

Encoder self-attention:
tokens look at each other

queries, keys, values
are computed from
encoder states



Feed-forward network:
after taking information from
other tokens, take a moment to
think and process this information

Decoder-encoder attention:
target token looks at the source

queries – from decoder states; keys
and values from encoder states

Decoder self-attention (masked):
tokens look at the previous tokens

queries, keys, values are computed
from decoder states

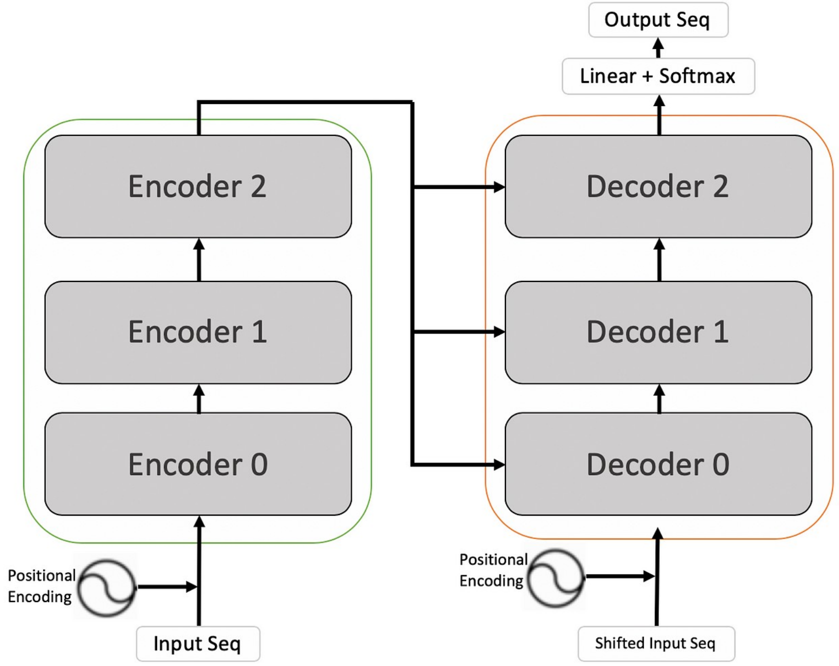
Image Credit: https://stats.stackexchange.com/questions/512242/why-does-transformer-has-such-a-complex-architecture

How Transformers Work

- Input Embeddings:
  - The input text is transformed into numerical representations
  - Each word is converted into a fixed-size vector called a word embedding
  - The same process we saw in the section on autoencoding
- Positional Encoding:
  - Positional encoding is added to the word embeddings to indicate their positions in the input text
- Attention Mechanism
  - Attention allows the model to focus on relevant parts of the input sequence when processing each word
  - Attention scores are calculated between each word and all other words in the input sequence, highlighting the words that are most important for understanding the current word

How Transformers Work

- Multi-Head Attention:
  - Captures different types of dependencies and relationships in the input text
  - Performs attention computations multiple times with separate sets of parameters and then combines the results.
  - Each "head" looks for different patterns and features in the data
- Encoder and Decoder Stacks:
  - Architecture consists of an encoder and a decoder stack
  - The output of the encoder is used to feed the decoder stack, which generates the output



How Transformers Work

- Encoder Stack:
  - Composed of multiple layers
  - Each layer contains a self-attention mechanism followed by feedforward neural networks
  - Self-attention allows the model to consider all words in the input sequence simultaneously
  - The feedforward networks capture additional context and relationships.
- Decoder Stack:
  - Also contains multiple layers including self-attention and feedforward layers
  - Also includes an additional attention mechanism called encoder-decoder attention
  - Allows the decoder to consider the relevant information from the encoder stack when generating the output – i.e. what information from the encoder should be paid attention to

How Transformers Work

- Output Layer:
  - Produces the output predictions based on the processed information from the encoder and decoder
  - For example, in language translation, it predicts the next word in the translated sentence
- Training:
  - During training, the model is provided with input-output pairs
  - Learns to adjust its parameters (weights) to minimize the difference between its predictions and the ground truth outputs
  - Recall that we can do this with self supervised learning
- Inference:
  - Can be used to generate predictions for new input text
  - Decoder generates the output step by step, producing one word at a time until a special end-of-sequence token is generated or a maximum output length is reached

Transformer Advantages

- Parallel Processing
  - Allows for parallel processing of input sequences so that all words in a sentence can be processed simultaneously
  - Allows for parallel compute on multiple devices
- Long-Range Dependencies
  - Attention mechanism captures long-range dependencies in the input text effectively in large context windows
- Efficient Attention:
  - Multi-head attention mechanism focuses on different parts of the input sequence simultaneously
  - Allows the model to selectively attend to the most relevant information, reducing redundancy and improving efficiency

Transformer Advantages

- Scalability
  - Can be efficiently scaled to handle large datasets and more complex tasks
- Transfer Learning
  - Can be fine-tuned on specific downstream tasks with comparatively little data
  - Useful for tasks with limited labeled data
- Universal Representation
  - Remarkable performance across a variety of NLP tasks, demonstrating their ability to learn universal language representations
- No Recurrence
  - Do not rely on recurrent connections, which makes them more computationally efficient during training and allows for better parallelization

Transformer Disadvantages

- Memory Requirements
  - Require large amounts of memory
  - can make training and deployment challenging, especially on resource-constrained devices.
- Training Time
  - Can be time-consuming and computationally expensive, requiring access to high-end hardware
- Limited Interpretability
  - lack interpretability so that understanding why a specific prediction is made can be difficult, making it harder to diagnose model errors
- Attention Overhead
  - Introduces additional computation overhead, especially for long sequences, which may impact the model's efficiency

Transformer Disadvantages

- Bias Amplification
  - Can learn and amplify biases present in the training data, leading to biased outputs and potential ethical concerns
- Data Dependency
  - Heavily relies on the availability and quality of training data
  - Insufficient or biased data can lead to suboptimal results.
- Sequential Information Loss:
  - Can loose sequential information, which can be important for certain tasks that require strong temporal context understanding.

End of Module