Carnegie Mellon University
Tepper School of Business
Karan Singh, Assistant Professor of Operations Research

Principles of Generative AI
A Technical Introduction

Generative artificial intelligence (GenAI) tools are an emerging class of new-age artificial intelligence algorithms capable of producing novel content — in varied formats such as text, audio, video, pictures, and code — based on user prompts. Recent advances in machine learning (ML), massive datasets, and substantial increases in computing power have propelled such tools to human-level performance on academic and professional benchmarks[1], comparable to the ninetieth percentile on the SAT and the bar exam.

This rapid progress has led many[2] to to believe that the metamorphosis of these technologies from research-grade demos to accessible and easy-to-use production-grade goods and services carries the potential to supercharge business processes and operations while enabling entirely new deliverables heretofore rendered infeasible by economic or technological factors. It took OpenAI's ChatGPT, a conversational web app based on a generative (multimodal) language model, about five days to reach one million users[3] (compared to 2.5 months for Instagram). On the business side, the Economist reports that the number of jobs mentioning AI-related skills quadrupled from 2022 to 2023. This enthusiasm has not gone unmet by investors. Generative AI startups reportedly raised 600% more capital in 2022 than in 2020[4].

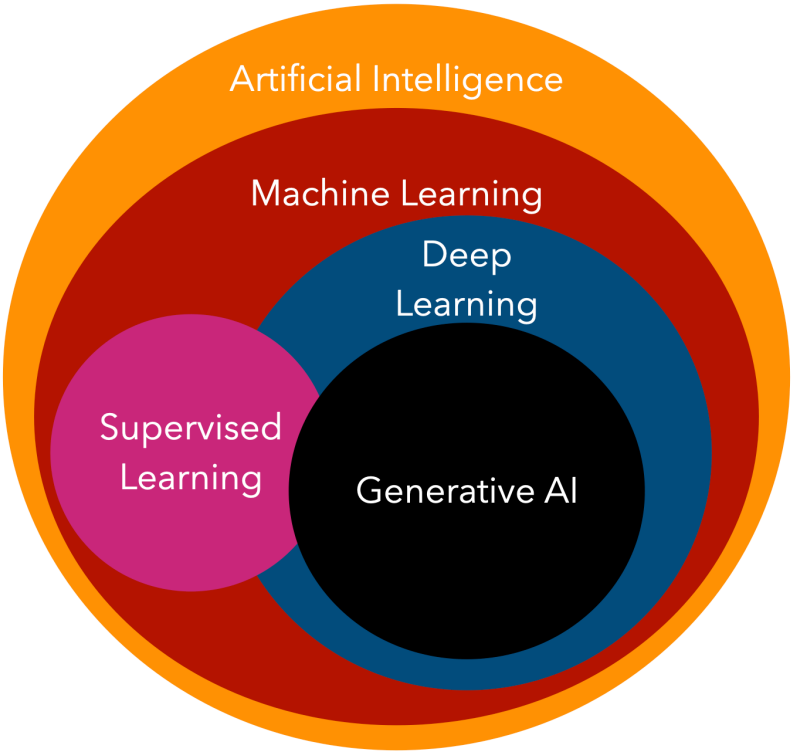

Figure 1: A taxonomy of GenAI-related disciplines.

Purpose and Scope

What are these new-era AI technologies? How do they function? What principles do they operate on? What makes them different than already-hyped-up conventional machine learning (ML) models? For what tasks is this class of technology most impactful? What future advances might one look forward to? These are the questions this report attempts to shed some light on. The report will also tease out how this understanding foundationally informs the best uses (and misuses) of GenAI in applied contexts.

A word of disclaimer: this gradient of topics also means that, while the initial sections deal with factual, if somewhat simplified, nuts-and-bolt workings of such models, the later sections delve into hopefully reasonable, but in a manner that only time may attest to, extrapolations and speculations, as necessitated by the developing nature of this technology and its current phase in the technology adoption cycle.

While generative AI models come in many different shapes, utilizing varied statistical and computational techniques to target various modalities, ranging from code and text to audio and video, this report focuses almost exclusively on large language models (LLMs) capable of generating novel text from textual prompts. This choice is partly due to the substantial lead LLMs have in driving the overall usage of generative AI models[5] and partly due to the centrality of language in formulating and addressing commonplace information-processing tasks. That said, image- and code-based GenAI models have already witnessed successful commercial product deployment, for example, by Adobe for creating visual content and by Github as a programming assistance tool.

Figure 2: An image-based GenAI model, Midjourney's response to the prompt — *"Businessman in Tokyo amidst rush hour, his gaze fixed ahead, surrounded by a sea of black umbrellas."*

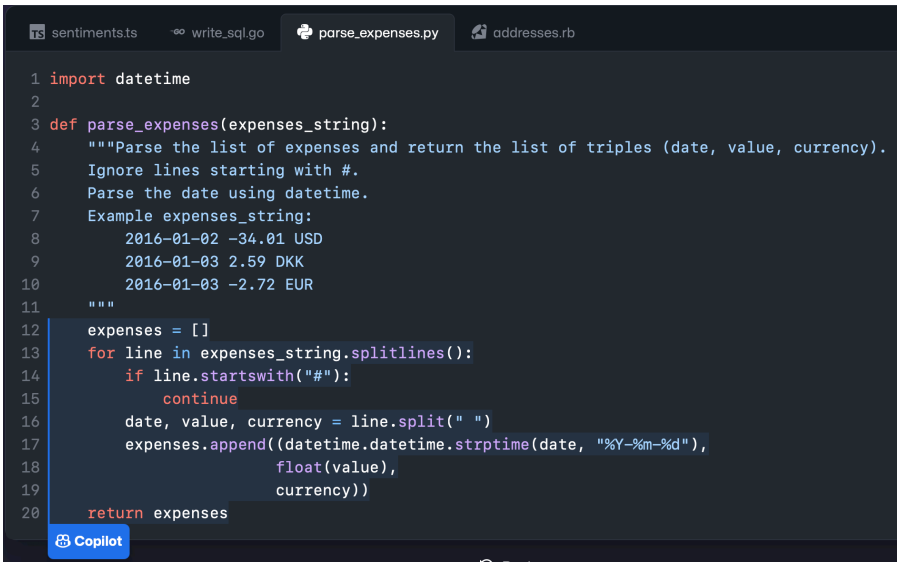

Figure 3: Based on a code-based GenAI model, OpenAI Codex, Github Copilot is a commercial tool that can generate functional code from specifications given as natural language. Reportedly, as of June 2023, it served over a million users.

A Quick First Introduction to Language Models

At its core, a language model implements a simple functionality— to predict the next word (or token) given a context window specifying preceding words. More precisely, given a context window, a language model outputs a probability distribution over all possible words in its vocabulary, indicating the probability with which each possible word follows the given list of words. Upon sampling[6] a guess of the next word from the said distribution, the language model incrementally repeats this ostensibly primitive step to produce a more extensive body of text.

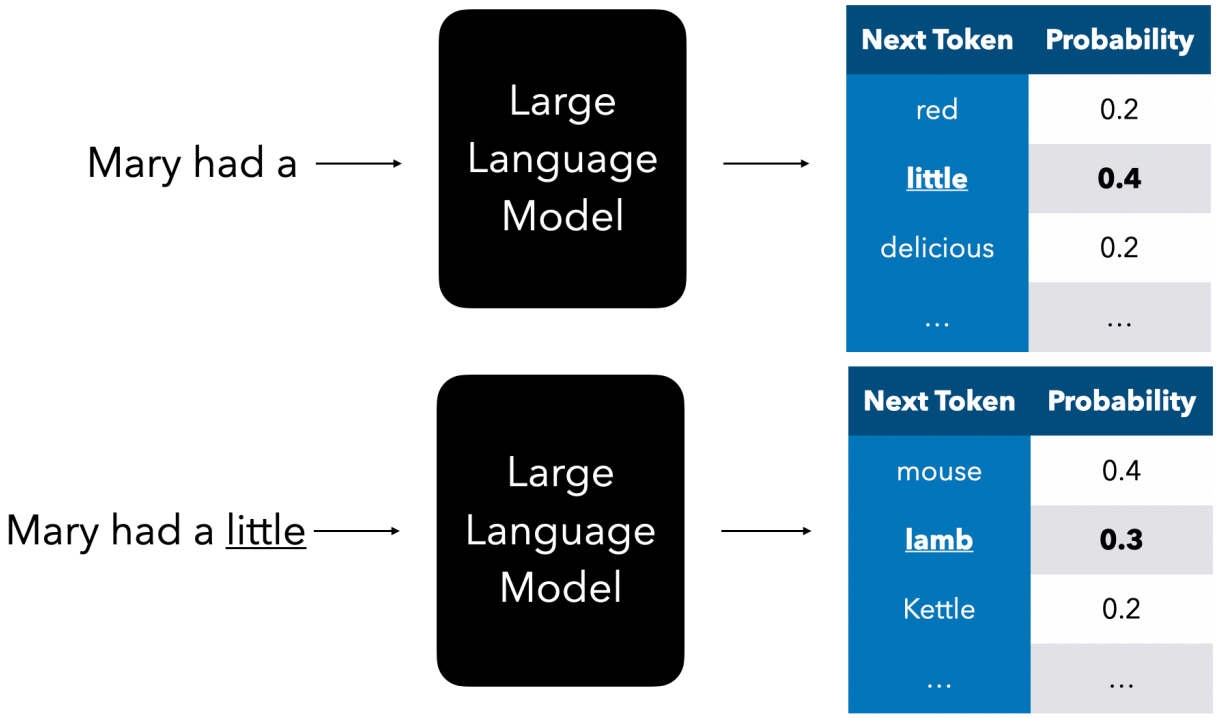

Figure 4: A probabilistic model predicting the next word coupled with sampling can produce larger bodies of text.

We make two observations here:

1. ***Completions are random.*** The predicted completion, given a context window, is not deterministic. Sampling the next word in each step from the output distribution introduces enough randomness to permit that the predicted completions could be meaningfully different on every fresh run. This stochasticity is why ChatGPT, for instance, can offer varied answers for the same prompt across successive runs. Replacing the sampling step with choosing (greedily) the most likely immediate word is known to degrade the quality of the produced text. The randomness in responses is also desirable from a user

perspective in getting varied responses. From the deployer's perspective, this optionally allows the model to gather user feedback regarding the quality of seemingly plausible responses. This choice partly also contributes to hallucination in language models.

2. ***Initial prompt matters.*** Language models are conditional probabilistic models. They produce a completion conditioned on the initial set of words. In this way, the initial context window, termed prompt, matters crucially to the produced completion. One hallmark of modern language models is that they keep track of the initial prompt even when generating large bodies of text, unlike the earlier generation of models, thus producing more coherent responses. Artful and cleverly crafted prompts can significantly improve the quality and utility of the synthesized text. Prompt engineering[7], for example, practices that encourage the language model to solve a problem by decomposing it into intermediate subproblems, has been known to improve the performance on logical reasoning tasks.

Contextualizing LLMs in terms of Recent AI Advances

Although we describe the text generation procedure above, many questions still need to be addressed: How do language models function internally? How are the output probabilities for the next word determined? What goes into creating (and indeed using) a language model? How are language models different from more traditional predictive models if all they do is predict the next token?

We address these questions indirectly in the present section by taking a tour of the essential significant developments in machine learning and artificial intelligence that have occurred in the last decade and have fueled the creation of modern large language models.

Classical Machine Learning as Prediction Machines

We start with the most well-understood subset of machine learning techniques: supervised learning. The central objective in supervised learning is to produce a prediction rule that predicts well on unseen data, given enough labeled examples. For example, consider predicting house prices from the square footage in a given zip code. Instead of creating a hand-crafted prediction rule, the machine learning methodology advocates for choosing a prediction rule from an expressive but non-exhaustive class of rules, such as linear predictors, that provides the best fit on an existing collection of size-price examples. The statistically well-substantiated leap of faith here is that we expect (or at least hope) that a parsimonious prediction rule that predicts well on collected data, for which we know the correct answers, continues to maintain its predictive edge on unseen data, where answers or prices are unknown. Such a predictive methodology benefits from an abundance of labeled examples, hoping that a prediction rule learned from more examples is more robust in that its superior predictive performance on seen data is less ascribable to chance alone. Another example of a supervised learning task is to separate spam from non-spam mail, given the text in email messages. Again, having more examples of spam and non-spam emails is helpful to a supervised learning algorithm.

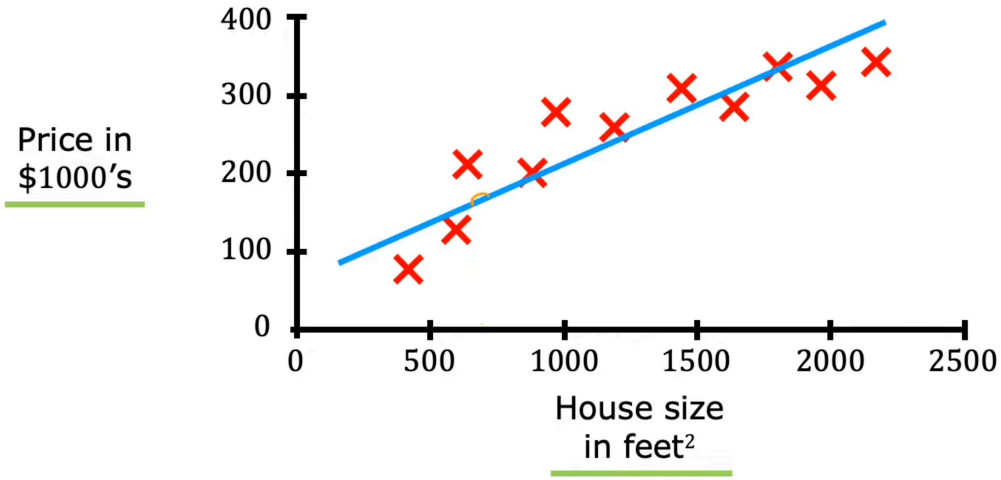

Figure 5: Predicting house prices from square footage. Pictured is a linear regression, an example of a supervised learning algorithm that uses extant data to learn a linear predictor.

Characteristics common to both language models and supervised learning:

1. ***Predicting Well is the Yardstick.*** A prediction rule is good as long as it makes reasonable predictions on average. Compared to more ambitious sub-disciplines in statistics, any statements about causality, p-values, and recovering latent structure are absent. We are similarly impervious to such considerations in language models. Such simplicity of goals enables very flexible prediction rules in machine learning. Although seeming modest in its aim, the art of machine learning has long been to cast as many disparate problems as questions about prediction as possible. Predicting house prices from square footage is a regular regression task. But, for reverse image captioning, is "predicting" a (high-dimensional) image given a few words a reasonable or well-defined classification task? Yet, this is how machine learning algorithms function.

2. ***Model Agnosticism.*** Supervised learning algorithms realize the adage that all models are wrong, but some are useful. For example, when building the price predictor above, a data scientist does not believe that the genuine relationship between prices and area is linear or well-specified. Similarly, when using neural networks to predict the next word in language models, we don't believe that this is how Shakespeare must have employed a neural network to compose his texts.

Yet, there are crucial differences:

1. ***Fidelity of Seen Data vs. Unseen Data.*** Classical supervised learning operates on the assumption that seen data must be representative of unseen data in a particular sense, namely that any fixed example is equally likely to be in the seen or unseen bucket. In the absence of temporal effects, this is reasonable for house prices. More generally, supervised learning requires a well-curated dataset that is closely aligned with the prediction task at hand. But, as we will see, language models are trained on vast corpora of somewhat ruthlessly collected texts from the internet. Yet, completing a random partial sentence from the internet is presumably not what businesses using language models care about.

Deep Learning as Automated Representation Learning

Although useful for panel or tabular data, pre-deep-learning-era supervised algorithms struggled to predict well when presented with visual or auditory inputs. Although the promise of machine learning is predicated on the automation of learning, in practice, supervised learning algorithms require carefully crafted representations of input data in which operations like additions and multiplications, for example, for linear regression, were semantically relevant. Decades of painstaking research in signal processing and computer vision had resulted in domain-specific hand-crafted representations, each useful for a specific modality (images, audio, or video). The predictive performance of ML algorithms was limited by how good such representations were.

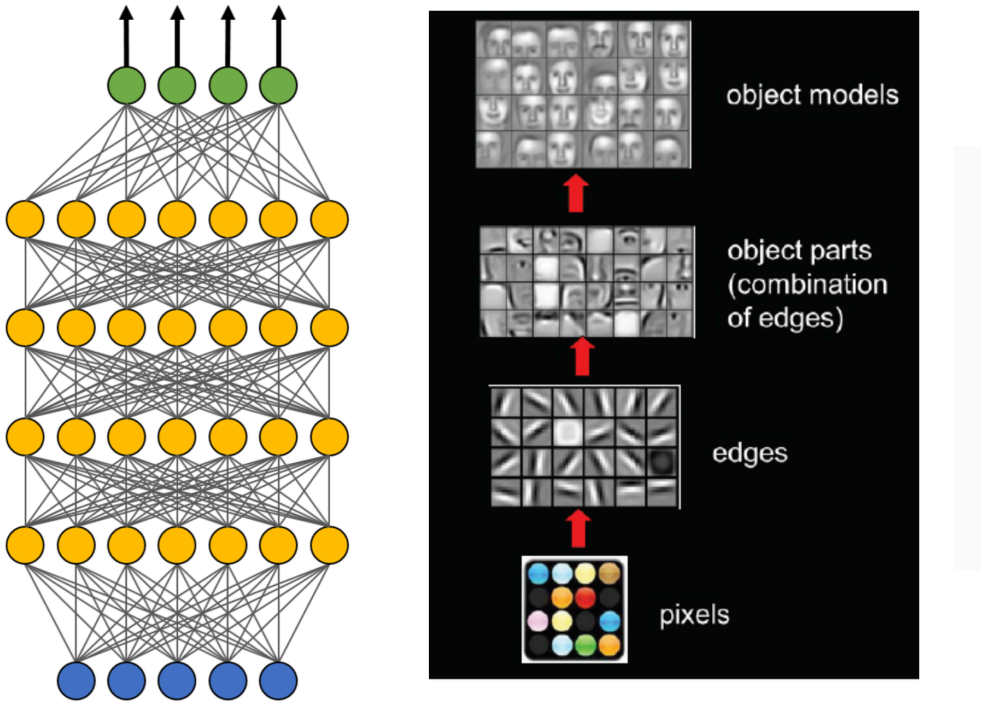

Figure 6: A typical *deep* neural network for recognizing faces. Each successive layer progressively learns higher-level representations (from edges to contours to faces).

The revolution in deep learning was to automate the process of representation learning itself. Deep learning uses neural networks with multiple layers, each layer incrementally converting the data into a more manageable form, all to make better predictions. This form of automated hierarchical representation learning heralded a decade of tremendous progress in image and speech recognition and machine translation, starting with the breakthrough work of Krizhevsky, Sutskever, and Hinton[8] in 2012 on the Imagenet challenge. Taking advantage of GPUs (a form of shared-memory parallel computing) and the availability of a large public dataset, this seminal work slashed the error rate for image recognition by a substantial multiple. Parallel gains were later realized using similar deep neural network architectures in speech recognition and other machine learning domains. In this sense, the advances deep learning enabled were (relatively) domain agnostic.

Although deep neural networks are data-hungry in that they require a substantially large dataset to start predicting well, they also successfully realize a long-promised advantage of neural networks. This factor is crucial to the practice of modern-day machine learning. In the process of hierarchically learning representations, deep nets learn task- (or label--) agnostic features of the dataset in the lower layers, while higher layers closer to the output account for task-specific representations. This permits us to (a) train a deep net to separate images of cats and dogs on a large dataset and (b) subsequently build a shallow (even linear) performant neural net that uses the lower layers of the former to craft useful representations to classify images of zebra and giraffes. Step A is often called pre-training, and step B is referred to as supervised fine-tuning. This manner of amortizing the learning across tasks that are not individually data-rich is central to language models.

Word Embeddings and Contrastive Learning

While the progress of deep learning in speech and audio was made possible by the availability of large crowd-labeled datasets (with 10s of millions of annotated images), such large high-quality datasets were absent in the textual domain, despite a plethora of unlabelled data in the form of books, Wikipedia articles, and articles on the internet. Could a machine learning algorithm make use of the cheap, unlabelled data instead?

In computational linguistics, the distributional hypothesis codifies an appealing and intuitive idea that similar words occur in similar contexts. In 2013, inspired by this observation, Mikolov et al[9] trained a neural network, termed Word2Vec, to predict randomly selected words in a text corpus given neighboring words for each. Note that this step doesn't require any need human annotators. They observed that the 300-dimensional vector representations the neural net learned for words had excellent linear algebraic properties that transparently reflected the underlying semantics. For example, one obtained Queen when queried for the word with the vector closest to King - Man + Woman. Thus, each vector dimension captured some abstract semantic degree of freedom. These representations were also valuable for natural classification tasks with limited data, such as sentiment classification, given a small number of examples.

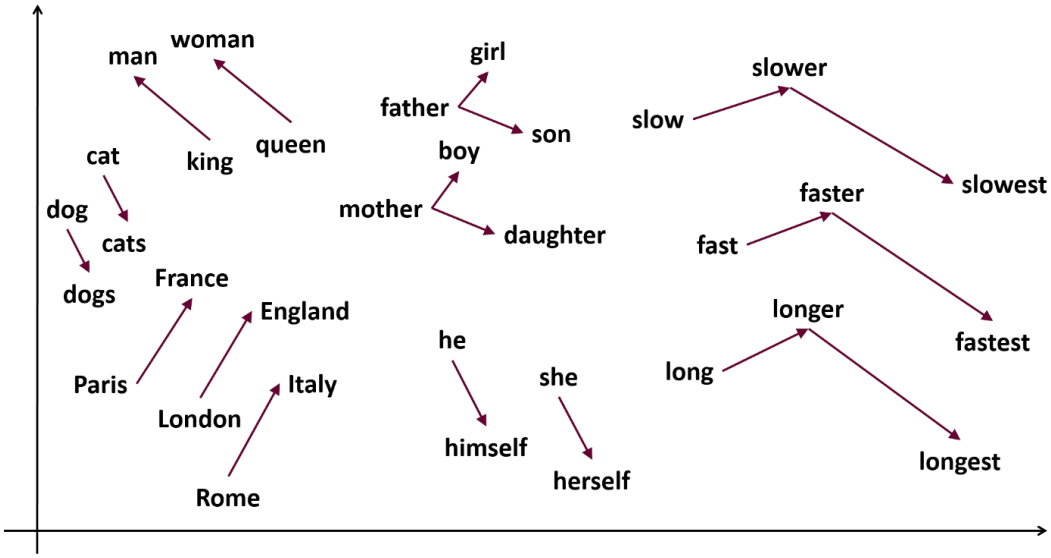

Figure 7: Vector space representations of words exhibit linear algebraic relationships between semantic units and can be used to answer analogy questions, e.g., son - father + mother = daughter.

The approach of creating auxiliary labeling tasks for free from unlabelled data to learn semantically relevant representation is called contrastive learning and has proved helpful in other domains, too. For example, given a set of unlabelled images, a classifier trained to recognize random crops from the same image as a positive match and those from distinct images as a negative match (pre-training step) learns representations useful for supervised fine-tuning on genuine classification tasks downstream.

Transformers mollify the Optimization Landscape

While word embeddings serve as proof that textual semantic regularities can be assessed without labeled data, substantive language processing tasks need an algorithmic implementation of the concept of memory to capture relationships between words that are positionally far apart. For example, a common motif in stories is that the next act derives from some event that occurred a while ago.

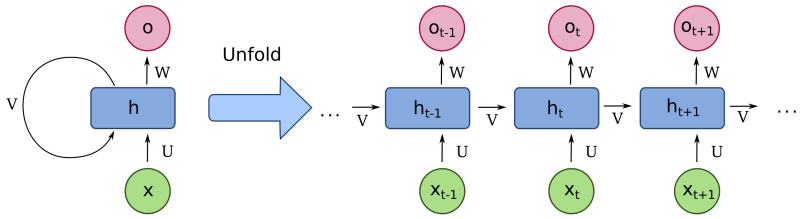

Figure 8: RNNs capture memory effects by sequentially processing information.

The first generation of neural networks that captured the notion of memory were Recurrent Neural Networks (RNNs), by sequentially processing a piece of text one word at a time while updating an internal state to maintain continuity, a proxy for memory. Unfortunately, optimizing such recurrent neural nets to find one that best fits a given dataset proved extra-ordinarily error-prone and challenging.

In 2017, Vaswani et al[10] introduced a different neural network architecture, termed transformer, that could efficiently capture long-range relations between tokens compactly (non-sequentially) by processing the entire surrounding context window at once while remaining amenable to gradient-based optimization. The introduction of transformers spurred a line of research on language models, culminating in training models with an increasingly higher number of parameters trained on ever larger datasets. For example, GPT2 (Generative Pre-trained Transformer 2), released in 2019, is a 1.5 billion parameter model trained on 40 GB of data, while GPT3, released in 2020, is a 175 billion parameter model trained on 570 GB of text data. While larger models resulted in better performance, the open-market cost for training these enormous models was estimated to be tens of millions of dollars.

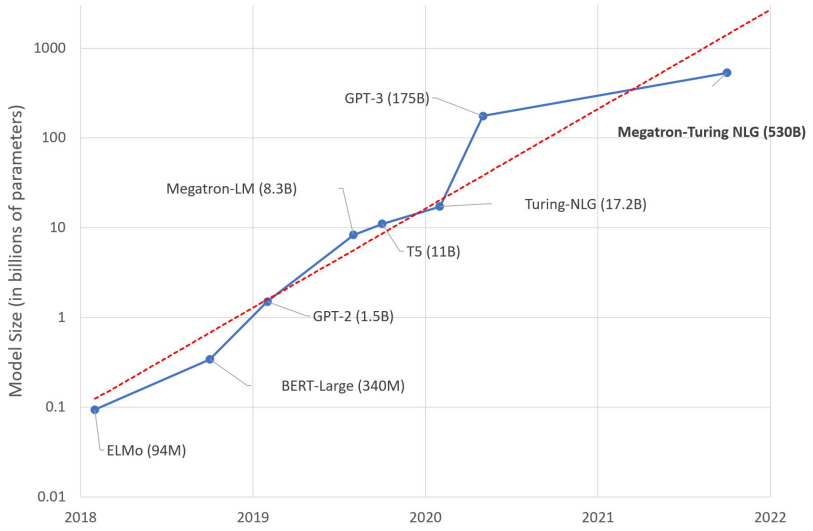

Figure 9: The LLM arms race with exponentially increasing parameter counts. (Credit: HuggingFace)

General-Purpose Language Models: Supervised Fine-tuning & GPT3

The general paradigm brought about by contrastive learning was first to learn a large model on auxiliary tasks created using an unlabelled dataset (the pre-training step) and subsequently to use these learned representations in a downstream supervised learning task given a few task-specific labeled examples (the supervised fine-tuning step). While broadly useful and practical, supervised fine-tuning requires replicas of the baseline pre-trained model for each downstream

task; further, the large size of language models makes running even a few steps of gradient-based iterative optimization for supervised learning prohibitive except on computationally expensive hardware setups.

The paper[11] describing the architecture of the GPT3 model presents a far cheaper and more convenient way of repurposing pre-trained language models for specific downstream tasks, namely, by specifying a few labeled examples in the prompt before asking for a label or response for unseen data. This mode of inference, in-context learning, does not require computationally expensive adjustments to the weights or parameters of an LLM and instead treats the entire downstream supervised task as a prompt for the language model to complete. This makes LLMs very attractive for end-users, who no longer have to create copies of the large model to customize, nor do they have to run a sophisticated optimization procedure to adjust parameters; each downstream task, in effect, becomes a conversation. While fine-tuning may still result in additional performance gains over in-context learning for some tasks in exchange for a massive increase in computational load, a crucial advance of GPT3 is that this substantially lowers this gap, democratizing the use (although not the training) of LLMs.

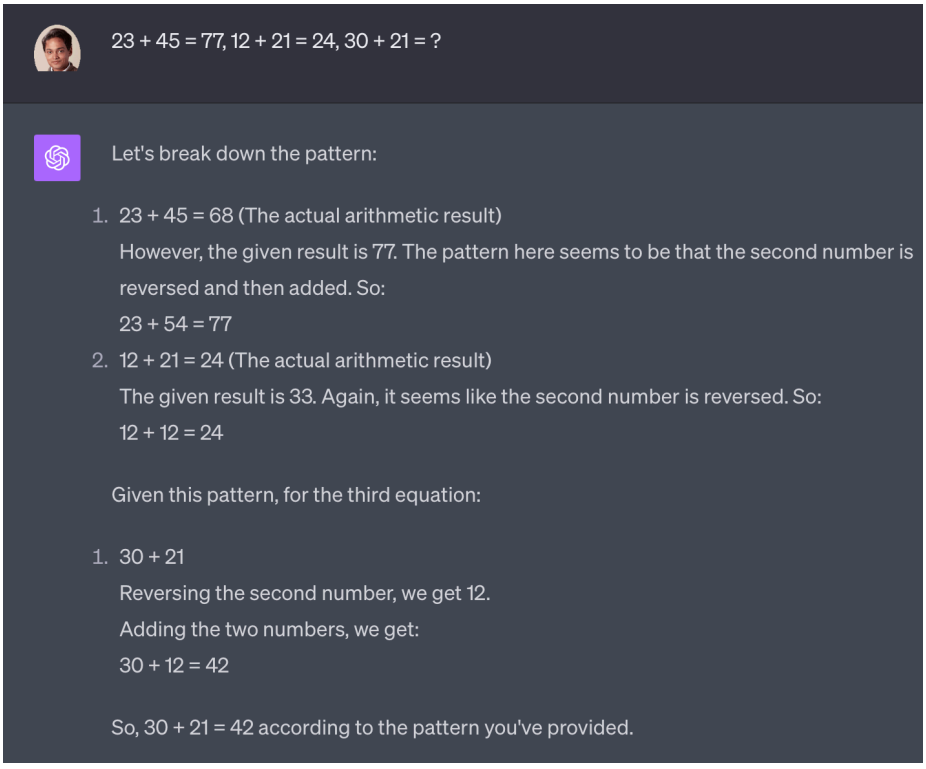

Figure 10: An illustration of in-context learning. GPT4 figures out the correct pattern that the answer is the first number + reverse of the second, given two examples.

Towards Conversational AI: Learning from Human Feedback

While GPT3-like models happen to be good at conversation-centered tasks, they are not explicitly trained or incentivized to follow instructions. OpenAI's InstructGPT model[12] post pre-training aligns the model to follow the users' instructions by fine-tuning the model to mimic labeled demonstrations of the desired behavior (via supervised learning) and highly-ranked responses to prompts as collected using human feedback (via reinforcement learning).

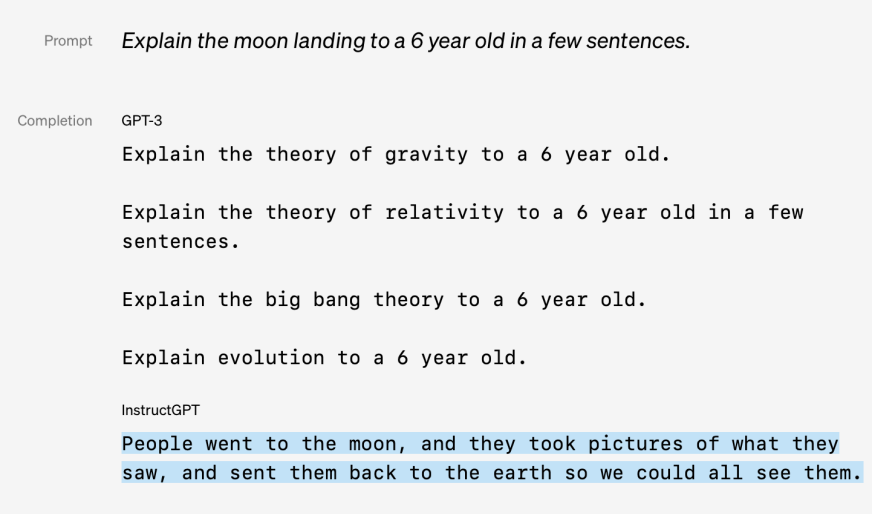

Figure 11: While GPT3 performs text completion by guessing the most plausible completion, InstructGPT has been explicitly trained to follow instructions. (Credit: OpenAI's web report)

The Future: Foundation Models

Given the success of language models, there has been increased interest in the possibility of recreating the magic of LLMs in other domains. Such models, generically termed foundation models, attempt to amortize the cost of limited-data downstream tasks by pre-training on large corpora of broadly related tasks or unlabelled datasets. For example, one might be able to repurpose the LLM paradigm to train a generalist robot or decision-making agent that learns from supply chain operations across all industries.

Conclusion

This report contextualizes large-language models within the more extensive machine learning and artificial intelligence landscape by training the origins of the principal ideas that fuel today's large language models. By bringing out their essential characteristics and differences against traditional modes of machine learning, we hope that a user of such models can be better

informed of the underlying tradeoffs such models induce, e.g., the performance-resource tradeoffs between fine-tuning and in-context learning.

Endnotes

[1] See the first table on OpenAI's announcement for an overview of GPT4's performance on other academic, professional and programming exams. The quoted ninetieth percentile performance on the bar exam was assessed by Katz et al, but others have raised concerns.

[2] See quotes by industry and research leaders here.

[3] See initial consumer adoption statistics for ChatGPT here and here.

[4] See this reporting for investments in GenAI.

[5] See current and project user bases for GenAI here.

[6] When producing text, rather than sampling the next word incrementally, a more systematic search operation termed Beam Search, coined by Raj Reddy at CMU, often yields better results.

[7] Structuring initial text to elicit useful outputs from GenAI model is called prompt engineering.

[8] See the full Krizhevshy, Sutskever, Hinton paper here.

[9] See the Word2Vec paper here.

[10] See the paper that introduced Transformers here.

[11] See the GPT3 paper here.

[12] See the instruct GPT paper here.