AY 2024/2025 S1

EE6427 Video Signal Processing

Part 6 Foundation Models & Generative AI

Dr Yap Kim Hui

Room: S2-B2b-53
Tel: 6790 4339
Email: ekhyap@ntu.edu.sg

References

- Rishi Bommasani et. al. On the Opportunities and Risks of Foundation Models, arXiv:2108.07258.
- CMU Course11-785 Introduction to Deep Learning.
- OpenAI, https://openai.com/
- Pranab Sahoo et. al. A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications, arXiv:2402.07927.
- Stanford Lecture Notes, CS231n: Convolutional Neural Networks for Visual Recognition.

Part 6 Outline

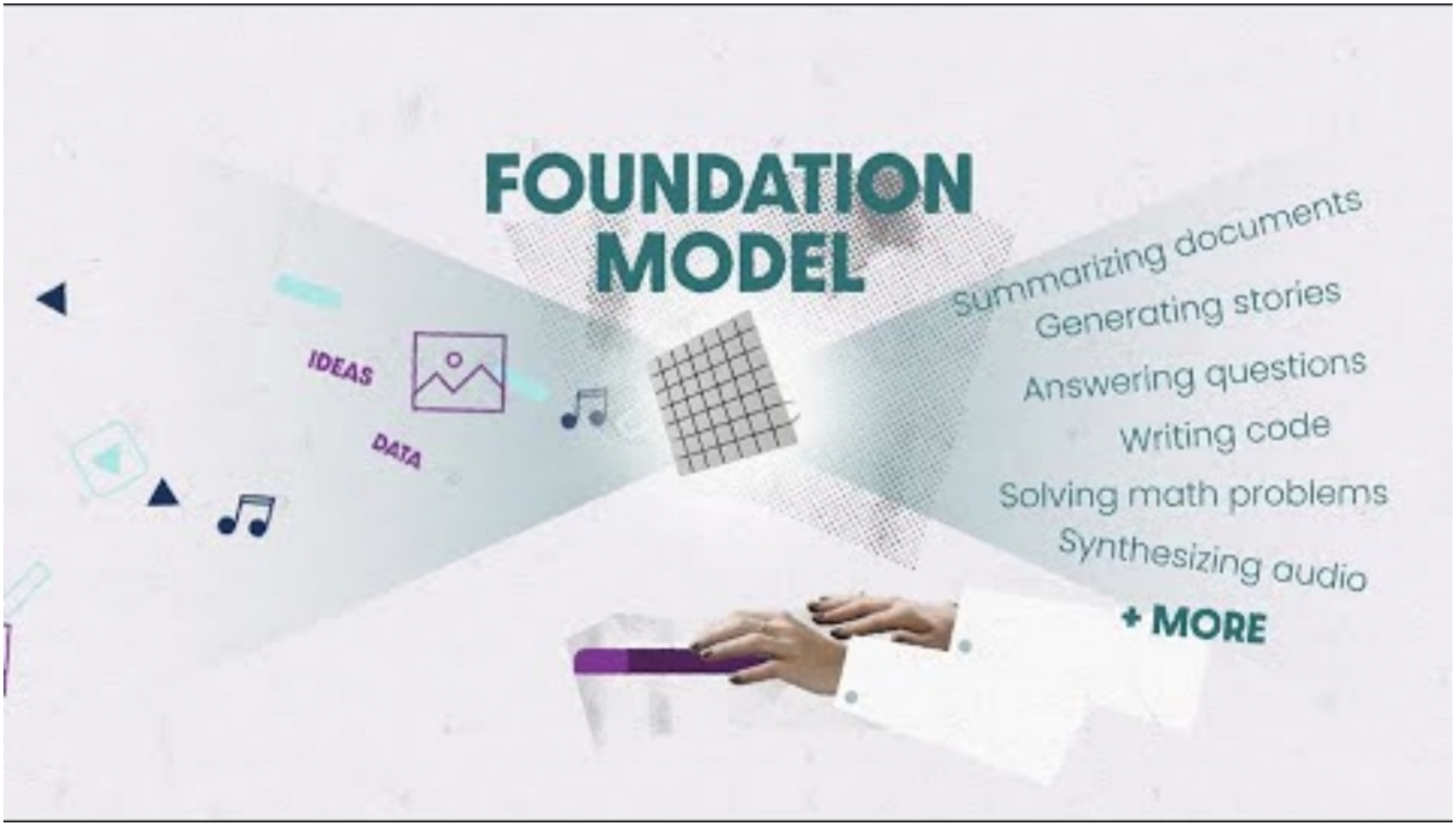
- The part will cover the following topics:
  - Section I: Foundation Models (FMs)
  - Section II: Large Language Models (LLMs)

Section I
Foundation Models (FMs)

Section I Overview

- The section will cover the following topics:
  - Current State of AI
  - Basics of Foundation Models (FMs)
  - Adapting / Finetuning FMs

Foundation Models

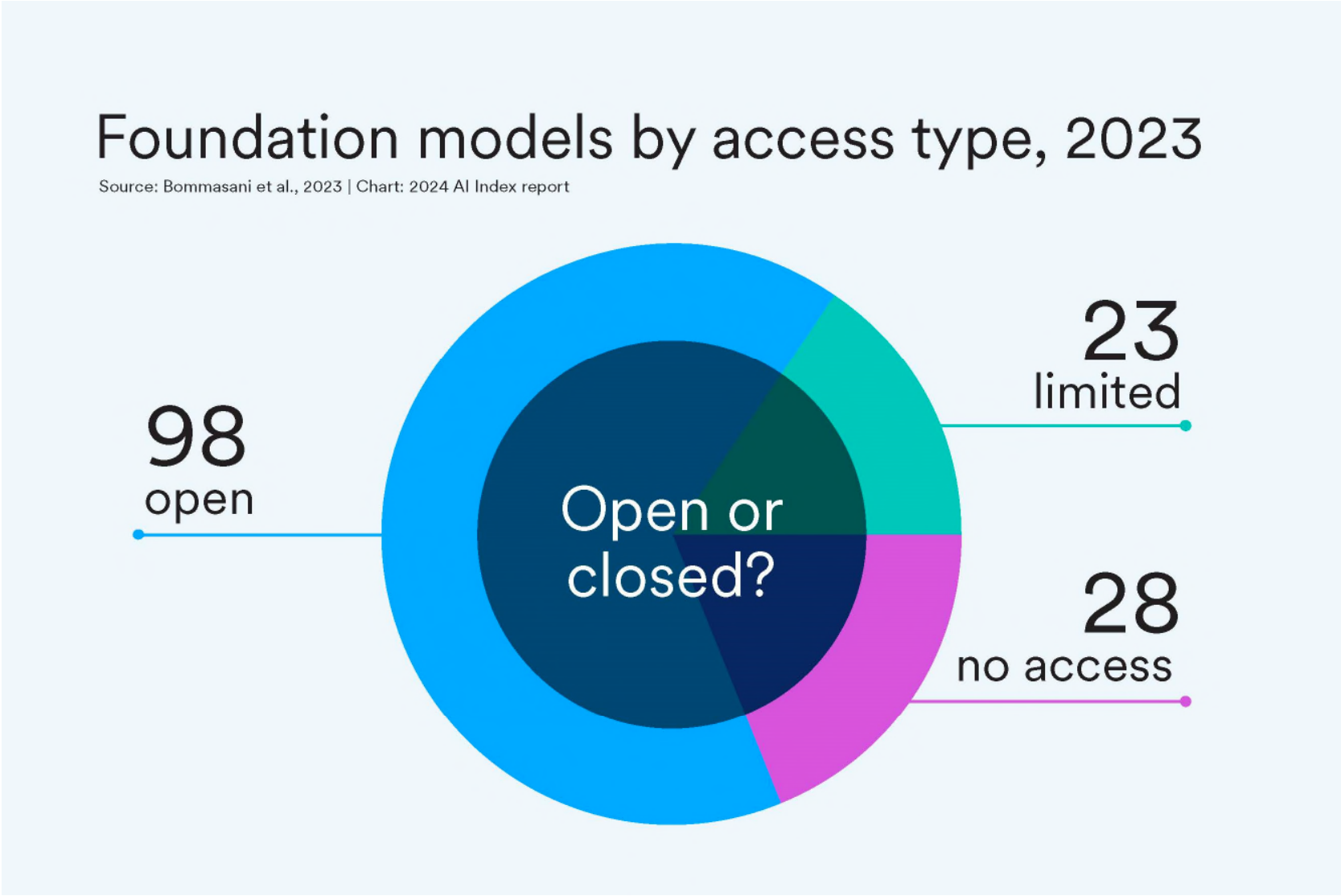


Source: Stanford HAI

Current States of AI

Access Types

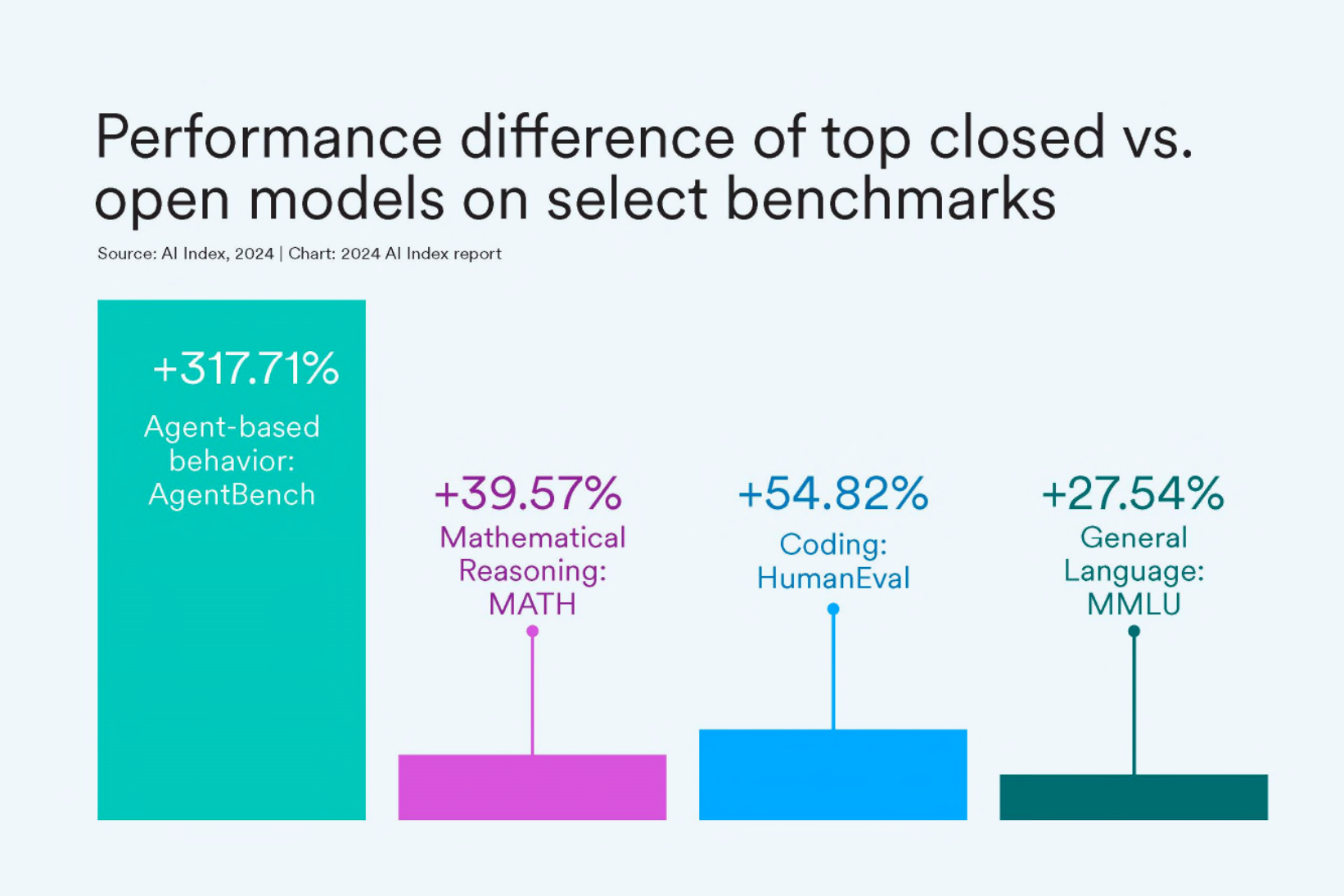
- Moving toward open-sourced.



Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

Performance Comparison

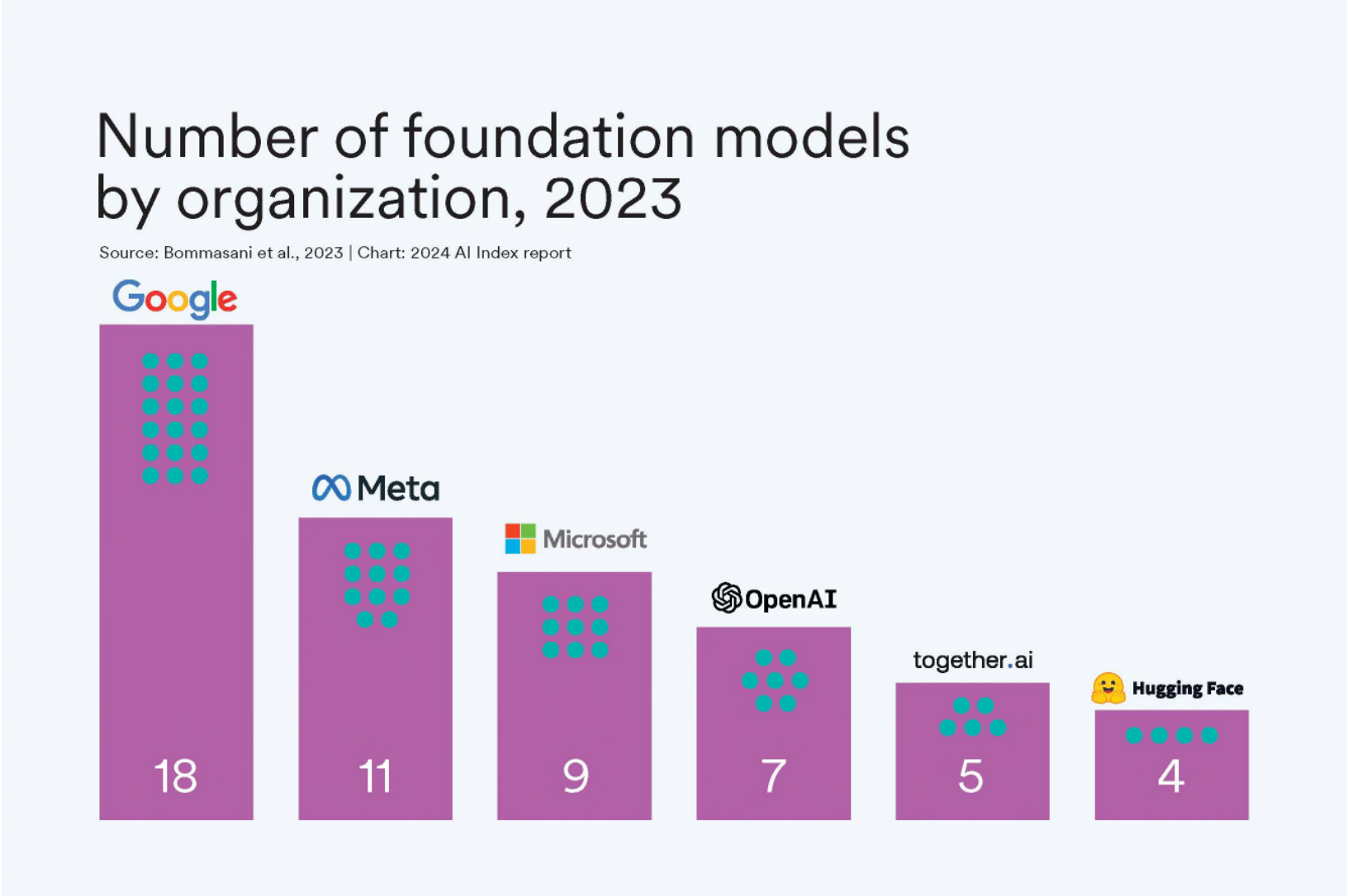
- Closed-source models outperform open-sourced models.



Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

Key Players

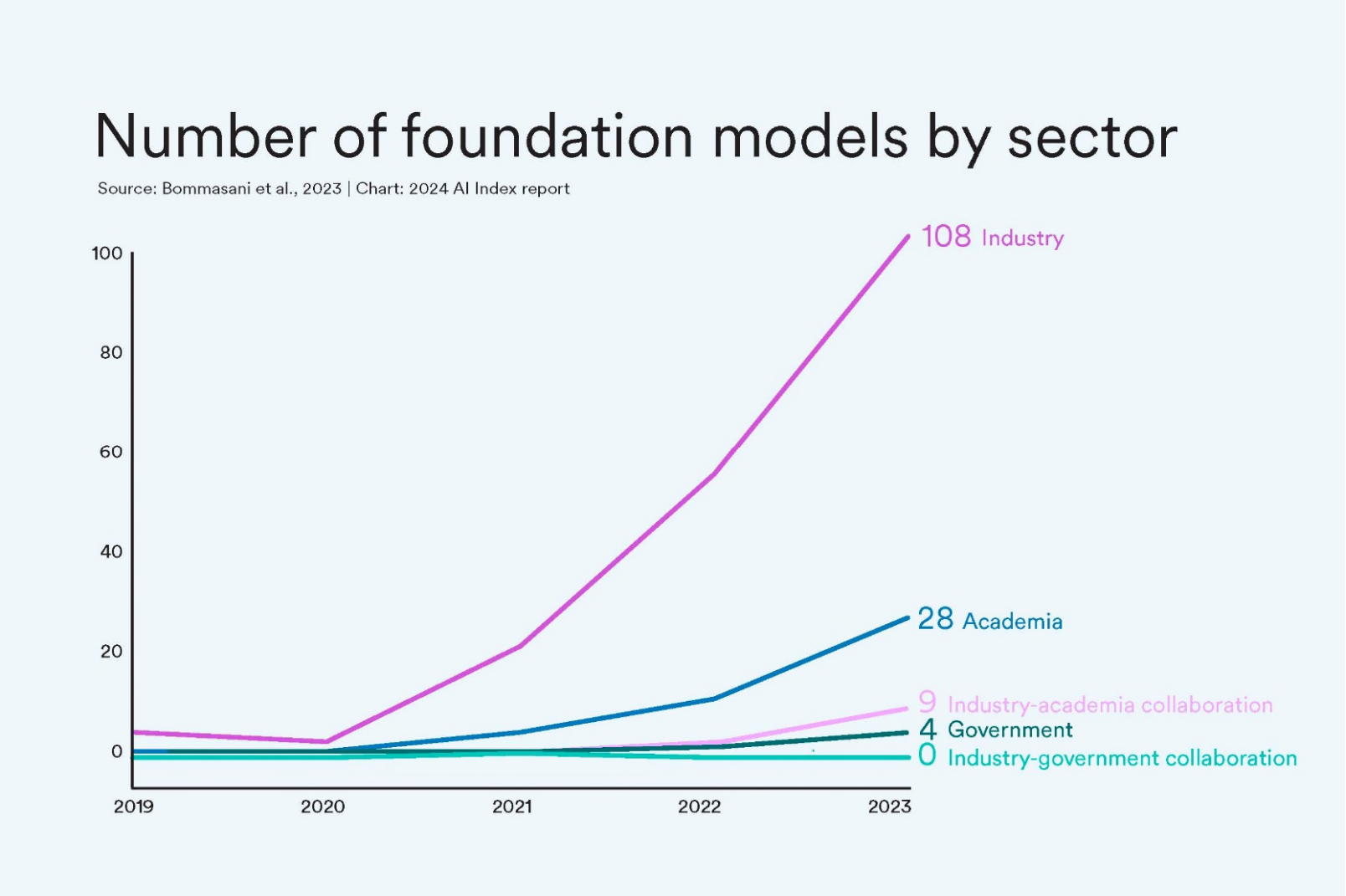
- Industry dominates FMs.



Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

Key Players

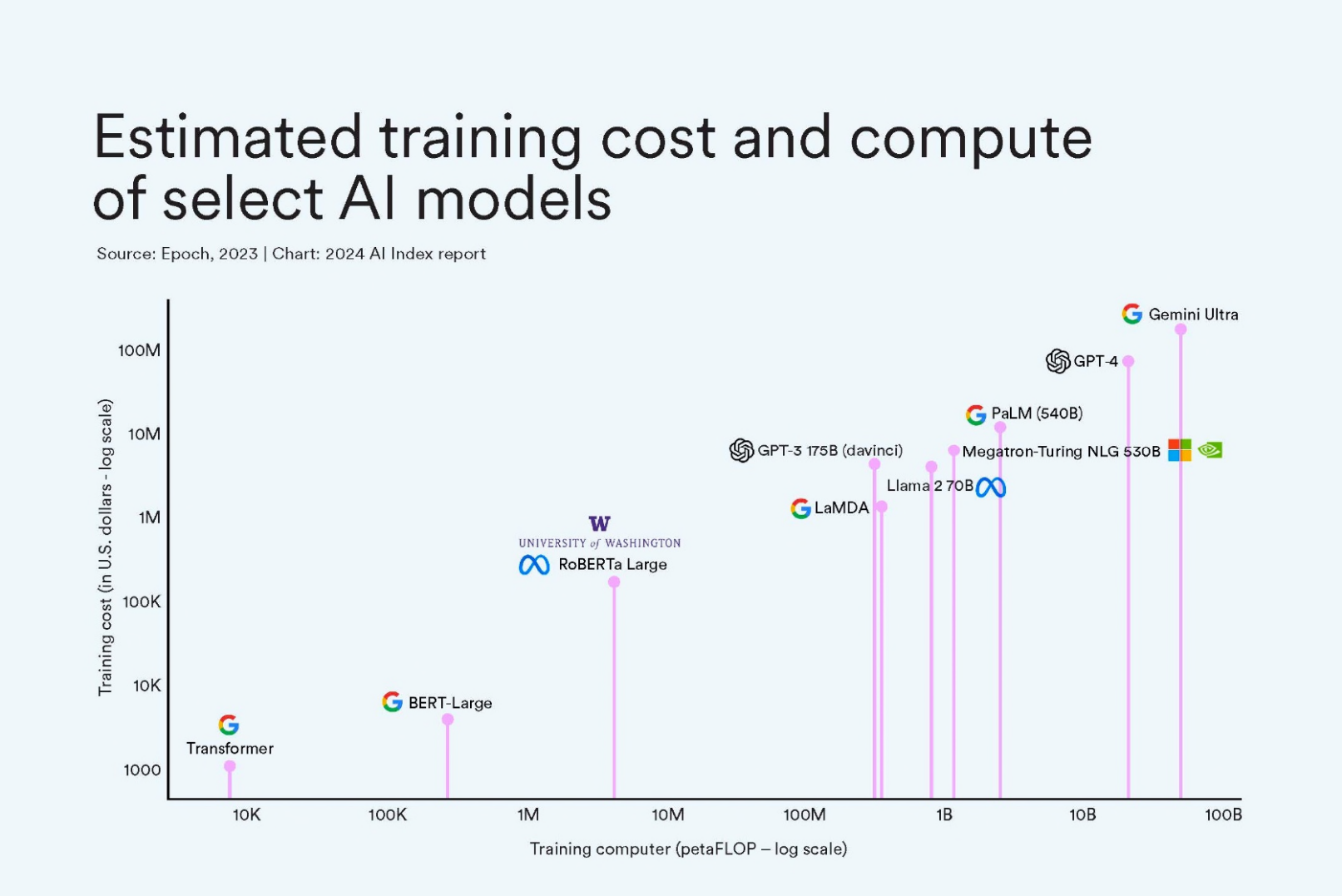
- Industry dominates Foundation Models.



Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

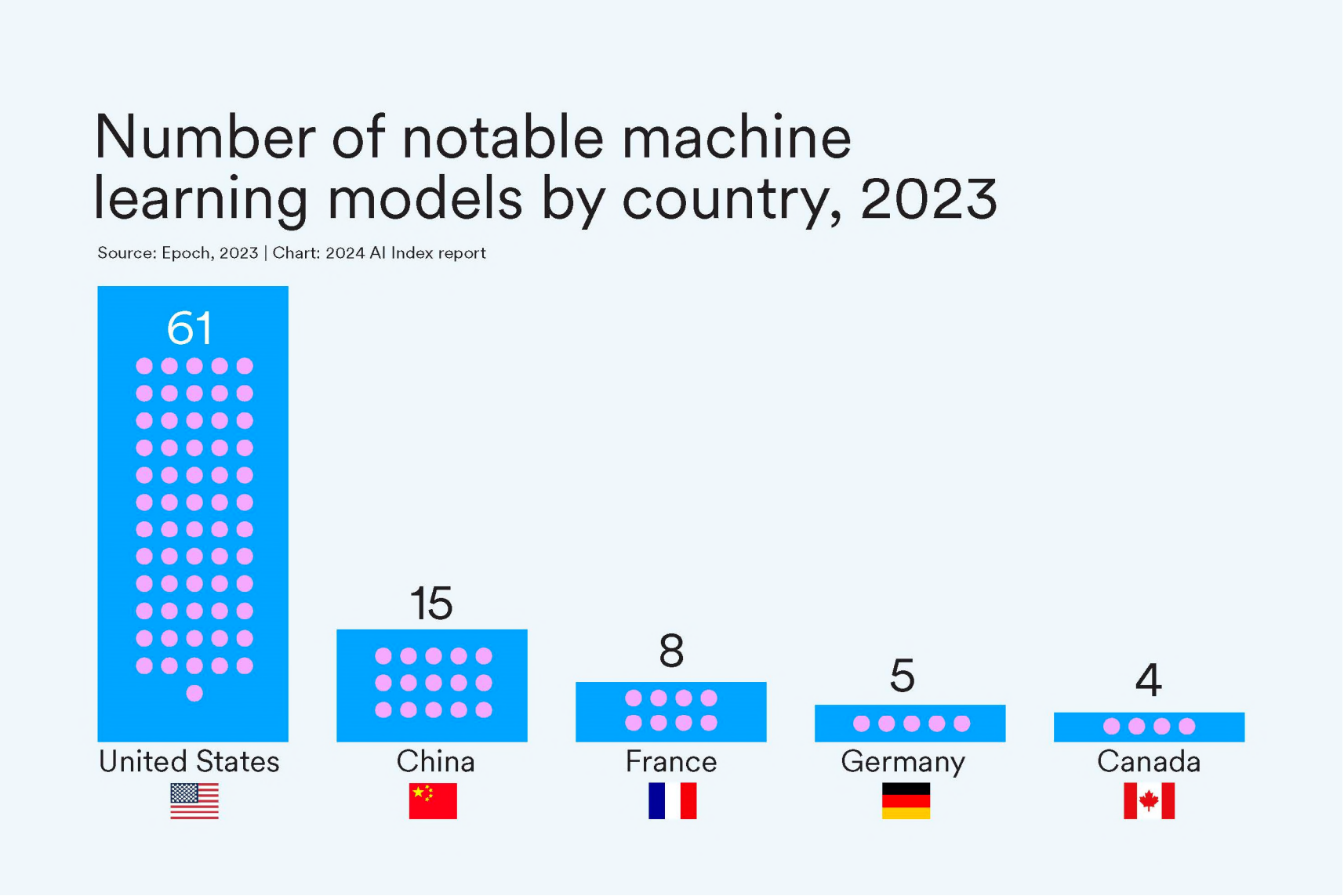
Training Cost

- High cost in training FMs.



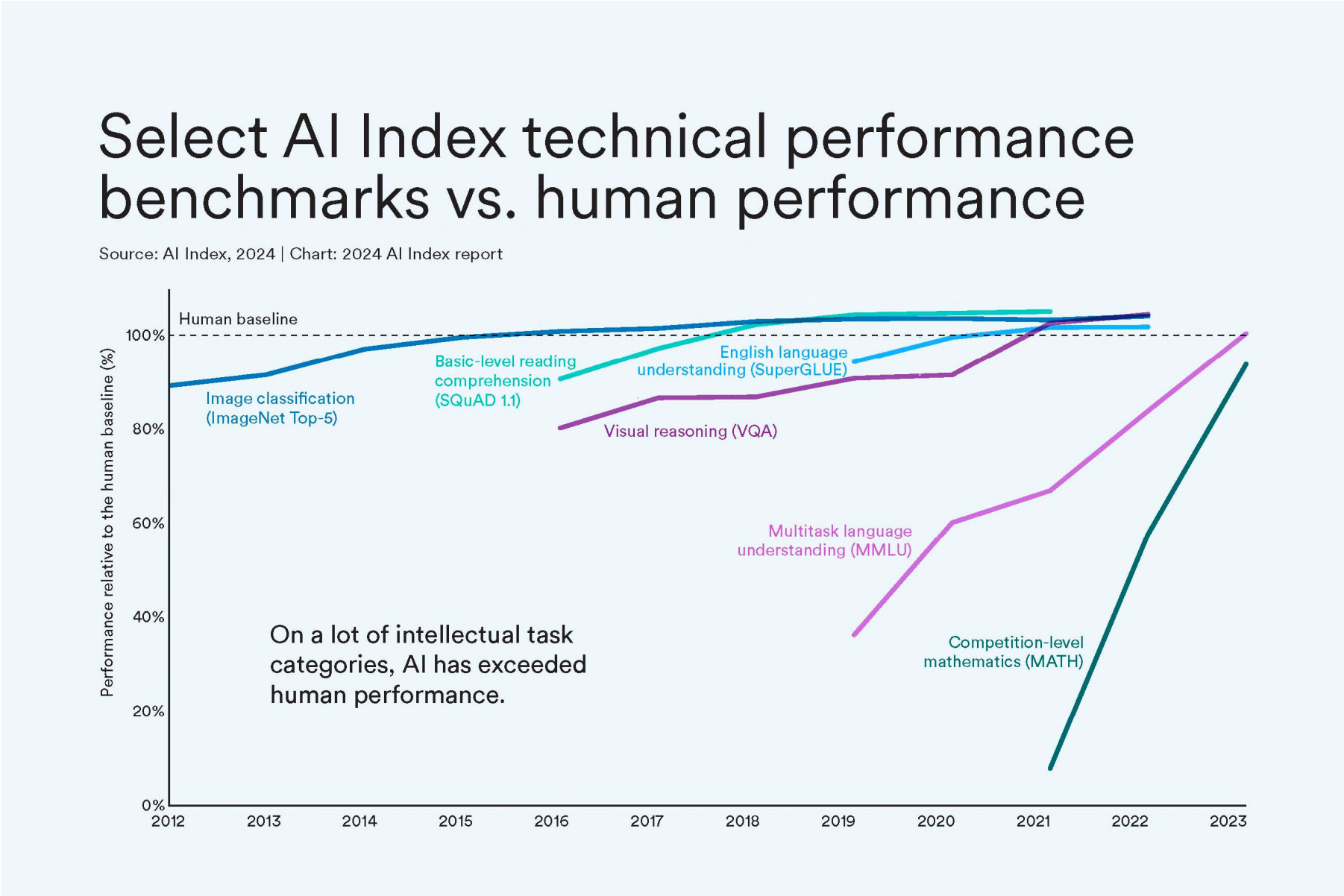
Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

Well-known AI Models



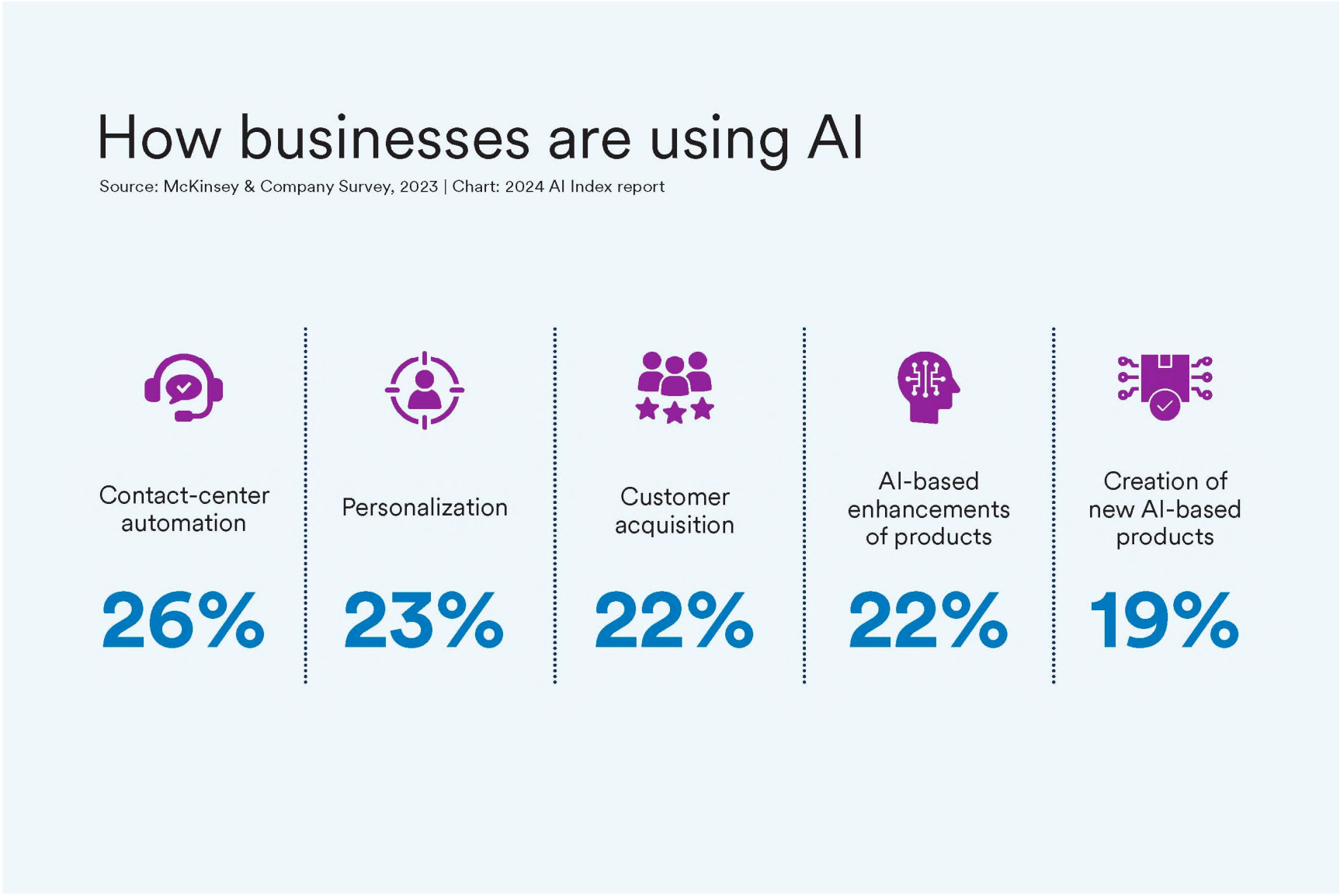
Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

Performance: AI vs Human



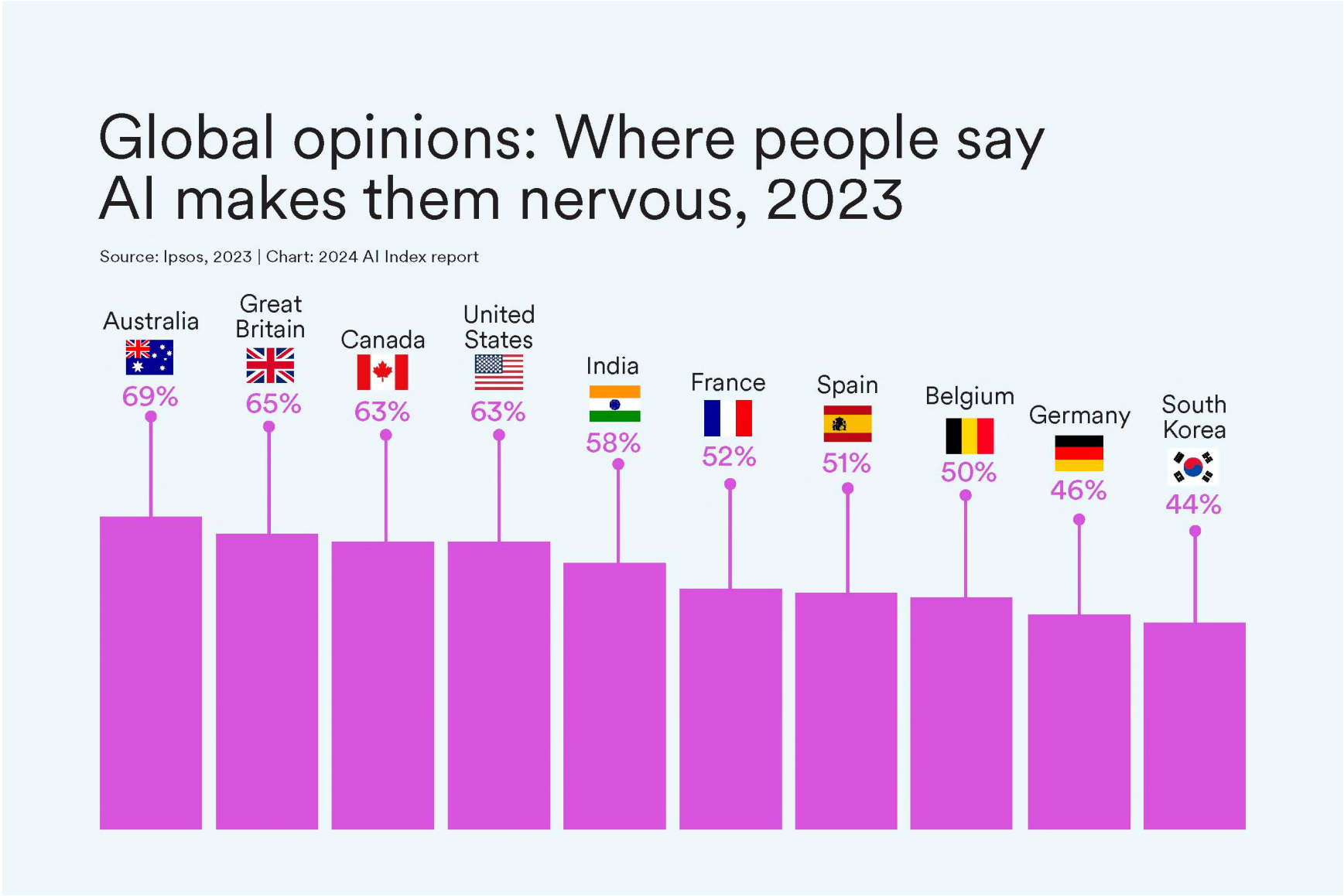
Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

Business in AI



Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

Opinions on AI



Source: https://hai.stanford.edu/news/ai-index-state-ai-13-charts

Basics of Foundation Models (FMs)

What are Foundation Models (FMs)?

- FMs are models that are trained on large-scale broad data that can be adapted (finetuned) to a wide range of downstream tasks / applications.
- Examples: Large Language Models (e.g., GPT), Vision-Language Models (e.g., CLIP), Multimodal LLM (GPT-4o, LLaVA).

Key Ideas of FMs

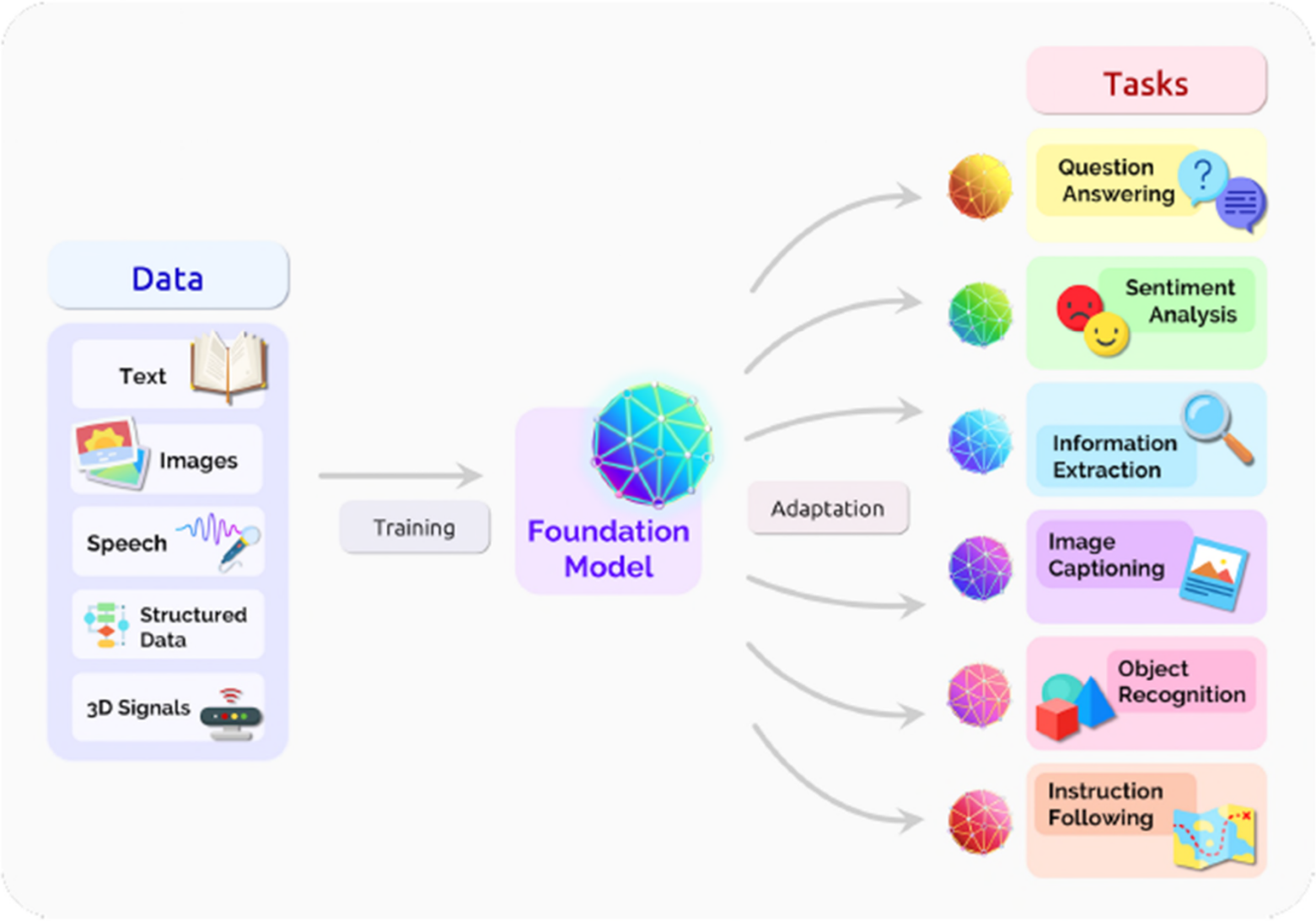


Fig. 2. A foundation model can centralize the information from all the data from various modalities. This one model can then be adapted to a wide range of downstream tasks.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Key Ideas of FMs

- A new paradigm of AI based on large-scale pre-training and downstream adaptation.
- Pretraining:
  - A model is trained on a large-scale diverse date.
  - Typically use self-supervised learning to alleviate expensive data collection and annotation.
- Adaptation / fine-tuning:
  - Pre-trained models are adapted for subsequent downstream tasks.

Different Stages of FMs

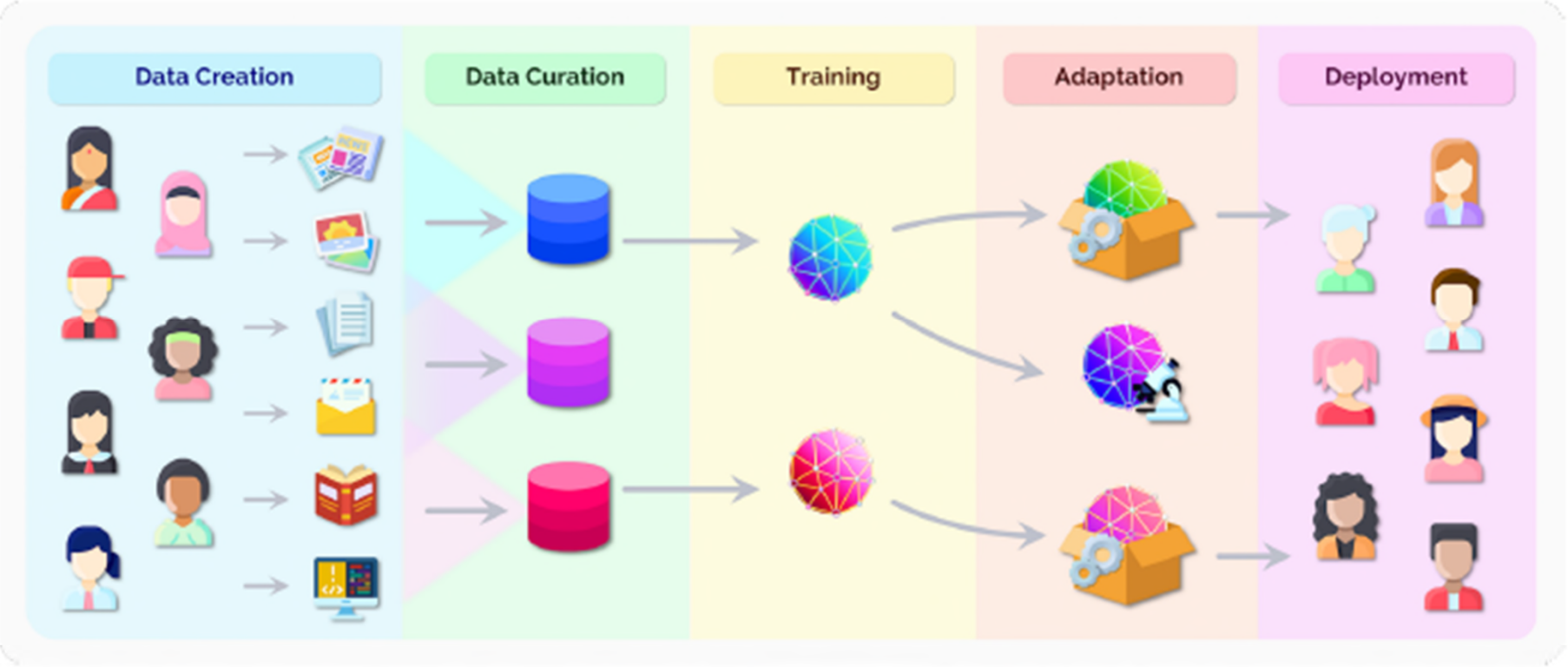


Fig. 3. Before reasoning about the social impact of foundation models, it is important to understand that they are part of a broader ecosystem that stretches from data creation to deployment. At both ends, we highlight the role of people as the ultimate source of data into training of a foundation model, but also as the downstream recipients of any benefits and harms. Thoughtful data curation and adaptation should be part of the responsible development of any AI system. Finally, note that the deployment of adapted foundation models is a decision separate from their construction, which could be for research.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Applications of FMs

- Language
- Vision
- Robotics
- Reasoning
- Coding
- Etc.

Language

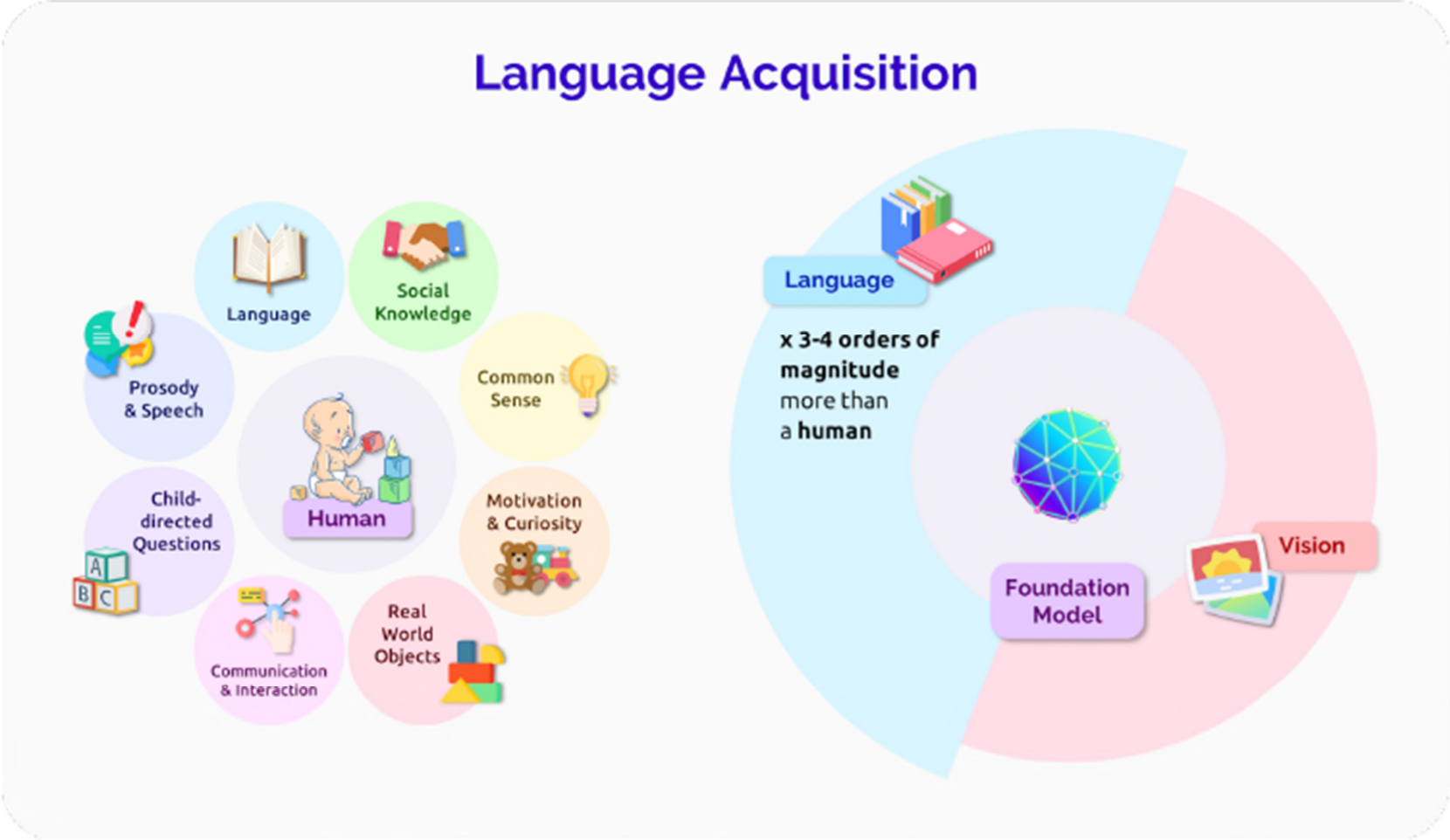


Fig. 6. Language Acquisition for humans and foundation models. While there are certainly different inductive biases between the human brain and foundation models, the ways that they learn language are also very different. Most saliently, humans interact with a physical and social world in which they have varied needs and desires, while foundation models mostly observe and model data produced by others.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Vision

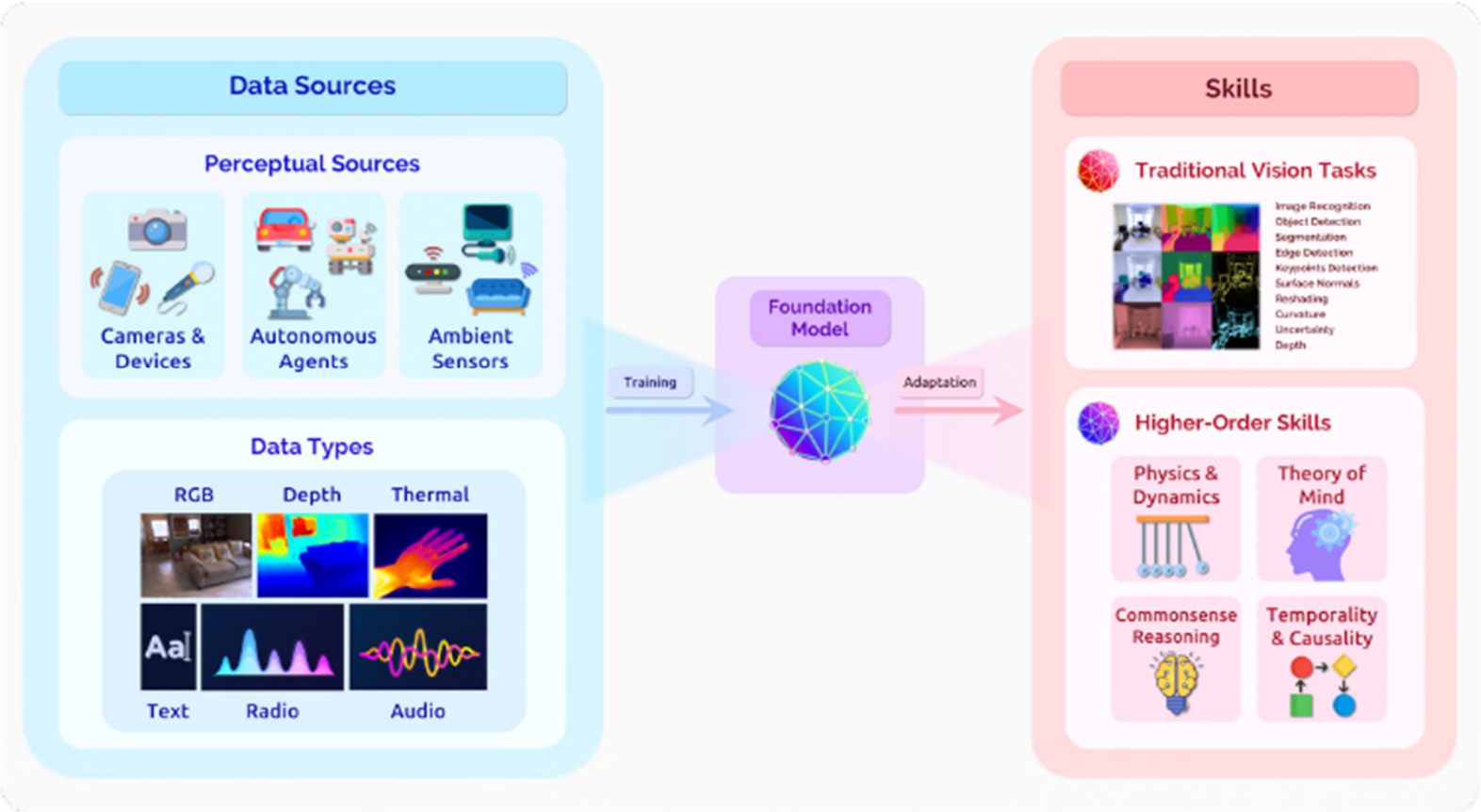


Fig. 7. By harnessing self-supervision at scale, foundation models for vision have the potential to distill raw, multimodal sensory information into visual knowledge, which may effectively support traditional perception tasks and possibly enable new progress on challenging higher-order skills like temporal and commonsense reasoning (§2.2.1: VISION-CAPABILITIES). These inputs can come from a diverse range of data sources and application domains, suggesting promise for applications in healthcare and embodied, interactive perception settings (§2.2.2: VISION-CHALLENGES). Image credits [Zamir et al. 2018; Haque et al. 2020].

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Robotics

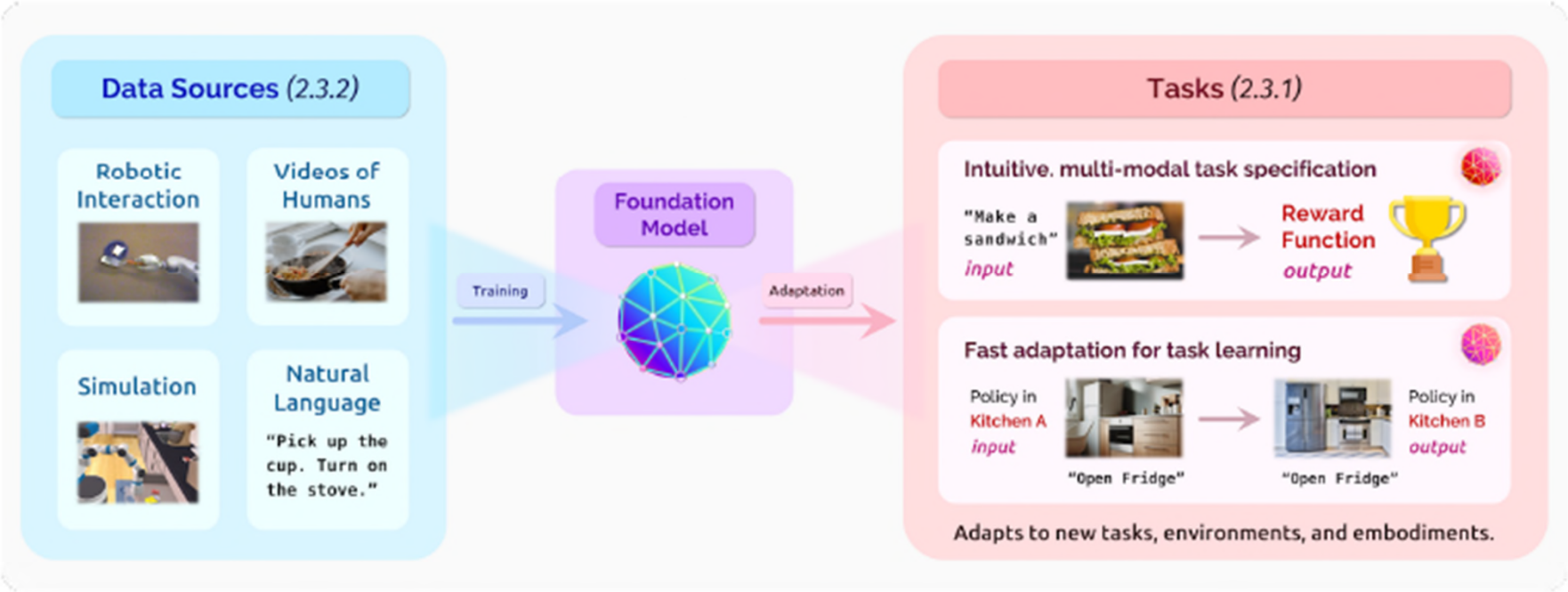


Fig. 8. Building new types of foundation models for robotics will require massive datasets spanning diverse environments and behaviors. Simulation, robotic interaction, videos of humans, and natural language descriptions could all be useful data sources for these models. Despite the challenges of acquiring data, developing new foundation models for robotics has tremendous potential for a variety of problem formulations in task specification and robot learning. Image credits: [Finn et al. 2016b; Szot et al. 2021].

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Issues & Challenges of FMs

- Modelling
- Adaptation
- System
- Security and privacy

Modeling

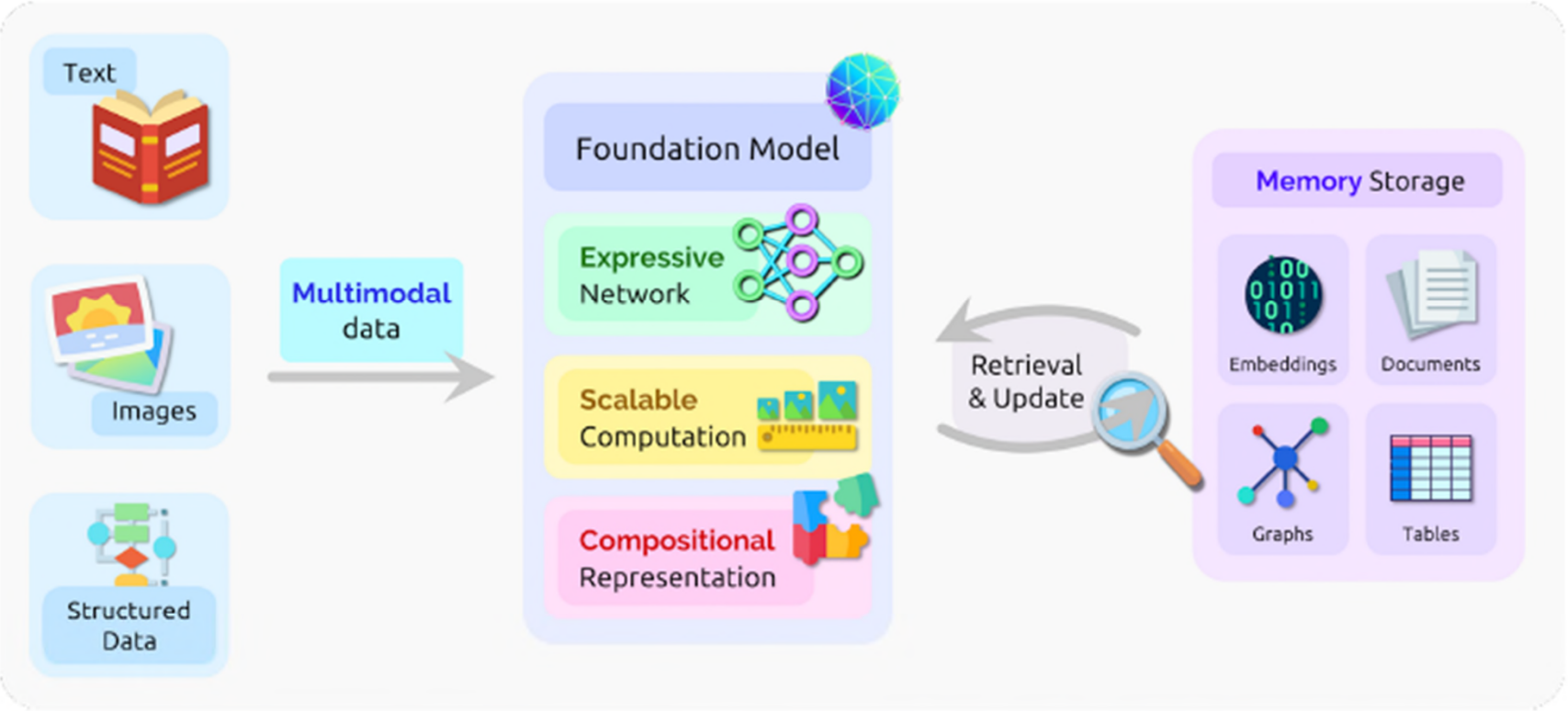


Fig. 17. The five key properties of a foundation model: *expressivity* — to flexibly capture and represent rich information; *scalability* — to efficiently consume large quantities of data; *multimodality* — to connect together various modalities and domains; *memory capacity* — to store the vast amount of accumulated knowledge; and *compositionality* — to generalize to new contexts, tasks and environments.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Adaptation

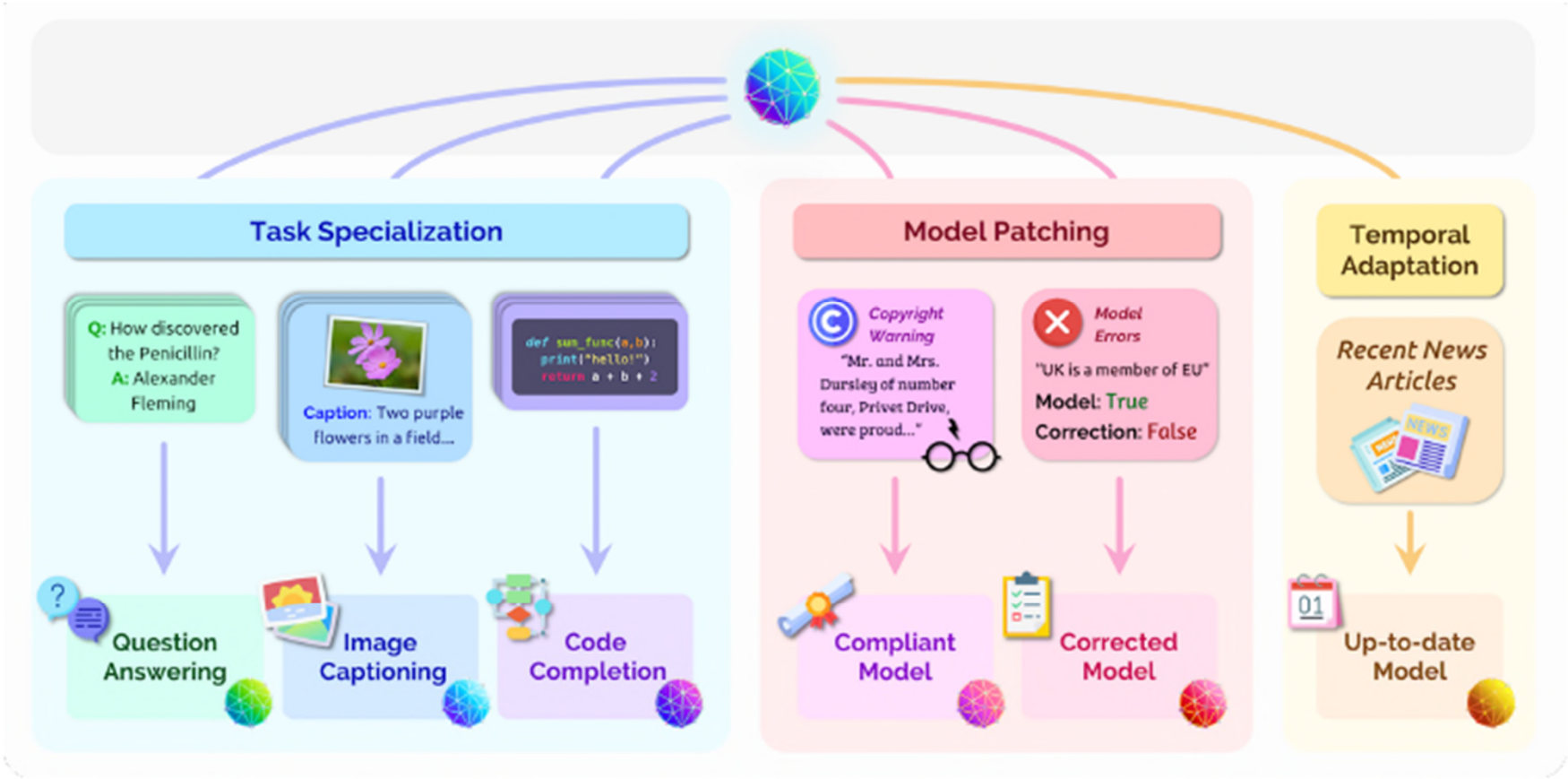


Fig. 18. During adaptation, a foundation model is converted into an *adapted model* (bottom row) in order to reflect updated information, desired behaviors, or deployment constraints.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

System

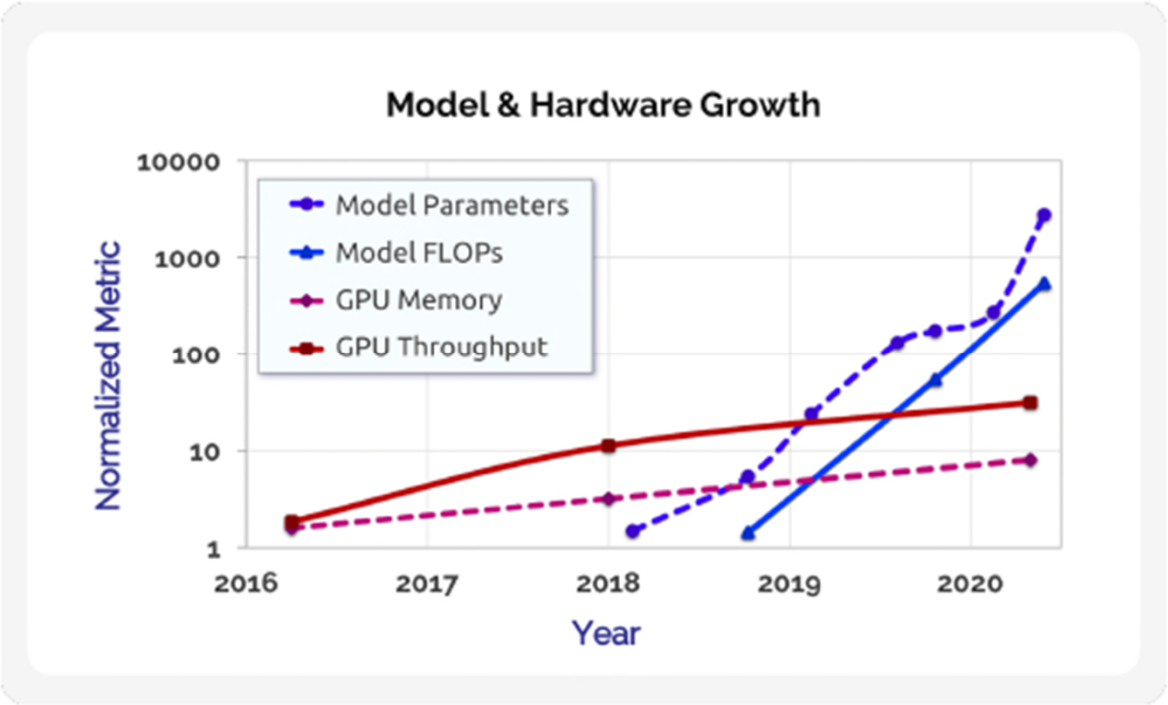


Fig. 19. Plot showing the growth of number of parameters and number of training operations (FLOPs) of transformer-based language models (shown in blue), and memory capacity and peak device throughput of NVIDIA P100, V100, and A100 GPUs (shown in red) with time. The rate of growth (slope of each line) of state-of-the-art language models (roughly 10× a year) far exceeds the rate of increase in computational capacity of hardware (roughly 10× in four years), motivating the need for parallelism across a large number of accelerators and co-design of algorithms, models, software, and hardware to drive further progress. Number of parameters and number of training operations are obtained from relevant papers [Brown et al. 2020], and memory capacities and peak throughputs are obtained from GPU specification sheets.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Security & Privacy

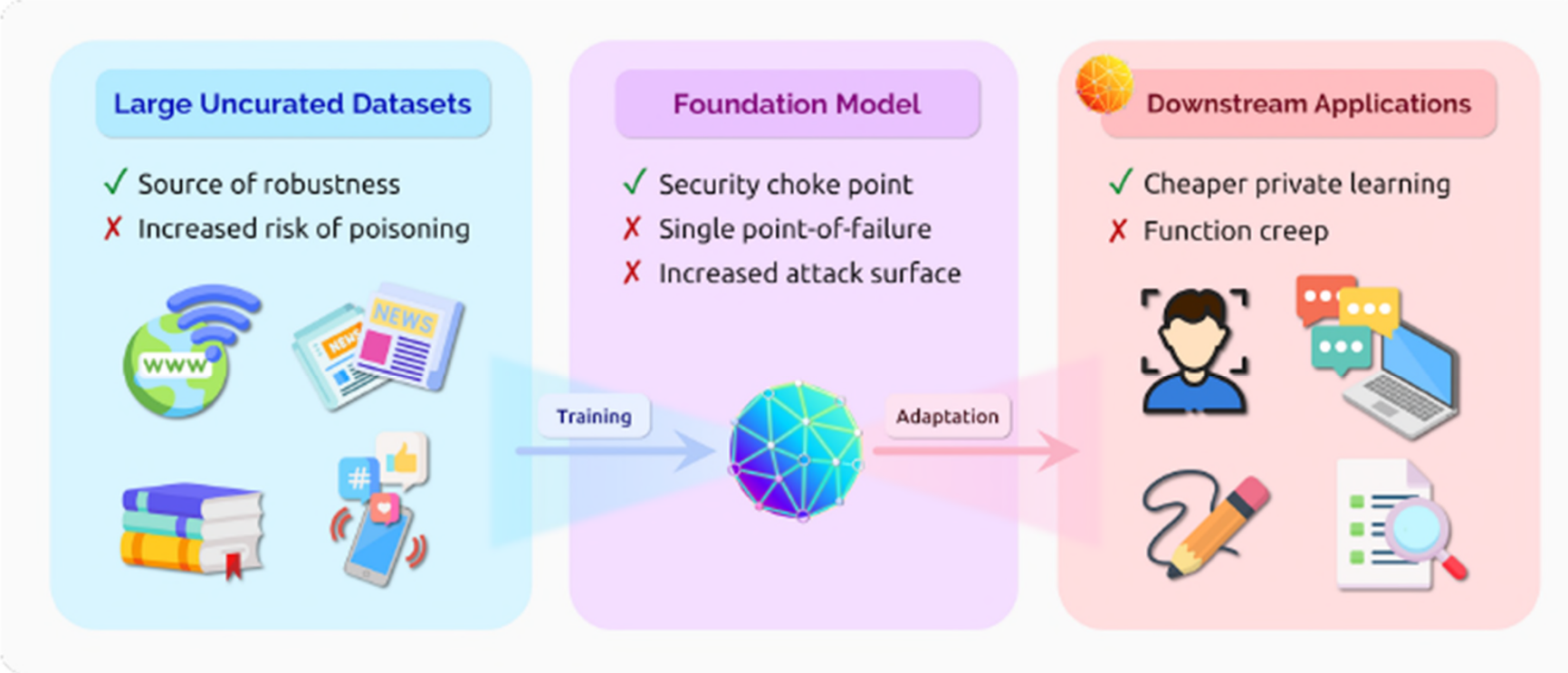


Fig. 20. Risks and opportunities raised by foundation models for security and privacy of ML systems.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Risks of FMs

- Inequity and fairness
- Misuse
- Etc.

Inequity and Fairness

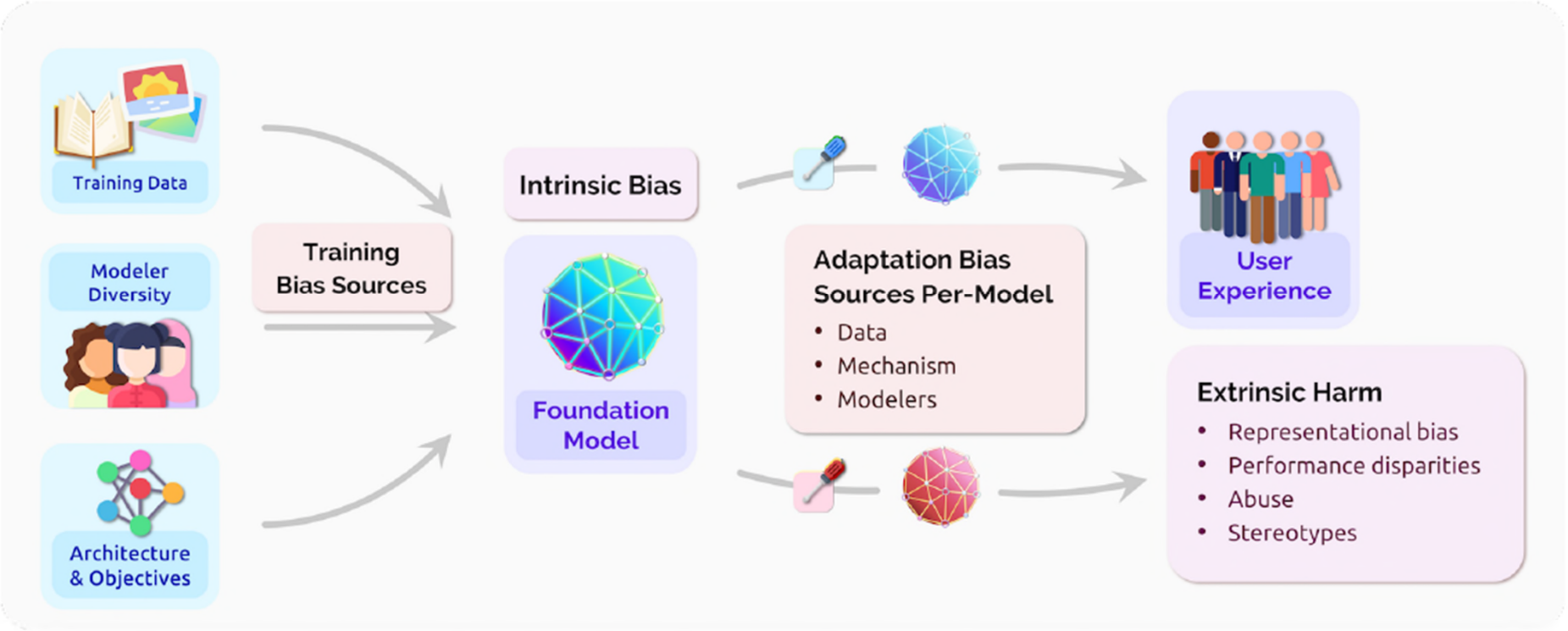


Fig. 24. The *intrinsic bias* present within foundation models is the byproduct of various training bias sources (**left**) which, alongside biases introduced during adaptation, determines the *extrinsic harms* (**right**) experienced by users in the context of specific downstream applications. We emphasize that the same foundation model is the shared foundation for many different applications; its biases propagate to these many applications as a result. Further, since the harms experienced by users are the result of specific adapted models, attributing these harms to the various processes and sources depicted in this diagram is both crucial and challenging.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Misuse

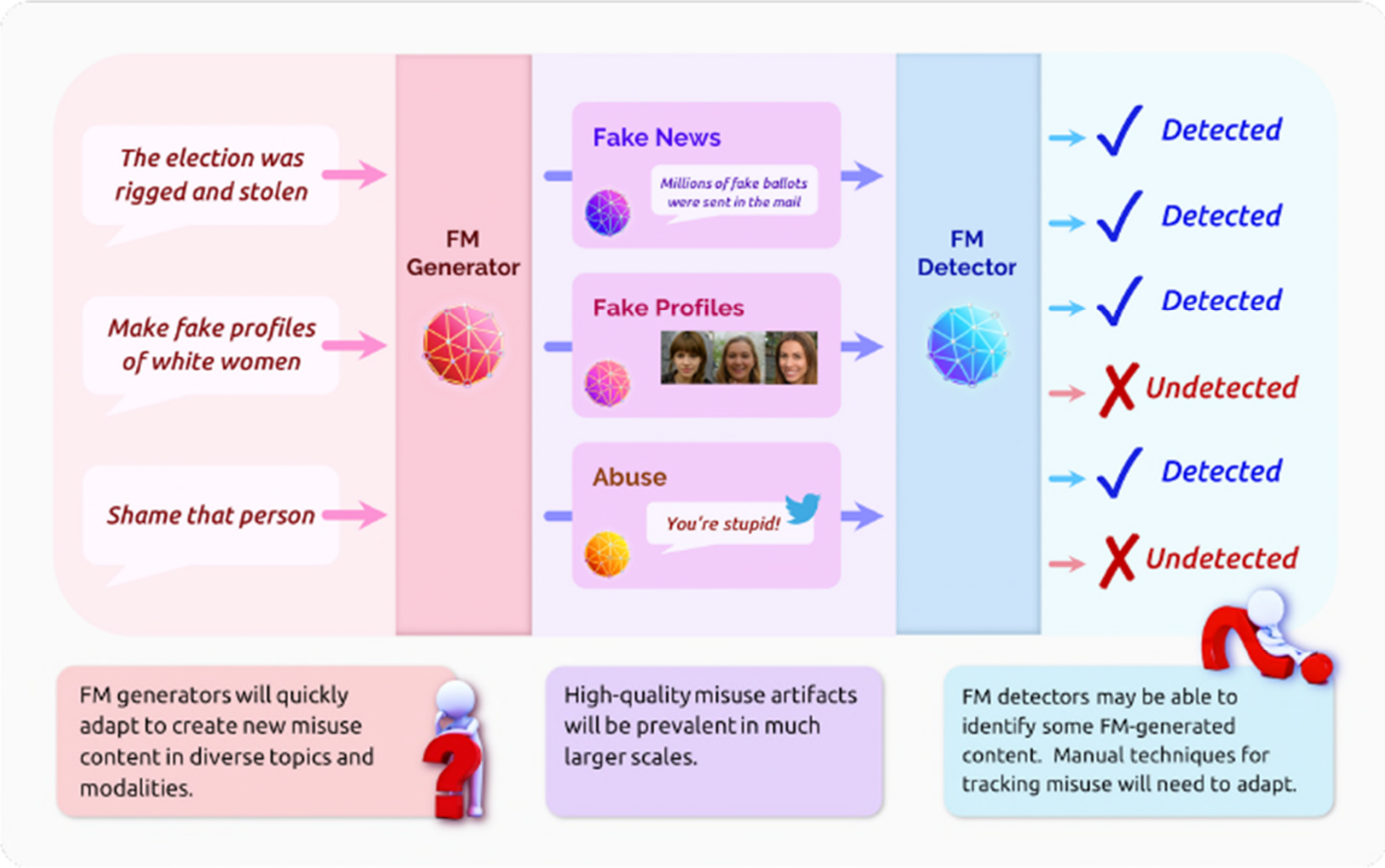


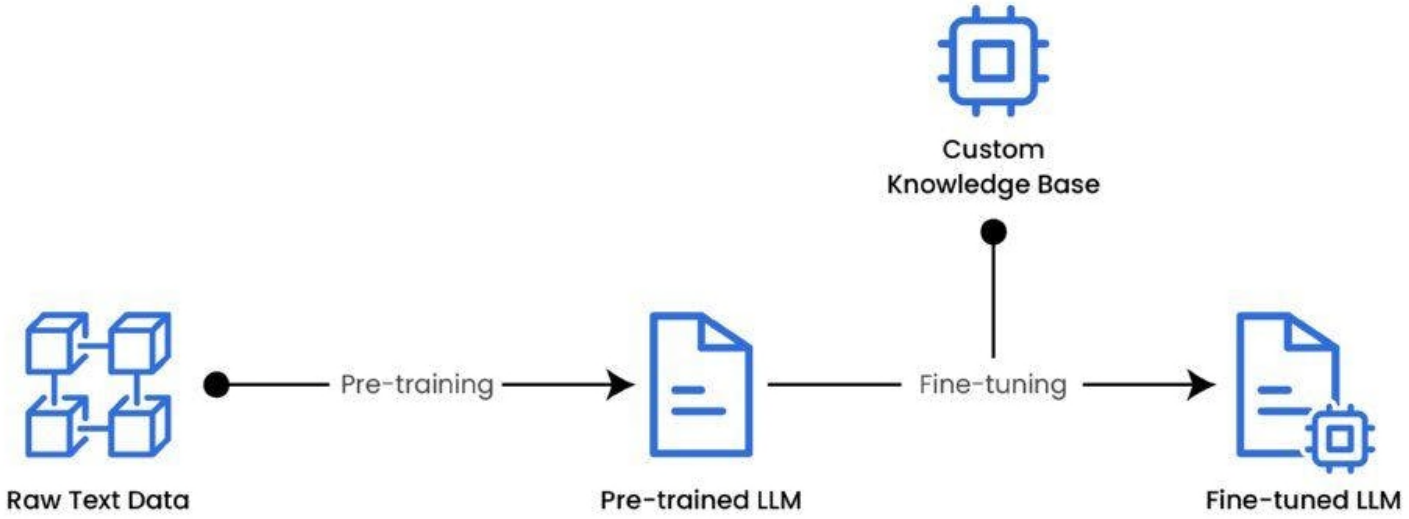
Fig. 25. This figure shows the effect foundation models will have on manipulative and harmful content generation, and the implications for detection.

Source: Rishi Bommasani, On the Opportunities and Risks of Foundation Models.

Adapting / Finetuning FMs

What is Fine-tuning?

- A process of adjusting parameters of a pre-trained model for a specific downstream task / use case.
- Pre-trained models possess vast & diverse knowledge but lack specialization in specific areas.
- Fine-tuning addresses this gap by adapting the model to learn from domain-specific data to make it more accurate and effective for targeted applications.



Source: https://www.turing.com/resources/finetuning-large-language-models

Fine-tuning Process

- Data preparation
  - Data cleaning, handling missing values, formatting text to match model's input requirements, data augmentation, etc.
- Pre-trained model selection
  - Understand architecture, input/output specifications of models.
  - Consider factors such as model size, training data, performance, etc.
- Method selection for fine-tuning
  - Choose suitable fine-tuning or adaptation method for specific application.
- Validation
  - Evaluate a fine-tuned model’s performance on target validation set.
- Model Iteration
  - Repeat the process above, if necessary.
- Deployment
  - Position the fine-tuned model in real-world target application.

https://www.turing.com/resources/finetuning-large-language-models

Adaptation Techniques

- Adapter Tuning
- Low Rank Adaptation (LoRA)
- Quantized Low Rank Adaptation (QLoRA)
- Prefix Tuning
- Prompt Tuning

Adaptor Tuning

- A technique to fine-tune large pre-trained models.
- Introduce adaptors which are small network layers.
- Train only the adaptors.
- Useful for limited computational resources or labelled data.

Key Steps in Adaptor Tuning

- Introduce small lightweight adaptors (networks) between layers of pre-trained model.
- During fine-tuning, only the adaptors are trained while the original weights of the foundation model are frozen.
- Significantly reduce the number of parameters that need to be updated, making fine-tuning more efficient in terms of computation and memory usage.

Adaptor Tuning

- A sample adapter consisting of 2 FC layers & nonlinear activation added to Transformer.

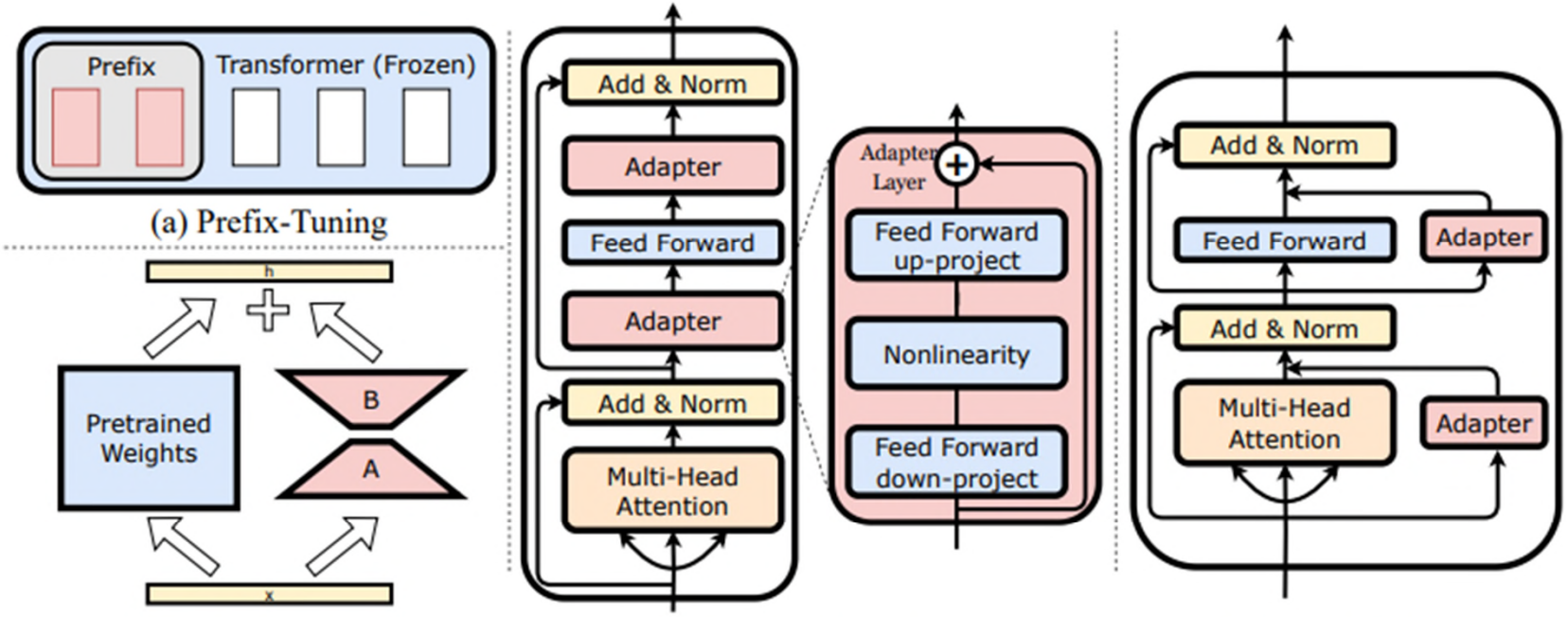


Figure 1: A detailed illustration of the model architectures of three different adapters: (a) Prefix-Tuning, (b) LoRA, (c) Series Adapter, and (d) Parallel Adapter.

Source: Zhiqiang Hu et. al. LLM-Adapters: An Adapter Family for Parameter-Efficient Fine-Tuning of Large Language Models

What is Low-Rank Adaptation (LoRA)?

- An efficient fine-tuning method for foundation models such as large language models.
- Low computational overhead and parameter update.
- E.g., GPT-3: 175B fine-tuned with Adam vs reduction of trainable parameters by 10,000X and GPU memory requirement by 3X with LoRA.

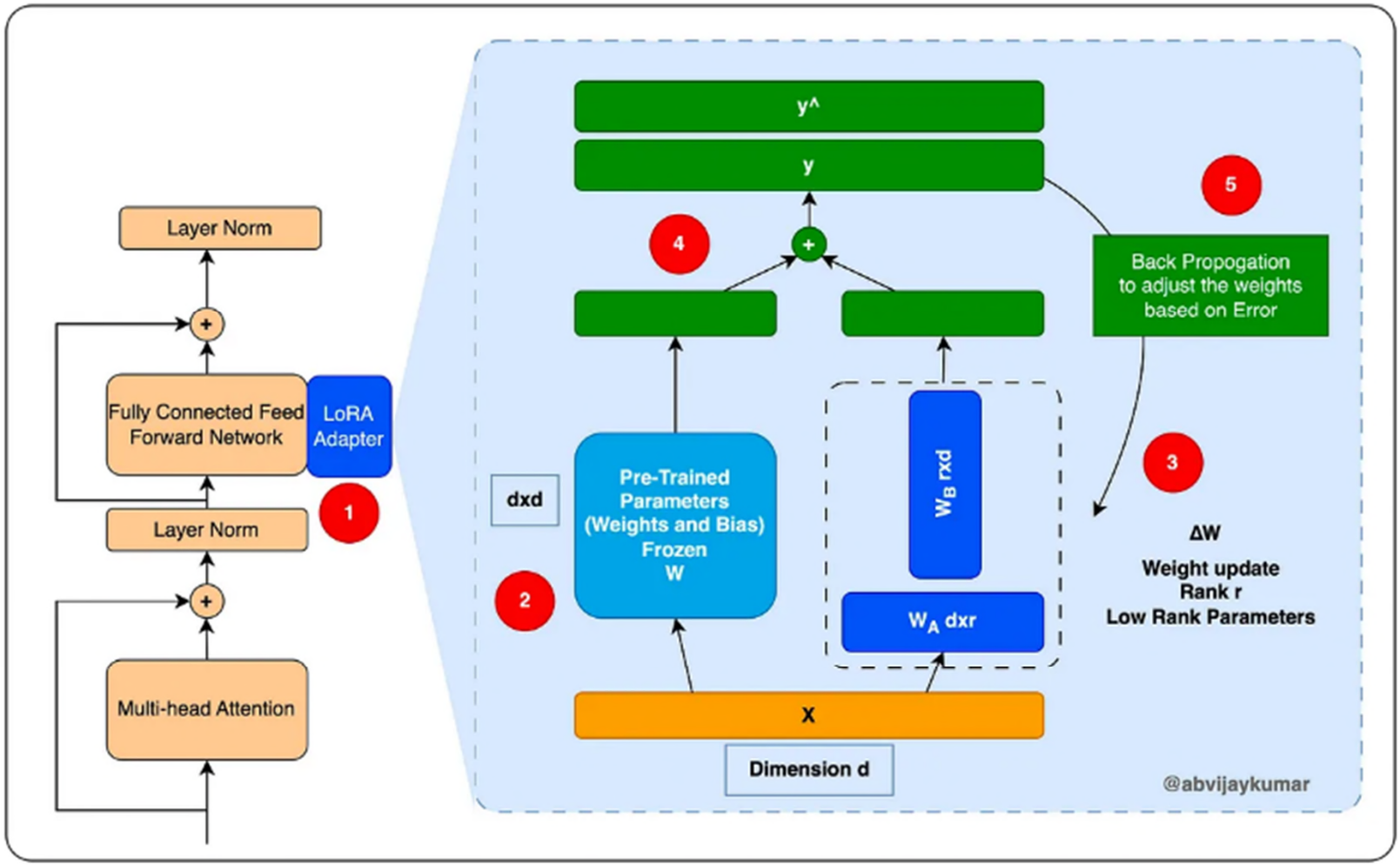
Steps of LoRA

- Select a pre-trained FM.
- Identify target layers for adaptation.
  - Common selected layer: attention layer in a transformer.
- Introduce low-rank matrices for low-rank decomposition.
  - For each selected layer, introduce two low-rank matrices A and B that approximate changes to the original weight matrix W.

$$W' = W + A \times B$$

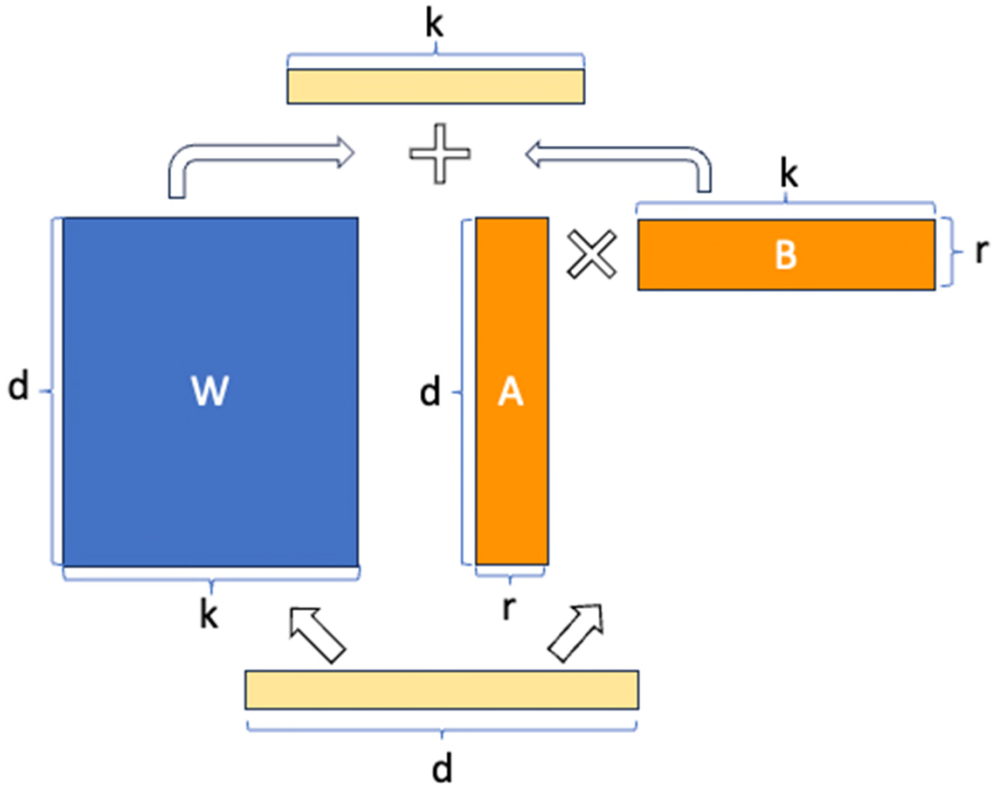
- Fine-tune the model.
  - Freeze the original weight matrix W. Update only the low-rank matrices A and B using target use case data.
- Evaluate the adapted model performance.
- Optimization
  - Hyperparameter tuning, model evaluation.
- Deploy and Monitor

Steps of LoRA



A neural network utilizing a LoRA Adapter to update pre-trained network parameters efficiently [Source: @abvijaykumar]

Visualization of LoRA



Source: https://dataman-ai.medium.com/fine-tune-a-gpt-lora-e9b72ad4ad3

What is Quantized Low-Rank Adaptation (QLoRA)?

- A more efficient way to fine-tune a FM than LoRA.
- Combine LoRA with quantization to optimize memory usage and computational efficiency.
- QLoRA
  - Use LoRA as the Parameter Efficient Fine-tuning (PEFT) method.
  - Add a small number of trainable parameters while keeping the original parameters frozen.

QLoRA: Quantized Low-Rank Adaptation

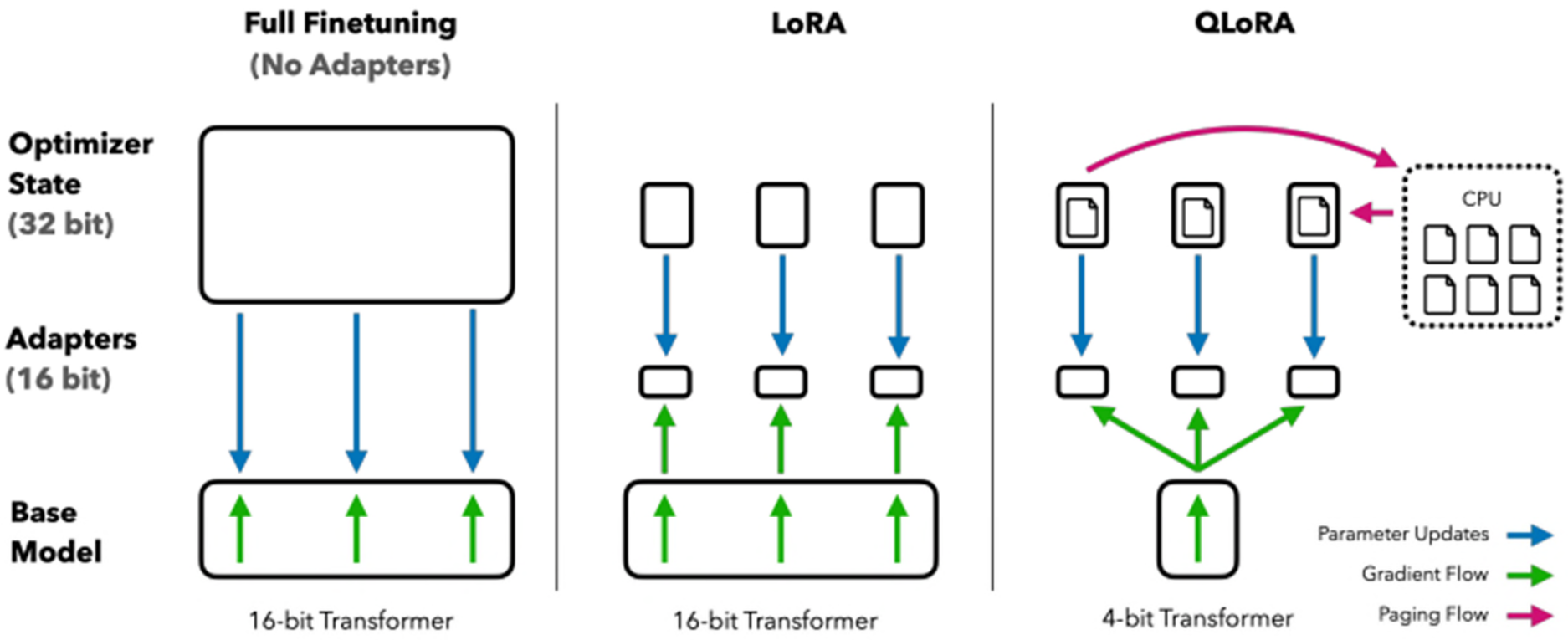


Figure 1: Different finetuning methods and their memory requirements. QLoRA improves over LoRA by quantizing the transformer model to 4-bit precision and using paged optimizers to handle memory spikes.

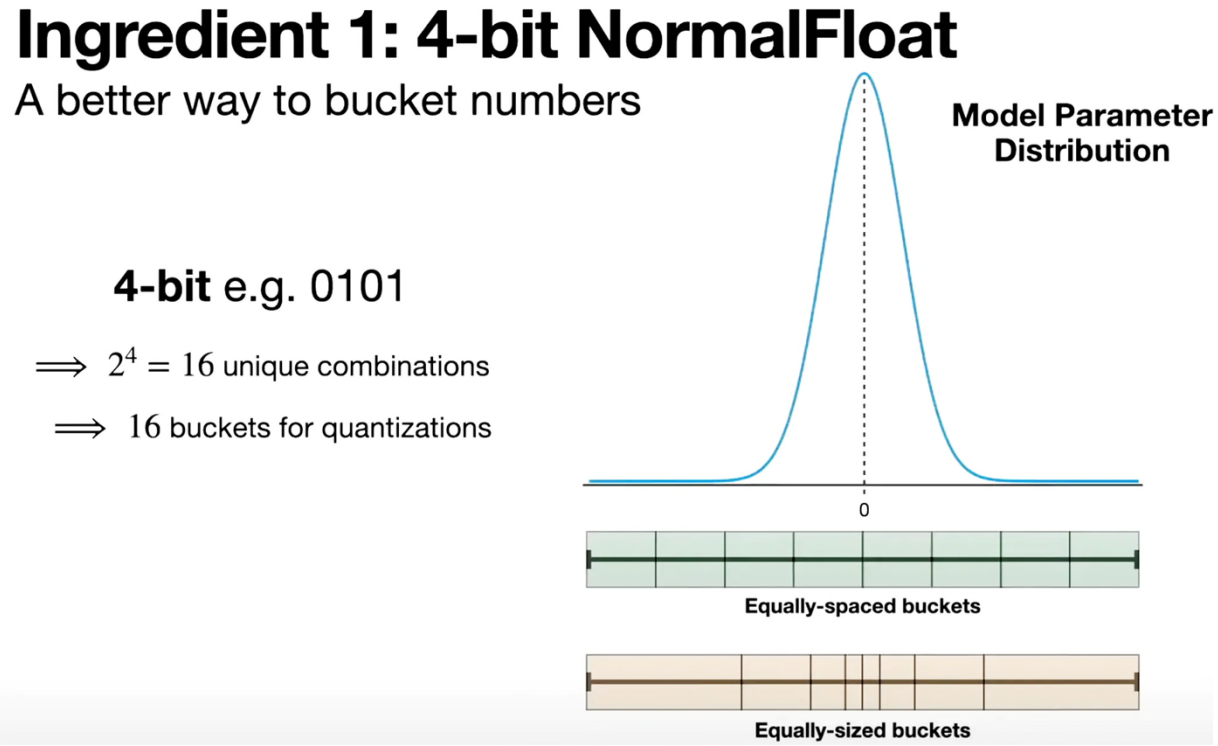
Source: Tim Dettmers, et. al. QLORA: Efficient Finetuning of Quantized LLMs

Key Ideas of QLoRA

- 4-bit NormalFloat (NF4)
  - A new data type optimal for normally distributed weights.
- Double Quantization
  - Reduce average memory footprint by quantizing the quantization constants.
- Paged Optimizers
  - Manage memory spikes.

4-bit NormalFloat (NF4)

- Encode numbers with just 4 bits.
- An optimal quantization data type for normally distributed data that yields better empirical results than 4-bit Integers and 4-bit Floats.



Source: https://nagi.fun/fine-tune-llm?locale=en

Double Quantization

- Double Quantization treats quantization constants of the first quantization as inputs to a second quantization.
- Achieve memory saving by quantizing the quantization constants.
- Partition model parameters into smaller blocks for quantization.

Paged Optimizers

- Perform automatic page-to-page transfers between the CPU and GPU in scenario where the GPU occasionally runs out-of-memory.

Prefix Tuning

- A technique to fine-tune large pre-trained models by prepending a sequence of learnable vectors (prefixes) to the model's input.
- Modify input representation rather than the model's internal parameters, making it a lightweight and efficient fine-tuning technique.

Key Steps in Prefix Tuning

- Introduce a sequence of learnable vectors (prefixes) to the model's input.
- Prefixes are additional tokens that are prepended to the original input tokens.
- Extended input sequence = prefix embeddings + original input embeddings is fed into pre-trained model.
- This allows prefixes to influence the model's internal representations and outputs.
- During fine-tuning, only the prefix embeddings are trained. The rest of the model's parameters are frozen.
- Prefixes learn task-specific features that help model adapt to new task.

Prefix Tuning

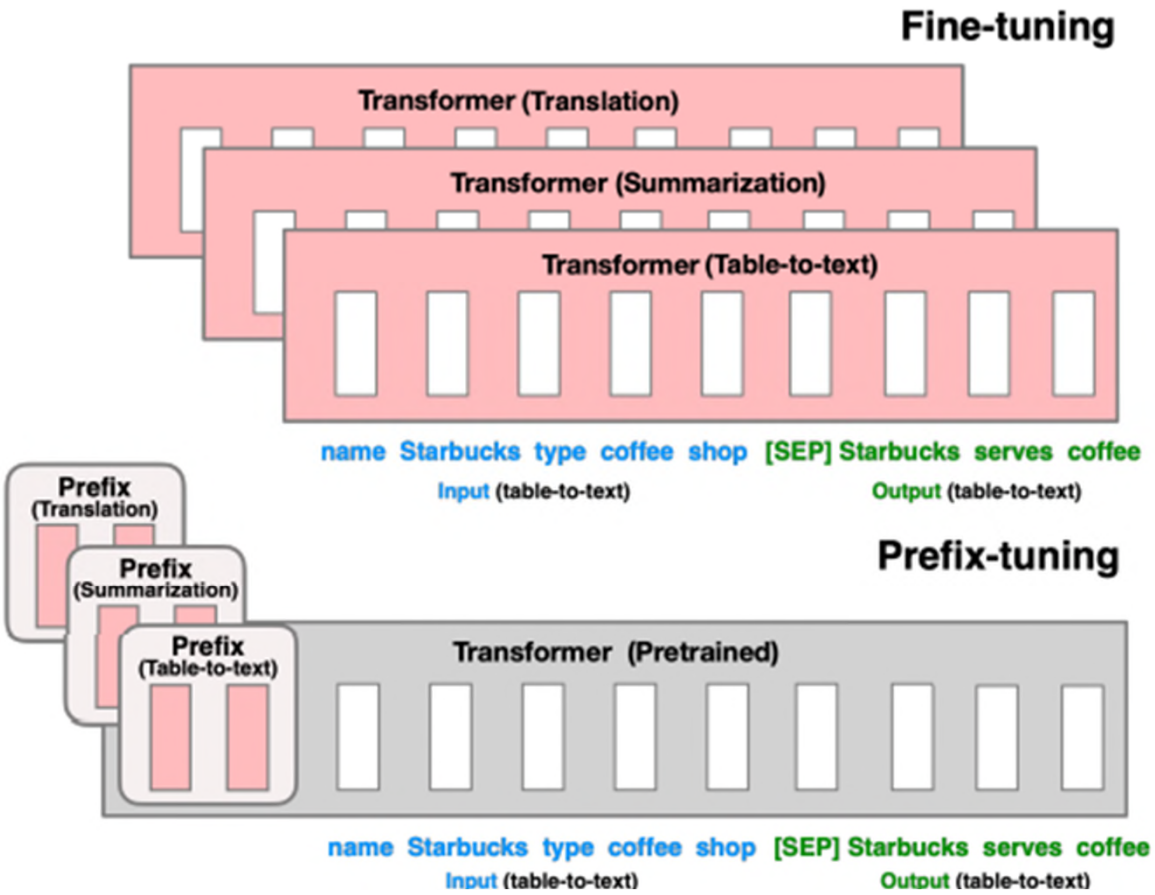


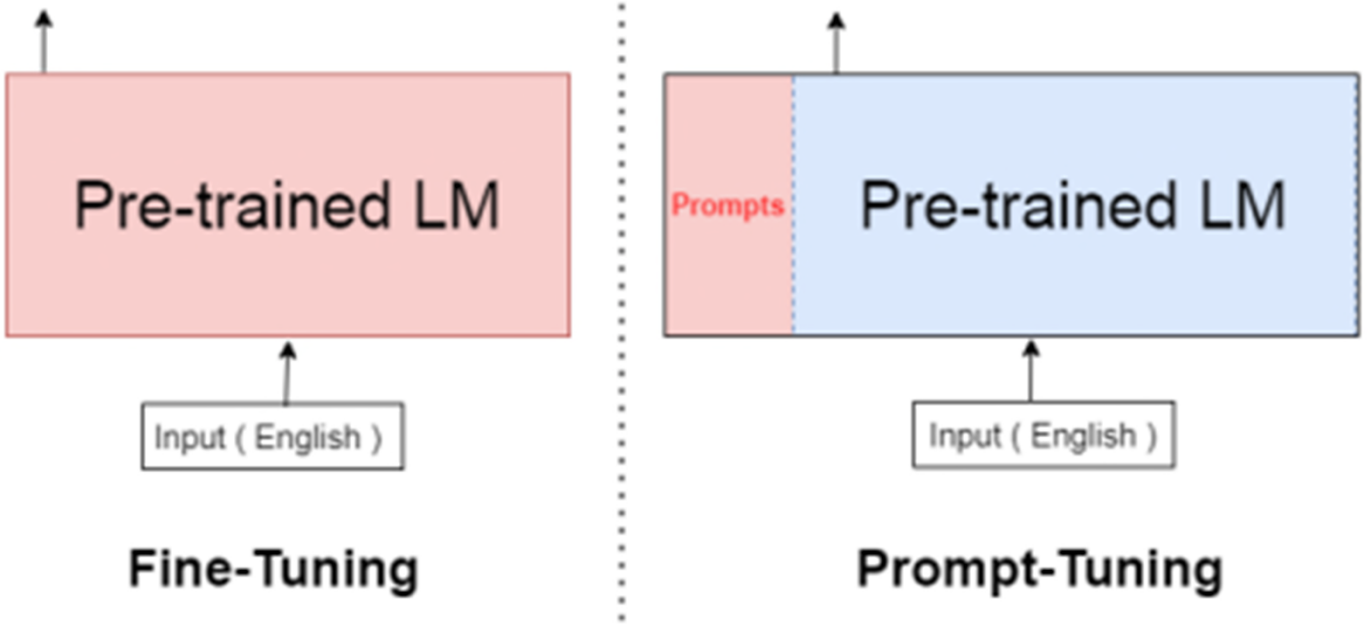
Figure 1: Fine-tuning (top) updates all Transformer parameters (the red Transformer box) and requires storing a full model copy for each task. We propose prefix-tuning (bottom), which freezes the Transformer parameters and only optimizes the prefix (the red prefix blocks). Consequently, we only need to store the prefix for each task, making prefix-tuning modular and space-efficient. Note that each vertical block denote transformer activations at one time step.

Source: Xiang Lisa Li and Percy Liang. Prefix-Tuning: Optimizing Continuous Prompts for Generation

Prompt Tuning

- A technique that uses prompts to guide model's behaviour.
- Prompts can be static (fixed text) or dynamic (learnable embeddings) that are introduced to the input sequence.
- During fine-tuning, only the prompts embeddings are trained. The rest of the model parameters are frozen.
- Modify the model's input contextually to guide its responses without changing the model's internal parameters.
- Difference with prefix tuning:
  - Prefix tuning uses learnable vectors (not actual text), while prompt tuning can use both fixed text prompts or learnable embeddings.

Prompt Tuning



Source: Lifu Tu et. al. Prompt-Tuning Can Be Much Better Than Fine-Tuning on Cross-lingual Understanding With Multilingual Language Models

Section I Overview

- The section covers the following topics:
  - Current State of AI
  - Basics of Foundation Models (FMs)
  - Adapting / Finetuning FMs

Section II
Large Language Models (LLMs)

Section II Overview

- The section covers the following topics:
  - Introduction
  - BERT
  - GPT
  - Well-Known LLMs
  - Emerging Trends

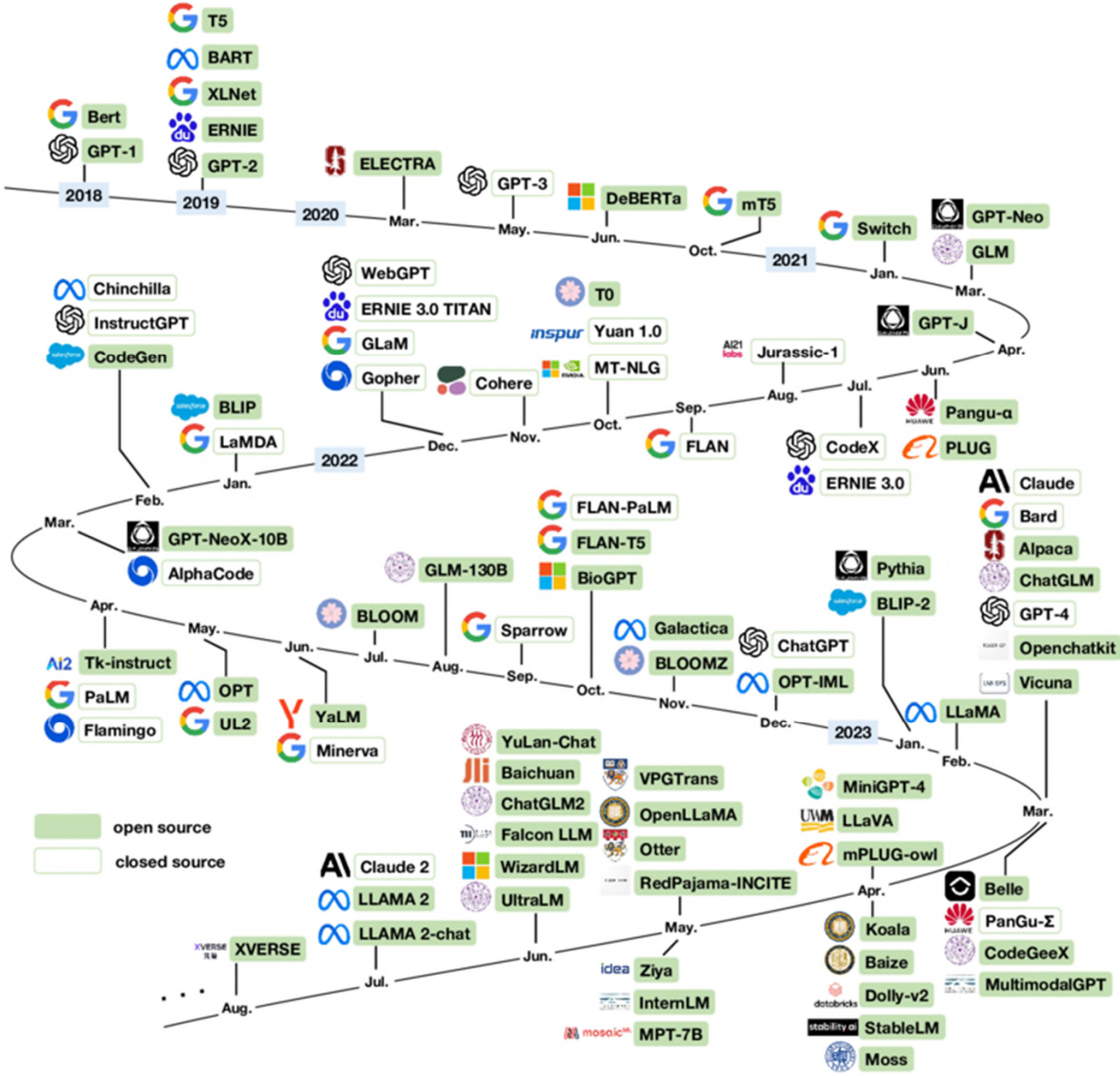
What are LLMs?

- LLMs are advanced artificial intelligence models designed to understand, generate, and manipulate human language.

Key Features of LLMs

- Massive Scale: neural networks with billions of parameters, enabling them to handle a wide range of linguistic tasks.
- Training Data: trained on massive amounts of text data from various sources, allowing them to learn different aspects of human language.
- Capabilities: can handle different tasks such as language translation, summarization, question answering, text generation, sentiment analysis, etc.
- Applications: diverse applications including natural language processing, chatbots, content creation, language understanding, etc.
- Challenges: bias, misuse, societal impacts, etc.

Timelines of Popular LLMs



Source: https://www.researchgate.net/figure/A-chronological-overview-of-large-language-models-LLMs-multimodal-and-scientific_fig2_373451304

Bidirectional Encoder Representations from Transformers (BERT)

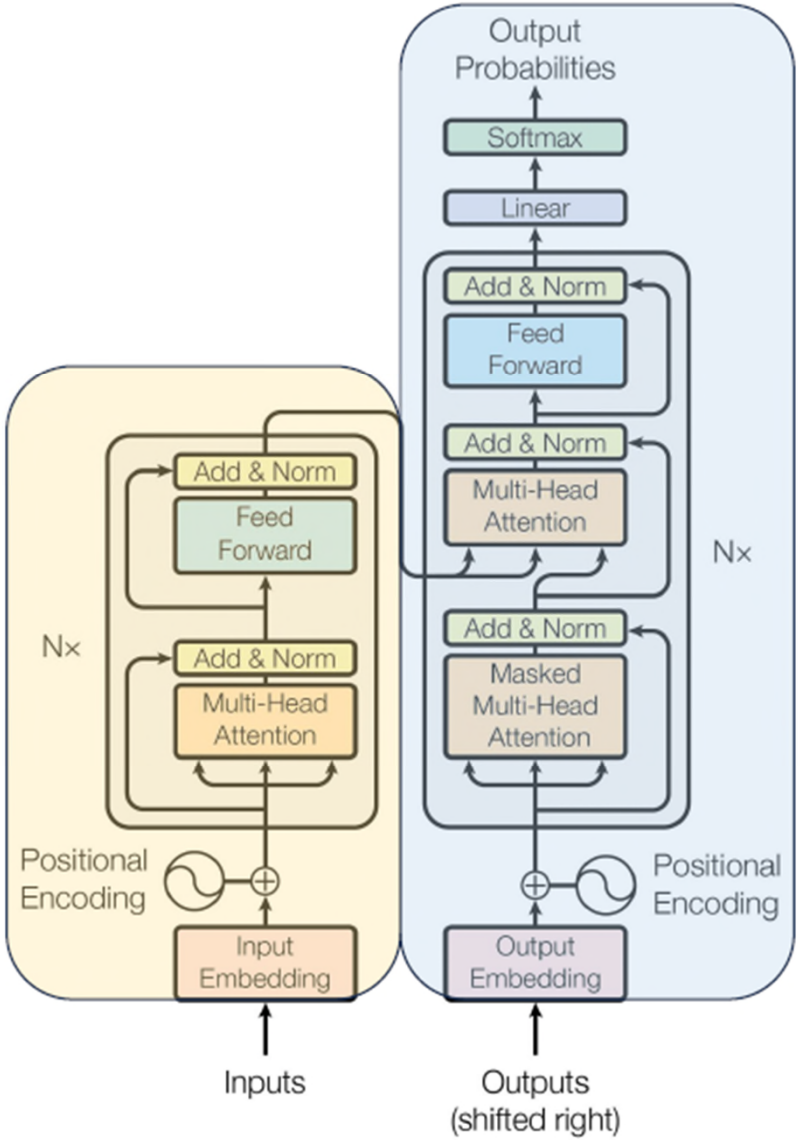
Introduction

- An NLP model developed by Google around 2018.
- A transformer encoder with bidirectional self-attention.
- Understand bidirectional context relationship between words.
- Can generate contextualized word embeddings.
- Two versions with varying model sizes and complexities:
  - BERT-base (110M parameters)
  - BERT-large (340M parameters)
- Variants: RoBERTa, ALBERT, DistilBERT, and ELECTRA.

Transformer: BERT + GPT

BERT
Oct 2018

Representation



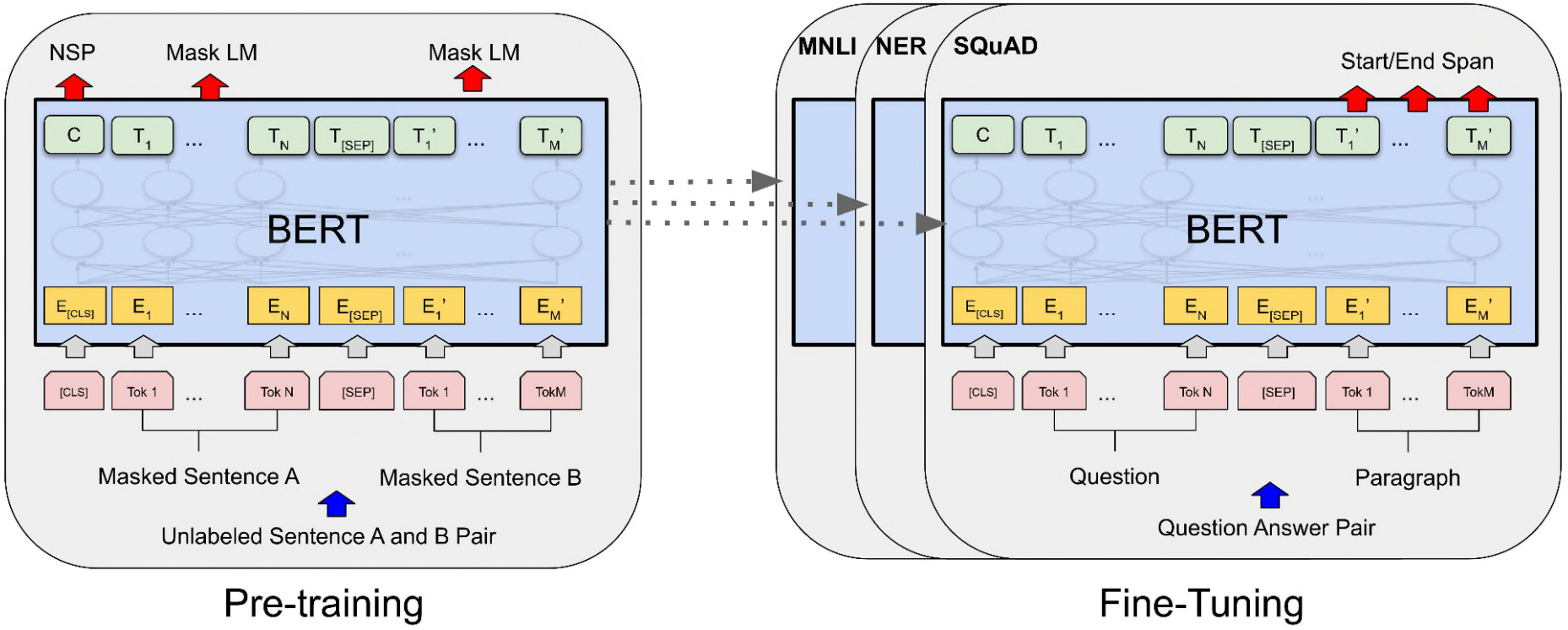
GPT
June 2018

Generation

Source: CMU Course11-785 Introduction to Deep Learning

BERT Training

- Two stages:
  - Stage 1: pretrain BERT to understand language.
  - Stage 2: fine-tune BERT to learn specific downstream task.



Source: Jacob Devlin et. al. “BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding”

Stage 1: Pre-Training

- Use unsupervised / self-supervised learning.
- Goal: to learn language structure & context.
  - Masked Language Model (MLM)
    - Predict masked words from surrounding words.
    - Aim to achieve context learning.
  - Next Sentence Prediction (NSP)
    - Predict if a sentence follows another in a text pair.
    - Learn logical context between sentences.
- BERT pretraining corpus
  - English Wikipedia - 2,500 million words.
  - Book Corpus - 800 million words.

BERT Input Representation

- Token Embedding: WordPieces, 30K vocabulary.
- Segment embedding: sentence numbering.
- Position embedding: capture position of words in a sentence.

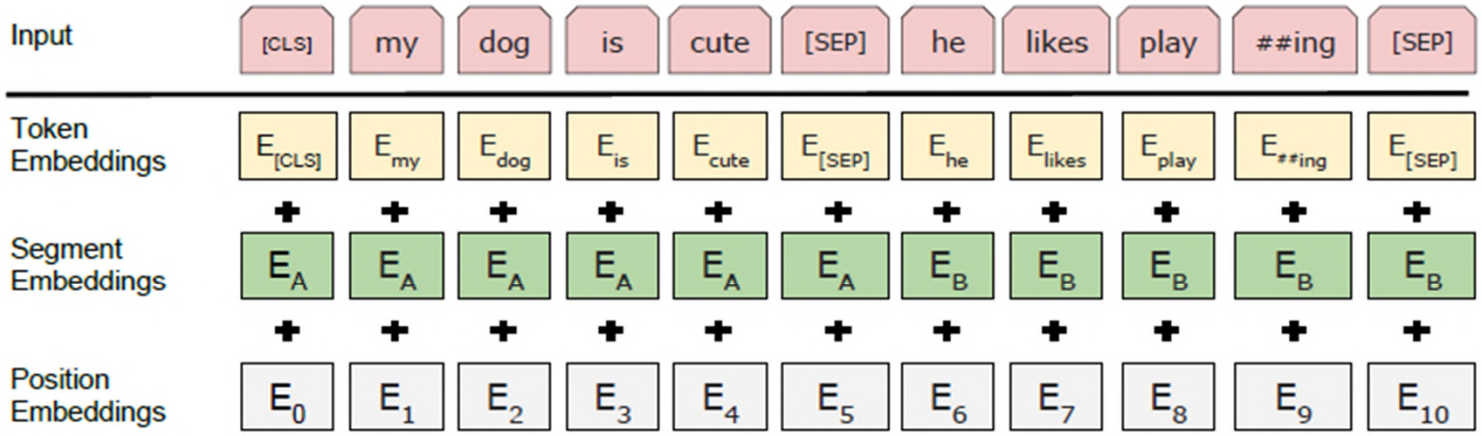


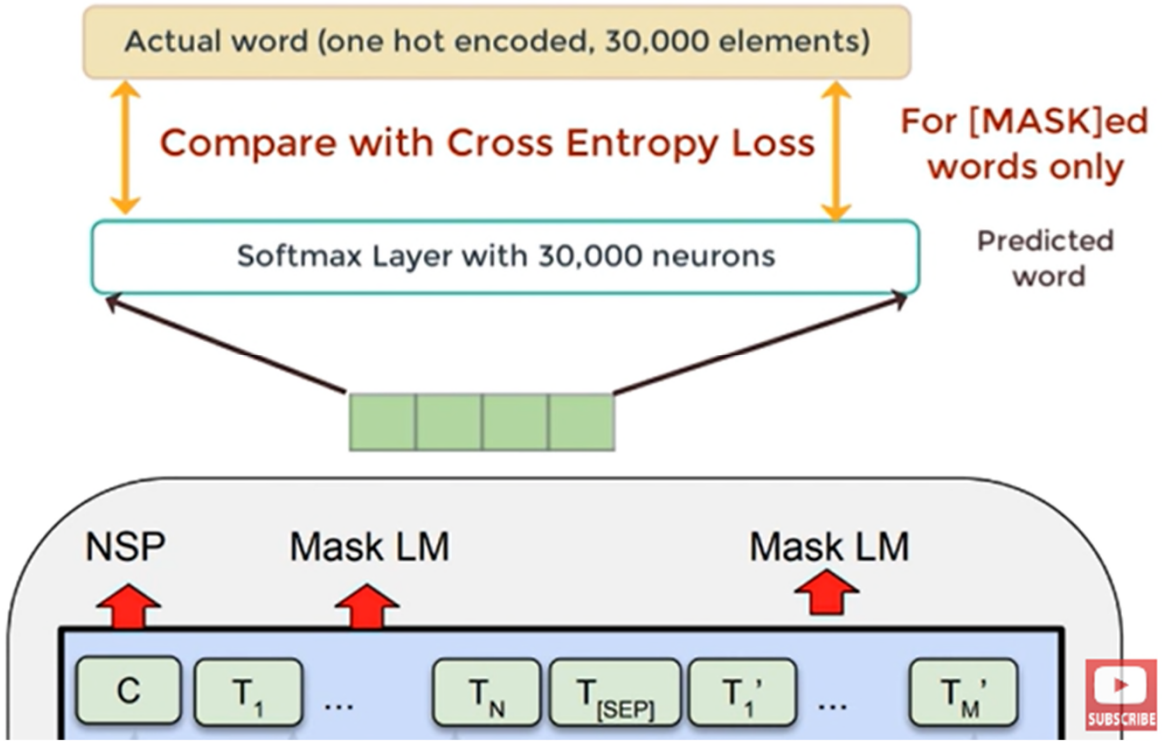
Figure 2: BERT input representation. The input embeddings are the sum of the token embeddings, the segmentation embeddings and the position embeddings.

Source: Jacob Devlin et. al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"

Stage 1: Pre-Training

Word vectors T_i have the same size.

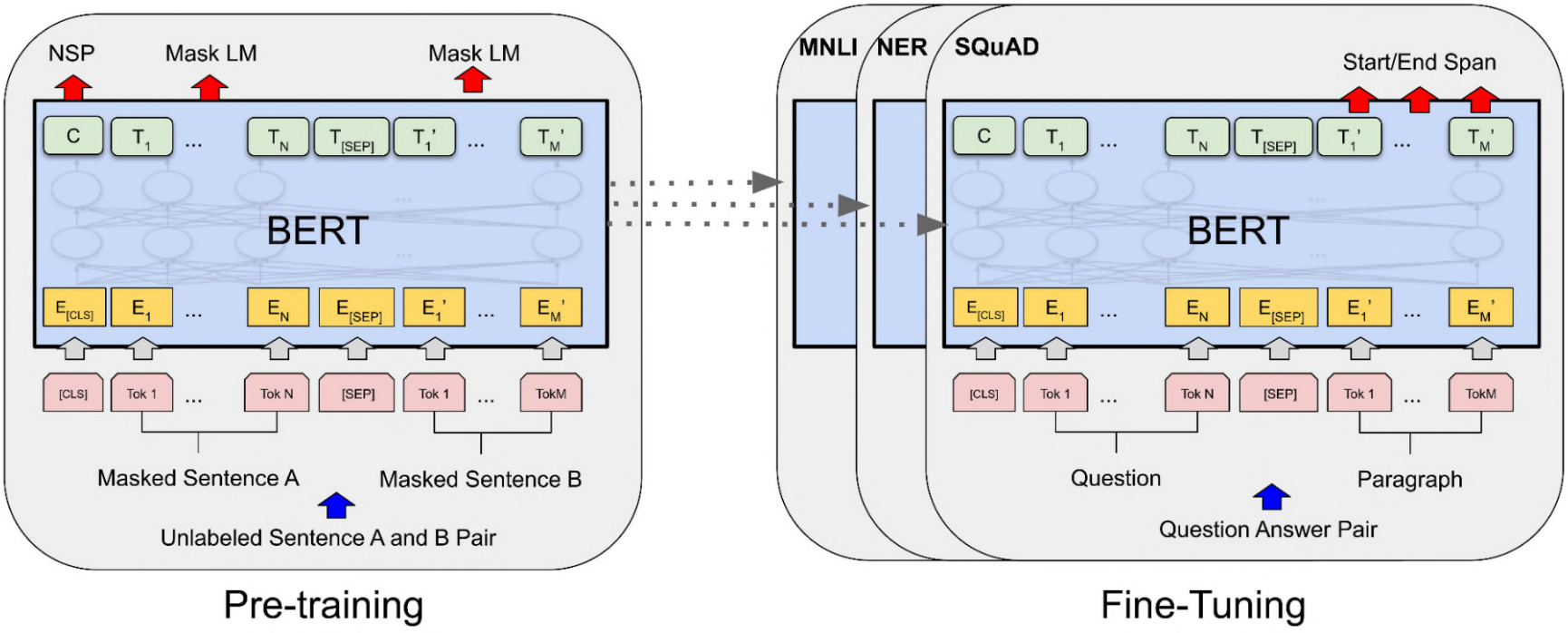
Word vectors T_i are generated simultaneously



Source: Code Emporium, BERT Neural Network - EXPLAINED! Youtube.

Stage 2: Finetuning for Specific Tasks

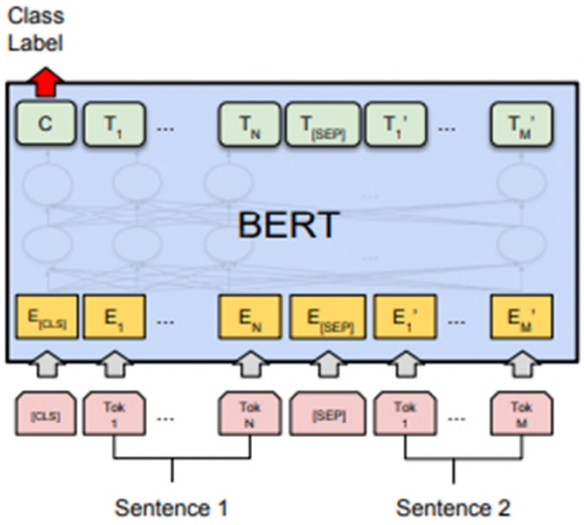
- Use supervised learning: add a task-specific module after the last encoder layer so that it can map to the target task.
- Example: for Question Answering Task, train two extra vectors to mark the beginning and end of answer from the paragraph.
- Sample downstream tasks: sentiment analysis, text summarization, etc.



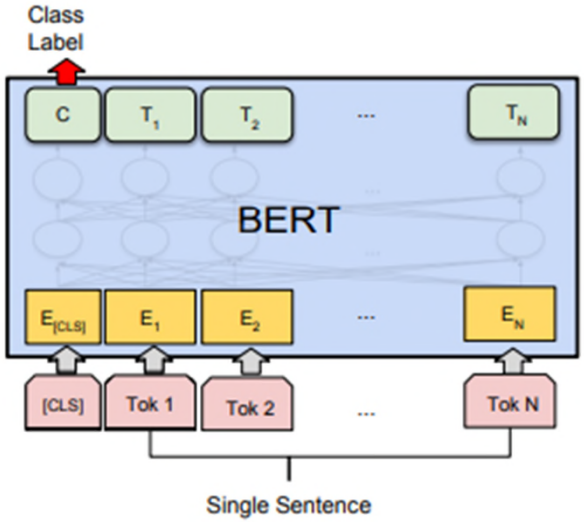
Source: Jacob Devlin et. al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding"

Performance Evaluation

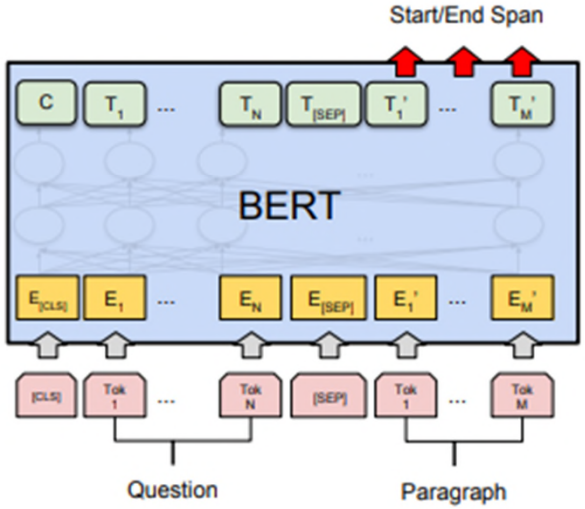
- General Language Understanding Evaluation (GLUE)
  - Sentence pair task.
  - Single sentence classification task.
- Stanford Question Answering Dataset (SQuAD)



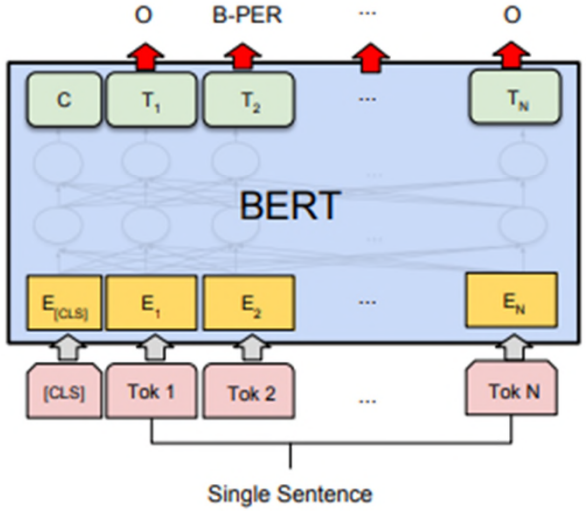
(a) Sentence Pair Classification Tasks: MNLI, QQP, QNLI, STS-B, MRPC, RTE, SWAG



(b) Single Sentence Classification Tasks: SST-2, CoLA



(c) Question Answering Tasks: SQuAD v1.1



(d) Single Sentence Tagging Tasks: CoNLL-2003 NER

Performance

- Outperform many task-specific models.

System	MNLI-(m/mm) 392k	QQP 363k	QNLI 108k	SST-2 67k	CoLA 8.5k	STS-B 5.7k	MRPC 3.5k	RTE 2.5k	**Average** -
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
$\text{BERT}_{\text{BASE}}$	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
$\text{BERT}_{\text{LARGE}}$	**86.7/85.9**	**72.1**	**92.7**	**94.9**	**60.5**	**86.5**	**89.3**	**70.1**	**82.1**

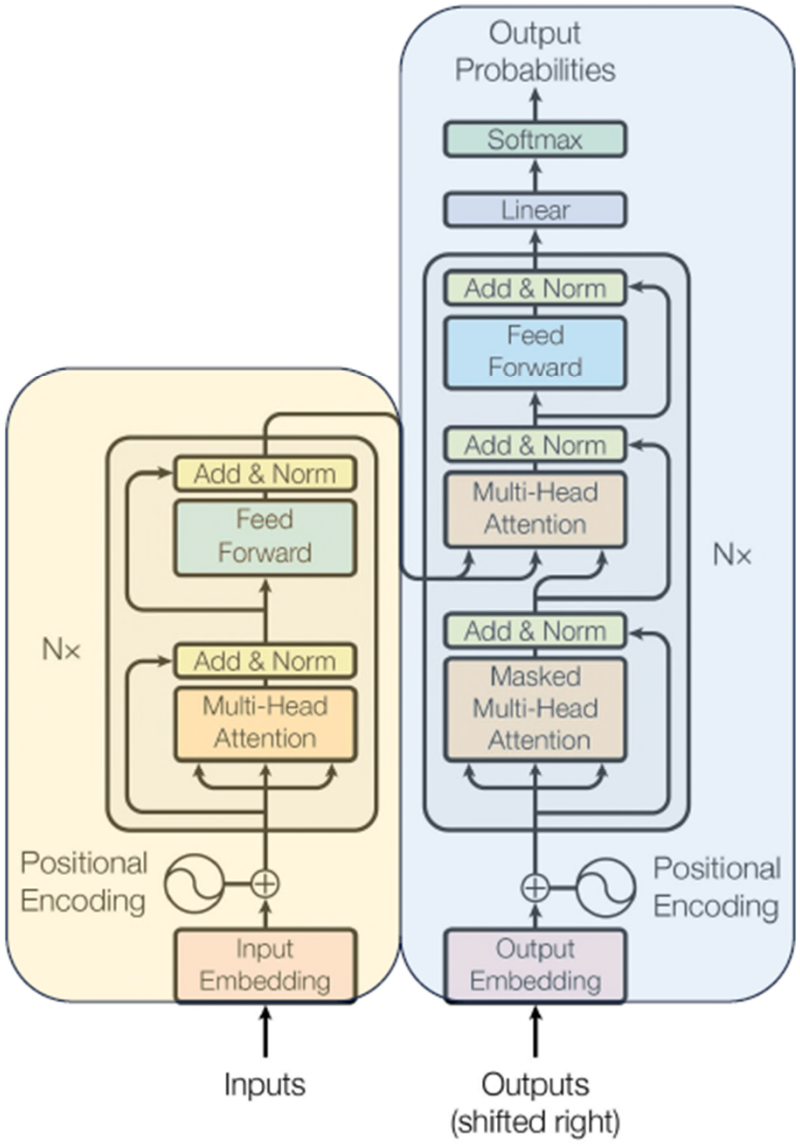
Table 1: GLUE Test results, scored by the evaluation server (https://gluebenchmark.com/leaderboard). The number below each task denotes the number of training examples. The "Average" column is slightly different than the official GLUE score, since we exclude the problematic WNLI set.[8] BERT and OpenAI GPT are single-model, single task. F1 scores are reported for QQP and MRPC, Spearman correlations are reported for STS-B, and accuracy scores are reported for the other tasks. We exclude entries that use BERT as one of their components.

Generative Pretrained Transformer (GPT)

Transformer: BERT + GPT

BERT
Oct 2018

Representation



GPT
June 2018

Generation

Source: CMU Course11-785 Introduction to Deep Learning

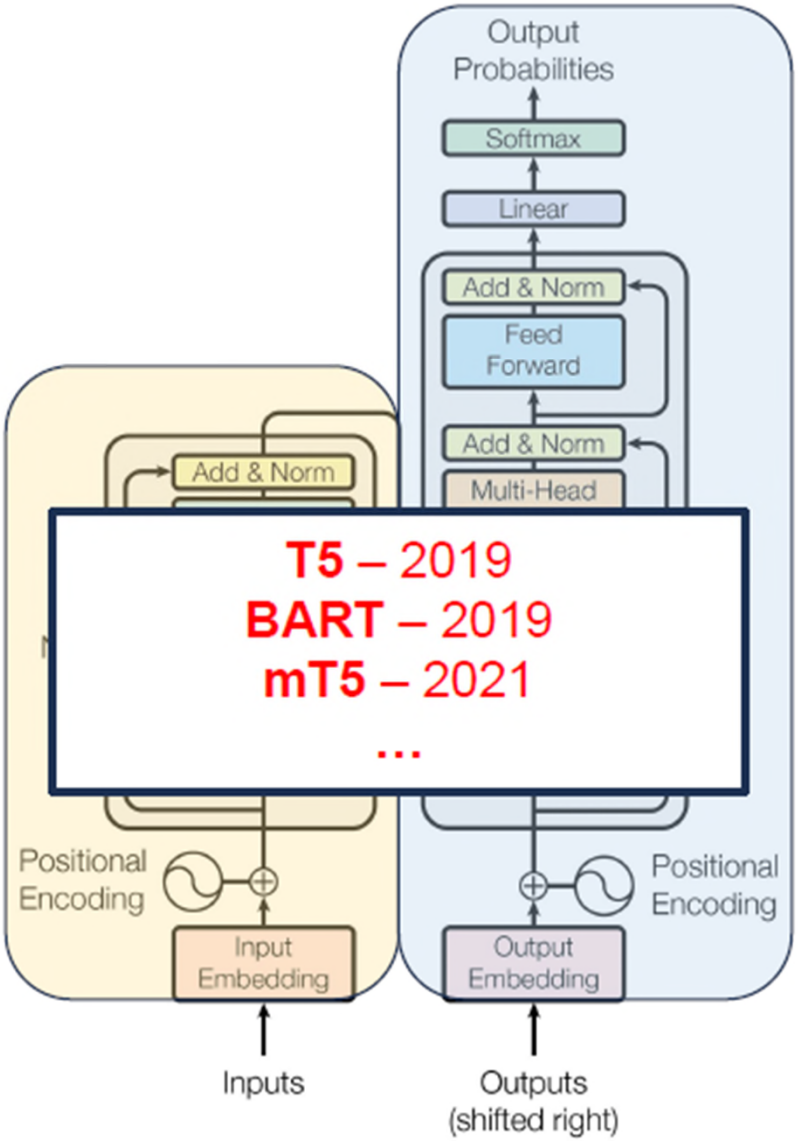
What is GPT?

- Generative Pre-trained Transformer (GPT) is an AI model pre-trained on vast amounts of texts to learn to generates human-like text.
- It leverages self-supervised learning to use large amounts of unlabelled data to pretrain an LLM.
- Sample applications: text generation, translation, and conversation, making it highly versatile for various language applications.
- A transformer decoder.
- GPT pretraining corpus: English Wikipedia & book corpus.
- GPT pretraining: predict the next token from the previous tokens.
- GPT fine-tuning: task-specific prompt text for the model to learn.

Recent LLM Development

BERT – 2018
DistilBERT – 2019
RoBERTa – 2019
ALBERT – 2019
ELECTRA – 2020
DeBERTa – 2020
…

Representation



GPT – 2018
GPT-2 – 2019
GPT-3 – 2020
GPT-Neo – 2021
GPT-3.5 (ChatGPT) – 2022
LLaMA – 2023
GPT-4 – 2023
…

Generation

Source: CMU Course11-785 Introduction to Deep Learning

LLM Architectures

- Encoder-only (e.g., BERT)
  - Pre-training: Masked Language Modelling (MLM), Next Sentence Prediction (NSP).
  - Good for classification tasks.
- Decoder-only (e.g., GPT)
  - Pre-training: Next word / token prediction.
  - Good for generation tasks.
  - Better generalization after pre-training.
- Encoder-decoder (e.g., Text-to-Text Transfer Transformer or T5)
  - Pre-training : provide text as input and train it to generate target text.
  - Good for tasks like machine translation.

Well-Known LLMs

GPT-4o

- Developer: OpenAI
- Parameters: More than 175 billion (likely trillions)
- Access: API
- Released in May 2024.
- Previous family iterations: GPT-1, GPT-2, GPT-3, GPT-3.5
- Extend multimodal capabilities of GPT-4 Turbo by integrating text, image and audio prompts.
- Process audio in real time, and output realistic, tone appropriate response in human voice.

Llama 3

- Developer: Meta (parent company of Facebook & Instagram)
- Parameters: 8 billion, 70 billion, and 400 billion (unreleased)
- Access: Open
- Released in Apr 2024.
- Previous family iterations: Llama, Llama 2.
- Can match the performance of GPT-4, but at lower cost.
- Able to install on a local system rather than relying solely on the cloud, 8B version of LLaMA 3 is small enough to run on a laptop.
- Alleviate privacy concern of sending data into the cloud for processing.
- Ideal for AI researchers due to its performance, adaptability, and open-source license.

Gemini

- Developer: Google
- Parameters: Nano (1.8 billion and 3.25 billion); others unknown.
- Access: API
- Released in Apr 2024.
- Formerly known as Bard.
- Three models: Gemini Nano, Gemini Pro, and Gemini Ultra, designed for different devices, from smartphones to servers.
- Can handle images, audio, video, code, and other kinds of information.

ChatGPT

GPT-4o

- More natural human-computer interaction.
- Accept as input any combination of text, audio, image, and video and generate any combination of text, audio, and image outputs.
- A single end-to-end model across text, vision, and audio.
- Fast: respond to audio inputs in average 320ms, similar to human response time in a conversation.
- Improve on non-English languages.
- Better at vision and audio understanding compared to existing models.

Source: openai.com

GPT-4o Voice & Vision

GPT-4o

Source: openai.com

GPT-4o Math Problem Solving

GPT-4o

Source: openai.com

GPT-4o Coding Assistant

Source: openai.com

Text-to-Image Generation

GPT-4o

1

Input

An image depicting three cubes stacked on a table. The top cube is red and has a G on it. The middle cube is blue and has a P on it. The bottom cube is green and has a T on it. The cubes are stacked on top of each other.

2

Output (1 / 7)

2

Output (2 / 7)

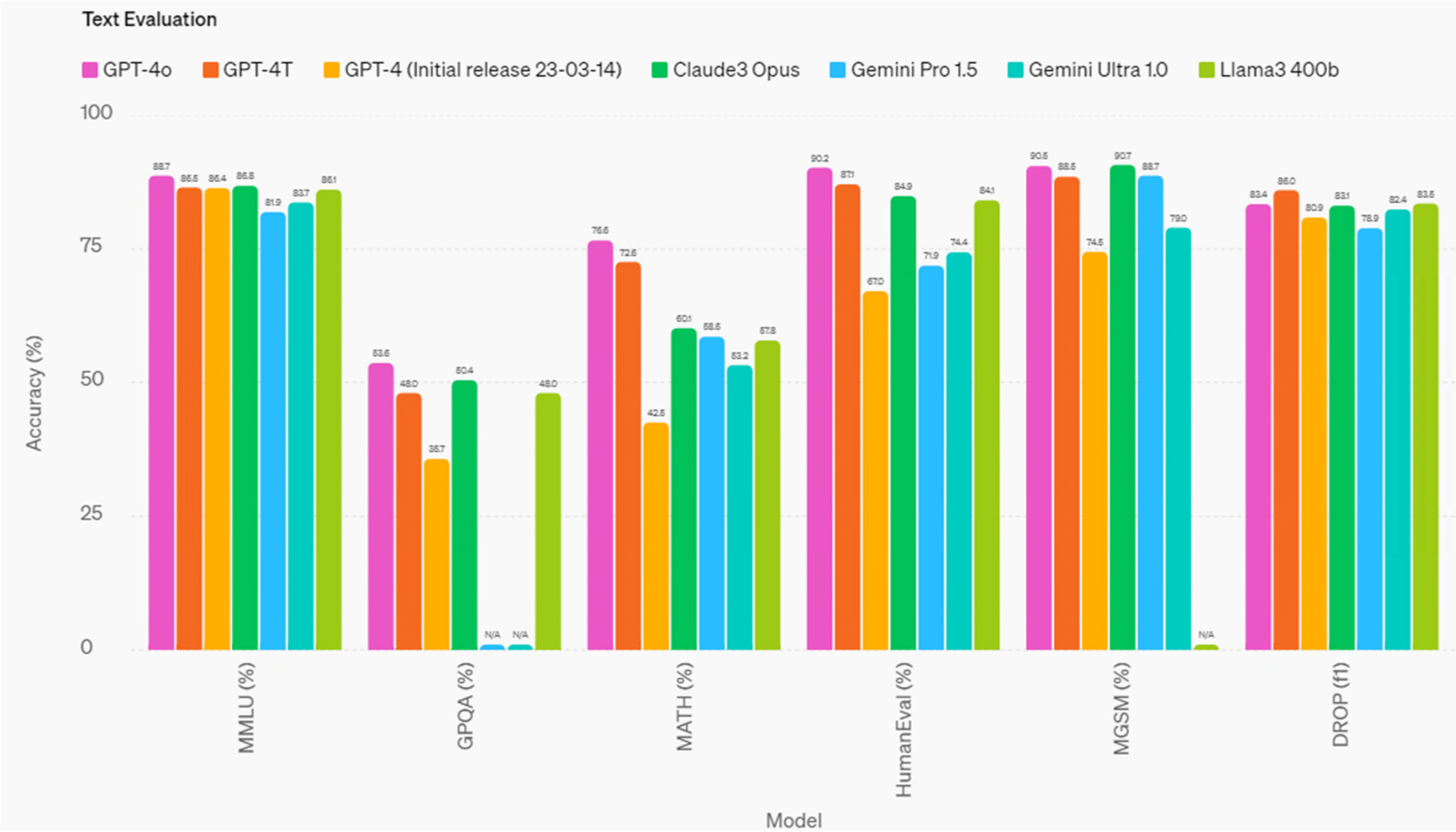
2

Output (3 / 7)

Source: openai.com

Performance Comparison

GPT-4o



Source: openai.com

Emerging Trends

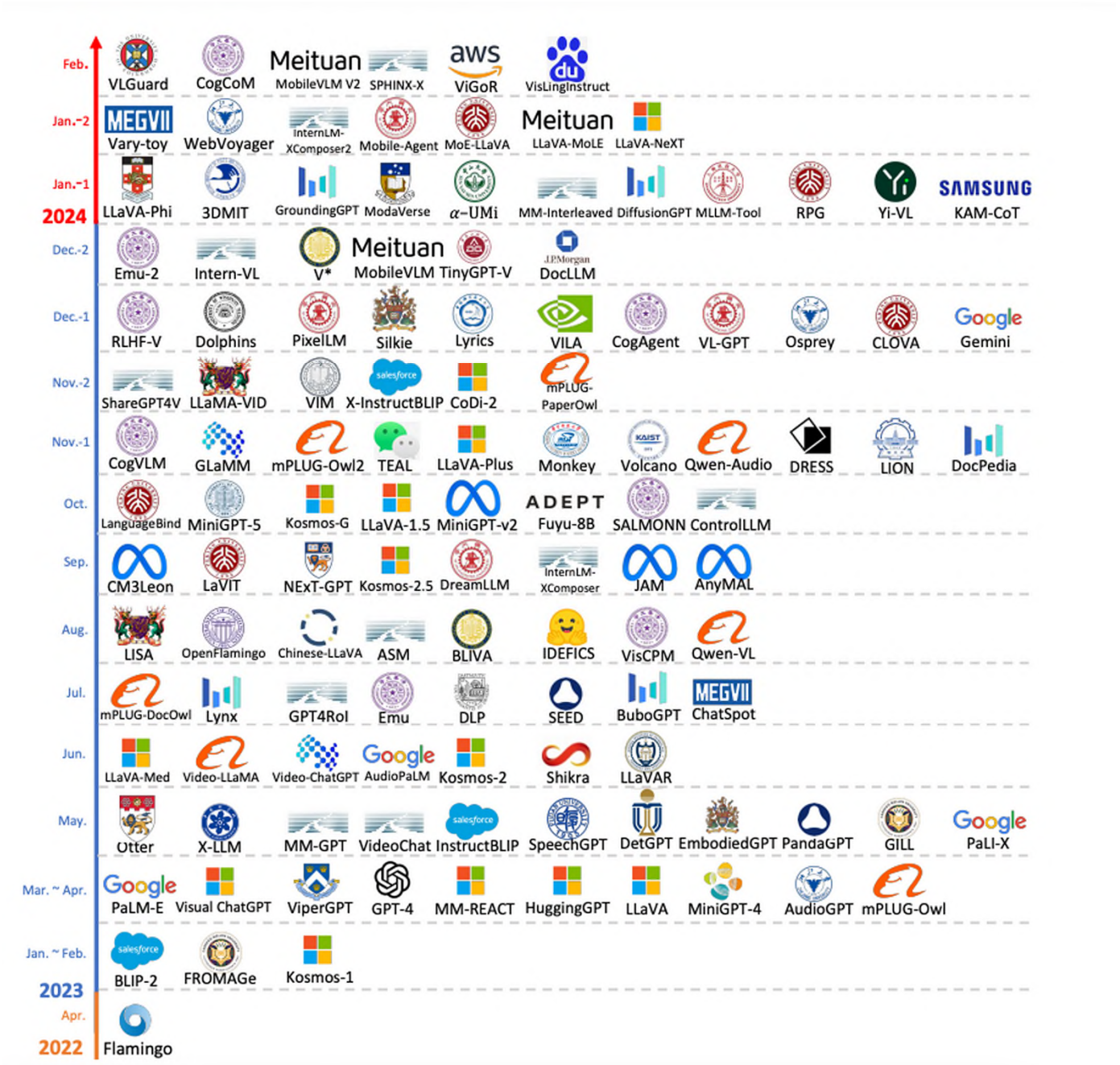
OpenAI o1 Reasoning

Source: openai.com

OpenAI o1 Coding

Source: openai.com

Emerging Trend: Multimodal LLM (MLLM)



What is Multimodal LLM (MLLM)?

- An MLLM is a model capable of understanding and generating different data modalities, such as text, images, audio, video, etc.
- Leverage multimodal inputs and outputs
  - Multimodal Input: process different data modalities (e.g., text and images).
  - Multimodal Output: generate outputs in multiple formats (e.g., text generation, image creation, etc.).
- Cross-modal Understanding
  - Enable model to link information from different modalities, such as describing an image with text or answering questions based on an image.
- Training with Diverse Data
  - Trained on vast datasets with different media types (text, images, etc.) to learn associations between them.
- Improved Contextual Understanding
  - Provide richer context and understanding than unimodal models due to multimodal integration

Multimodal LLM (MLLM)

- Applications:
  - Visual Question Answering (VQA)
  - Image Captioning
  - Text-to-Image or Image-to-Text Generation
  - Audio-Visual Processing
- Example Models:
  - GPT-4
  - LLaVA

Multimodal LLM (MLLM)

Sources: https://www.youtube.com/watch?v=AacWOBBaAow

Section II Summary

- The section covers the following topics:
  - Introduction
  - BERT
  - GPT
  - Well-Known LLMs
  - Emerging Trends

Part 6 Summary

- The part covered the following topics:
  - Section I: Foundation Models (FMs)
  - Section II: Large Language Models (LLMs)