Towards a Human-Centered AGI

Soumyadip Nandi

January 2026

1 Abstract

As current systems scale towards more general capabilities, performance gains on benchmarks are yet to translate into robust , long-horizon reasoning, coherent world models , or reliable alignment with human values. This gap suggests that Artificial General Intelligence (AGI) is not merely a question of scaling and subsequent emergent abilities of scaling, but of structure. In this paper, we argue that progress towards a human-centered AGI requires a multi-layered computational substrate that integrates technical design (the technical layer), epistemic reliability (epistemic layer) and human oversight (human layer) as co-evolving components of a single system. We introduce a layered perspective on AGI development with significant emphasis on the technical layer. This layer governs learning dynamics and representations. Here, architectural choices determine representational coherence, learning dynamics and long-term stability. We also analyze the limitations of scaling , the inability of models to continually learn and objectives of long-horizon planning and context. Drawing inspiration from cognitive science , we argue that human cognition should serve as an inductive prior rather than an engineering template that can shape representations, reasoning and accomodates hierarchial abstractions . Beyond technical performance, we situate the proposed framework with epistemic and human social contexts., highlighting risks such as misgeneralization, dependence on reward-centric optimization, etc. We contend that alignment cannot be achieved solely through post-hoc finetuning or preference optimization , but must be embedded structurally through continual human participation . Instead of presenting a single algorithmic solution to AGI, this paper extends foundational research agenda, arguing that human centeredness must be structurally embedded at the level of architectures, objectives, and evaluation metrics. By clarifying the usefulness of human values within a multi-layered framework, we aim to provide a principled foundation for developing AGI systems that are not merely powerful, but coherently aligned with human reasoning, values, and long-term interests. [1]

[1] Note from the Author:- A lot of the code that has been written with regards to Mechanistic Interpretability, chain-of-thought , LLM memory etc , BabyGPT has been used as a conceptual scaffolding, a project that I had done two years back. Link to those specific colab NoteBooks have been provided in those specific sections

Contents

1 Abstract **1**

2 Definitions of AGI **1**

3 Motivations for the Technical Layer **4**

3.1 The Brain on ChatGPT. . . . 6

3.2 On productivity . . . 6

3.3 Long-Horizon RL . . . 7

3.4 Does Scaling really work ? . . . 11

3.4.1 The Scaling Hypothesis . . . 12

3.4.2 Validation loss, Perplexity Score . . . 13

3.5 Continual Learning . . . 15

3.6 Context Rot . . . 17

3.7 Human Cognition as a Design Prior . . . 18

3.7.1 Disentanglement . . . 19

3.7.2 The Pre-Training Thesis . . . 19

3.7.3 Compression and Bottlenecks . . . 20

3.7.4 The Problem With Scaling . . . 21

3.8 Chain-Of-Thought . . . 22

3.8.1 Tokens During COT . . . 23

3.8.2 Priors in Transformers . . . 24

3.8.3 Implicit Priors . . . 25

3.9 RL Scaling . . . 26

3.9.1 Pre-Training Scaling for RL . . . 26

4 The Epistemic layer **27**

4.1 Uncertainty Modeling . . . 28

4.2 Interpretability Research . . . 29

4.3 Introduction To Mechanistic Interpretability . . . 30

4.3.1 Circuits features and visualizing weights . . . 30

4.4 Reverse Engineering in Mechanistic Interpretability . . . 31

4.5 The Curse Of Dimensionality . . . 32

4.5.1 Dictionary Learning . . . 33

4.6 Why is interpretability important for alignment ? . . . 34

4.6.1 Reward Hacking . . . 34

4.6.2 Gradient Hacking . . . 36

4.7 Can modern interpretability mitigate gradient hacking? . . . 37

4.8 Coherence Metric Formalism . . . 39

5 Introduction To The Human Layer **43**

5.1 What does RLHF do ? . . . 44

5.1.1 Training with RL . . . 45

5.1.2 RLHF Alignment . . . 46
5.2 Theory Of Mind . . . 48
5.2.1 ToM as a Consequence of Scaling . . . 49
5.3 Self Preservation Guardrails . . . 49
5.3.1 The Instrumental Convergence Thesis . . . 50
5.3.2 AI Control Problem . . . 50
5.4 Constitutional AI . . . 52
5.4.1 CAI and Alignment . . . 53
6 AI Memory **54**
6.1 How does today's AI memory actually work ? . . . 54
6.2 What an AGI memory can look like . . . 55
6.2.1 LLMs as core processors. . . . 55
6.2.2 AI Native Memory. . . . 56
7 Conclusion and Future Research Directions. **57**

2 Definitions of AGI

As we get closer to AGI, it is becoming less about a binary threshold but rather a spectrum. Since human intelligence also rests on a spectrum, it is essential that AGI is also treated as such. Richard Ngo in his blog [1] prefers to call a system t- AGI, if on most tasks it is able to beat humans experts on most cognitive tasks who are given time t to perform the task. In practice , it means that a 1-second AGI would need to beat humans at tasks like quickly answering trivia questions, basic intuitions about physics. A 1-minute AGI would be able to beat humans at exams, solving problem sets etc . Eventually, a 1- year AGI would be able to beat humans at everything. One of the more difficult problems to solve while trying to provide a substantial definition of AGI is **coherence**. We can thank evolution for optimizing coherence for us even though occasionally we still struggle with it. Coherence has also been a major bottleneck for LLMs. In principle, the longer an episode goes on, the more likely they are to go off the training distribution.

As a concrete model, it is almost always assumed that AGI is trained by a single large foundational model, using self supervised learning on (possibly multi-modal) data and then it is fine-tuned using model-free RL with a reward function that is learned from human feedback. Ngo et al. [2] . But one of the key elements that it ignores is continual learning , where the model lacks any ability to actually acquire knowledge.

Current frontier models almost entirely lack the ability to **continually learn**. A lack of situational awareness of AGI models can also end up becoming another drawback. To perform well on a range of real world tasks, policies would have to use knowledge of the wider world when choosing actions. Over time it is expected that most capable policies will become good at identifying which abstract knowledge is relevant to the policies themselves and to the context in which they are run. It also needs to be capable at applying that knowledge while choosing actions. This is a skill that Ajeya Cotra calls *self-reasoning*. [3] Since 2022, LLMs have shown early potential in situational awareness. As of March 2025, situational awareness has been measured more comprehensively. Situational awareness lies on a spectrum ranging from basic to advanced. A policy with high levels of situational awareness would possess and be able to apply knowledge like:- How humans will respond to its behaviour in a range of situations. This ties into the model's own perception , reasoning, memory and goal stability and brings us back to coherence in terms of model consistency and integration of cognition across goals , representations and necessary actions. In simple terms, a coherent AGI should *"think, reason, plan LeCun [4] and act"* as one mind/model instead of a collection of disjointed pattern generators. Humans have a solidified version of such coherence and thanks to evolutionary optimization, we have done pretty well to lack any sort of bottleneck.

As AI continued to evolve, a simple and clear definition of AGI is yet to be stated without any ambiguity. One of the primary reasons for the ambiguity is the constant shifting of the goal-post on what the exact criteria needs to be for a formal definition. As specialized AI systems master tasks once thought to require human intellect—from mathematics to art—the criteria for "AGI" continually shift. This ambiguity fuels unproductive debates, hinders discussions about how far AGI is, and ultimately obscures the gap between today's AI and AGI. The original definition from Openai was " an automated system capable of doing economically valuable work that a human being can do". Like humans , they can need some additional training for a job, that's fine. As of this writing in November of 2025, Recent works by Hendrycks et al. [5] takes a decisive step towards formalizing a definition for AGI. Building on the informal notion that "AGI is an AI that can match or exceed the cognitive versatility and proficiency of a well-educated adult," their framework grounds evaluation in human psychometrics through the **Cattell-Horn-Carroll (CHC)** theory

of cognitive abilities. [McGrew [6], Schneider and McGrew [7]] CHC is the most empirically validated model of human intelligence. It is primarily derived from the synthesis of over a century of iterative factor analysis of diverse collections of cognitive ability tests. By applying the CHC framework, the authors of the paper conclude that contemporary AI systems, even though they have managed to perform incredibly well on complex human benchmarks, lacks the core cognitive capabilities that are necessary for human intelligence. By testing the fundamental abilities that underpin human cognition—many of which appear simple for humans, it has been found that contemporary AI systems can solve roughly half of these often-simple assessment . While elegant in its formulation, This Fourati [8] paper assumes a degree of compensability.

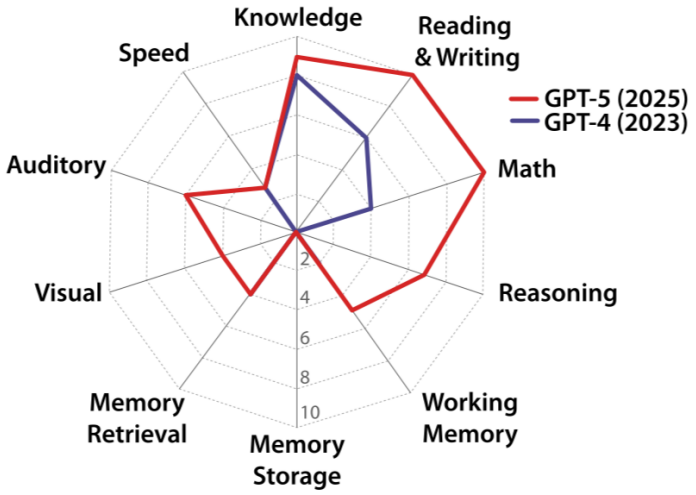

Figure 1: - Hendrycks et al. [5]. CHC capabilities of GPT-4 and GPT-5. GPT-5 's answers questions here in 'Auto' mode.

This means that exceptional performance in some faculties can offset severe deficiencies in others. As a result, an AI system could appear "general" while failing entirely in critical domains such as reasoning or memory. This assumption conflicts with psychometric evidence . Hence, there is a need for a coherent metric where we can quantify how consistently , a system maintains its reasoning and goal trajectories over time and context. In essence , it measures how much of a mind a single large foundational model really is. An AI that fails in long term memory or perception cannot be considered "general". Another distinction that needs emphasis here is between LLMs acting as simulators vs AGI as entities. A coherence metric should be able to tell us if an AGI keeps consistent world representations across contexts, avoids drifts in reasoning over long horizon tasks and maintains a stable goal pursuit and does not misgeneralize to a faulty outcome. Now, what would such a coherence metric formalism look like ?. We will be discussing more about it in the the epistemic layer.

The current discourse surrounding AGI can be understood as a conflict between two competing thought processes, Subasioglu and Subasioglu [9] What a system does (**functionalism** of a system) vs what a system is (**phenomenology**). The functionalism side of the argument focuses on the externalities of a model's ability and observable outputs. The Turing test , which is a foundational benchmark that judges

intelligence based on a machine's ability to generate responses that are indistinguishable from a that of a human's response is one such cognitive test that is required for an intelligent system to pass. A system is yet to pass the Turing test, A 2025 study Jones and Bergen [10] shows that GPT- 4.5 passed the test by convincing judges it was human 73 % of the time. But, because there's so much more that is still missing in the system, it doesn't really mean much whether a system has passed the turing test or not. A recent and highly influential framework by Google DeepMind proposes "Levels of AGI" based on performance and generality, from "Emerging" (unskilled human) to "Superhuman" (outperforming all humans). This approach, while providing a good initial metric for performance, fundamentally focuses on what a system can do rather than how it does it, assuming the mechanisms are secondary. It can be argued that these definitions lack a clear guidance on what AGI truly is as it lacks a clear architectural structure/quality that define what true intelligence is.

A lot of progress has been made since 2014 on mimicking human responses, but such definitions still lack the human bottleneck that has been so efficiently provided as well as optimized through evolutionary mechanisms. This is exactly where the phenomenological side of the argument produces the narrative that in order to build a general intelligence entity, it must fully replicate human- like cognitive processes. In theory this isn't a terrible idea , but in practice it almost always omits the kind of natural evolutionary mechanism that it took for humans to get to where we are. This perspective, often voiced by cognitive scientists, neuroscientists, and a subset of AI researchers, emphasizes the necessity of internal states like consciousness, sentience, and genuine understanding, distinguishing intelligence from what could be mere sophisticated mimicry. From this viewpoint, a system that passes the Turing Test or excels at economically valuable work might still be a **"philosophical zombie"**—a system that perfectly imitates intelligent behavior without any accompanying subjective experience. A classic example of this distinction is John Searle's **Chinese Room argument**, which challenges the idea that a machine can have a "mind" simply by following a program. A fundamental question that arises subsequently is the idea that if representations mimic human responses are already included in pre-training, wouldn't it be imperative for a model to pick up let's say - a langauge new language very easily ?. The answer is still a NO , as models still lack fluid intelligence , something that peaks in humans during adulthood. The lack of fluid intelligence in models makes it very brittle as we are going to see in the epistemic layer. If pre-training truly increased the ability to learn new tasks , then LLMs would perform much better on ARC tasks. **Abstraction and Reasoning Corpus (ARC)**, introduced by Francois Chollet, who was also the inventor of Keras , to specifically measure abstract reasoning and novel problem-solving abilities. ARC puzzles have really low complexity and each require very little knowledge to solve. Even children are also able to them very easily. But LLMs still struggle, even with 100,000 times more knowledge. The only thing that makes ARC special is that it was designed with this intent to resist memorization. This is the huge blocker for LLM performance. If one looks at LLMs closely, it's pretty obvious that they're not really synthesizing new programs on the fly to solve the task that they're faced with. Instead, they're reapplying things that they've stored in memory. Despite the exponential scaling between 2020 and 2024 , the state of the art scores hovered between 34 % , Chollet et al. [11] well below the estimated human baseline of 60-70%. While recent breakthroughs have pushed some systems' performance to surpass human scores , on these benchmarks, these results were often a consequence of exponential scaling rather than a fundamental leap in cognitive architecture. This struggle has been further underscored by the introduction of the new more challenging ARC-AGI-2 benchmark in 2025 , which is specifically designed to resist these methods. Chollet et al. [12] Early results show that even in frontier AI models that excelled on the original benchmark struggled with ARC-AGI-2 , scoring in the

low single digits , while humans can solve every task with a higher rate of efficiency. Although one may include the evidence that Gemini 1.5 was able to learn a language in context , including its grammer, it can be assumed that it has simply mined the required template from a vast corpus of training data and is just reusing it. Models still struggle with a poor ability to synthesize new program templates like this on the fly, or even adapt existing ones. They're very much limited to fetching data. By late December of 2025, while writing this piece, Google as well as OpenAI came out with their own models like GPT 5.2 and Gemini 3 that have crossed the 50 % mark ARC-AGI leaderboard on the ARC-AGI scale. But it is still well below the 70 % human baseline.

But if you think about it, evolution is not a mimic based mechanism, rather it is a hierarchy that is based on fundamental drives like motivation, reward, danger and curiosity. For example- **drive theory** posits that physiological needs create an aroused state that motivates an organism to satisfy them, while **Maslow's hierarchy of needs** organizes human motivation from basic physiological and safety needs to higher-order drives like social connection and self-actualization. For an AGI system, its foundational objective should be to ensure understanding and reasoning, efficiently allocate compute, generate the appropriate actions and keep updating/ evolving the system. Arising from these objectives , one might think about **intrinsic motivations** that a system must possess which drives to larn , explore and master new skills, something in correlation with how biological systems operate. According to Deci and Ryan's Self-Determination Theory, this motivation stems from the fulfillment of three basic psychological needs: autonomy (the feeling of volition), competence (the feeling of mastery), and relatedness (the feeling of connection to others). This internal drive should fuel curiosity and exploration , something that is key to the acquisition of knowledge for a general intelligence system to grow and develop. An AI system that is motivated intrinsically isn't just doing as it has been programmed, but it is actively seeking out novel experiences that helps to shape its internal world model. In the context of RL, this can be formally expressed by a total reward function that combines both intrinsic and extrinsic rewards:- [13]

$$R_{total}(s_t, a_t) = R_{ext}(s_t, a_t) + \beta R_{int}(s_t, a_t)$$

where s_t is the state of the agent at time t, a_t is the action by the agent and β is the coefficient that controls the weight of intrinsic motivation. We will discuss the reward function in depth in the epistemic as well as the human layer.

A common but limited approach to intrinsic reward is to formulate curiosity as the prediction error of the agent's world model.

$$R_{int} = ||f(s_t, a_t) - s_{t+1}||$$

where f is a learned forward model and s_{t+1} is the observed next state. Pathak et al. [14]

3 Motivations for the Technical Layer

The technical layer which we are going to discuss in this section, provides the substrate upon which both the epistemic and human layers rest. It establishes the learning dynamics and also talks about some constraints (particularly in scaling) that determine what an AGI can represent, how it generalizes and how it reliably

performs especially on long-horizon tasks. While the human layer governs interactions and values (alignment) , the technical layer describes the model's phenomenology (what it is) that includes it's biases, the limitations(bottlenecks) and internal structures that goes way beyond a simple pattern recognizer.

Long-Horizon tasks as we will see require consistent reasoning, planning and credit assignment over thousands and millions of steps. Techniques such as trajectory level credit assignment, hierarchial abstractions etc, all aim at stable long-term reasoning, but we still lack scalable solutions that match the efficiency of autoregressive pre-training. Development of better Temporal Abstraction should be a long term goal. To fare better at long horizon tasks , we need evals to check for discontinuous jumps in capabilities.

Therefore, we need to ground AGI in architectures capable of handling temporal decision making without catastrophic forgetting. The scaling hypothesis states that increasing model size, dataset size and compute leads to predictable improvements in loss. Scaling can also help push models into regimes where emergent abilities arise but deceptive behavior also becomes possible. The technical layer of a human-centered AGI must integrate scaling with architectures that help in the preservation of interpretability while also being able to tackle diminishing returns on validation loss.

An AGI that is human centered must adapt to evolving information and tasks without overwriting earlier learned knowledge. Continual learning is therefore an essential component in a model being able to acquire knowledge autonomously. Continual learning should provide the foundation for *stability, memory and in-context learning*. If we want it to be aligned to humans, we need to use cognitive priors as a design pre-requisite , if we wish to align the model towards human-legible reasoning and human compatible generalization. This doesn't explicitly mean copying our brain, but being able to leverage cognitive principles. The goal should be to ensure that the AGI's representational geometry is aligned with human interpretable abstractions. Disentanglement also allows an AGI to separate features more efficiently.

A highly entangled model acts like a black-box where internal features lack clear boundaries. Disentanglement ensures that the AGI reasons in modular components much like humans do. This is a technical pre-requisite to interpretable circuits, which we are going to discuss in more detail in the epistemic layer. The pre-training thesis argues that cognitive abilities of AGI emerge during large-scale, unsupervised pre-training. Pre-training builds those inherent tools that alignment methods later attempts to shape. This makes pre-training sort of an alignment bottleneck. Whatever biases, goals, abstractions arise during pre-training becomes the substrate that we later hope to control and interpret. This layer also discusses the problems that arise with scaling.

An AGI model should be able to dynamically allocate compute when thinking/reasoning. The role of Chain-Of -Thought (CoT) in a human-centered AGI should be that of adaptive compute , where a model is able to dynamically expand computation allocated to a given problem by generating intermediate steps. CoT also shapes the model's internal priors thus influencing future attention . Within the technical layer, RL scaling is not treated as a capstone but as a final narrow tool-layer on top of a robustly pre-trained world model. RL scaling very evidently provides an economic bottleneck in terms of how far we can scale up to reach AGI and thus scaling as a concept in itself can be a constraint that ultimately inhibits the engineering of a foundational AGI model that is attuned to human needs and requirements.

Let us discuss a few of the human effects that we have already begun to experience as LLMs become more mainstream.

3.1 The Brain on ChatGPT.

A recent Kosmyna et al. [15] paper conducted a study on three different groups of participants to measure the level of long-term mental cost of using external tools. Tools which can be used to avoid the mental effort of critical thinking and learning , also known as cognitive debt. One group was assigned to use LLMs(ChatGPT), the other was assigned to use search and the last one was assigned to use their Brains(no tools). EEG was used to record the participants' brain activity in order to assess their cognitive engagement and cognitive load, and to gain a deeper understanding of neural activations during the essay writing task.

A quote from the paper:-

> " EEG analysis suggested that the Search Engine and Brain-only groups had significantly different neural connectivity patterns. Brain connectivity systematically scaled down with the amount of external support".

Even though the authors of the study acknowledge the limitations such as the limited number of participants, which limits the generalizability of findings to broader populations or different educational contexts, the lack of other LLMs etc, this is still a significant step towards understanding the homogeneity across the Named Entities Recognition (NERs), n-grams, ontology of topics within each group. In essence, the paper highlights possible cognitive costs of over-relying on AI tools like ChatGPT for writing tasks, though further research is needed for confirmation and broader implications.

Another Yakura et al. [16] study which is essential to mention for our discussion is from Max-Planck institute where a cause and effect relationship has been studied from 740,249 hours of human discourse from 360,445 YouTube academic talks and 771,591 conversational podcast episodes across multiple disciplines. The researchers have detected a measurable and abrupt increase in words like "delve", "comprehend" , "boast", "swift", and "meticulous" , preferably generated by ChatGPT. This raises the question about the Homogenization of language where a closed cultural feedback loop in which the cultural traits circulate between humans and , thus raising concerns about the erosion of both cultural as well as linguistic diversity.

3.2 On productivity

A recent paper suggests in terms of productivity , in this case programming using LLMs, it can make us feel like we are 20 percent more productive , all the while making us 19 percent slower. We have the illusion of productivity while using LLMs. Questions still remain when it comes to cognitive offloading as well as the code quality being produced. When we use LLMs to gain a significant productivity boost, there exists a much steeper learning curve. This Becker et al. [17] study had 16 participants, with a mix of previous exposure to AI tools - 56% of them had never used Cursor before, and the study was mainly about Cursor.

They then had those 16 participants work on issues (about 15 each), where each issue was randomly assigned a "you can use AI" v.s. "you can't use AI" rule.

So each developer worked on a mix of AI-tasks and no-AI-tasks during the study.A quarter of the participants saw increased performance, 3/4ths saw reduced performance. One of the top performers for AI was also someone with the most previous Cursor experience. The paper acknowledges that here:

> "However, we see positive speedup for the one developer who has more than 50 hours of

Cursor experience, so it's plausible that there is a high skill ceiling for using Cursor, such that developers with significant experience see positive speedup."

In the human context, what should we take into account ?

3.3 Long-Horizon RL

In case of Long Horizon tasks, humans still fare better than LLMs. If we manage not to offload our cognitive abilities to such external tools , we might be able to outlast the first wave of superintelligence but in the long run , we may lose sight of what needs to be done. So, what can be done right now ?. A bit of a theoretical background on Long Horizon tasks first.

Long-horizon learning refers to the challenge of making predictions, decisions, or credit assignments over extended temporal or causal spans — situations where the effects of an action, observation, or update may not be visible until far in the future. In long-horizon learning, the agent or model must learn from delayed feedback over many steps or episodes, often with sparse or noisy rewards.

In reinforcement learning, the purpose or goal of the agent is formalized in terms of a special signal, called the reward, passing from the environment to the agent.

At each time step, the reward is a simple number. An agent's goal is to maximize the total amount of reward it receives. This means maximizing not immediate reward, but cumulative reward in the long run. This is what is known as informal learning. Now, if the agent's goal is to maximize the rewards, we have to define this formally. The sequence of rewards received after time-step t can be written as:- R_{t+1}, R_{t+2}.... If we wish to maximize the rewards, we can take the sum total of the rewards:- $G_t = R_{t+1} + R_{t+2}$ where T is the final time-step. [13]. Now, in order to determine the present value of the reward after k time-steps , we have to introduce another concept. This is called *discounting*. According to this approach, the agent tries to select actions so that the sum of the discounted rewards it receives over the future is maximized. To maximize the expected discounted return, we can write

$$G_t = R_{t+1} + \gamma R_{t+2} + \gamma^2_{t+3} = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1}$$

where γ is the parametre $0 \leq \gamma \leq 1$, called the discount rate. When T is large, the *horizon* is long , which means that the return depends on distant future events . Small modeling or credit assignment errors compound over time. There are questions that arise when it comes to the problems that are introduced by Long-Horizon learning. There is the issue with Exploration:- How do we explore effectively when the rewards are delayed or denied ?. Small predictive or value errors can compound over time, thus creating a snowball effect. There is also an issue with model bias because for model-based RL, even slight inaccuracies in the learned dynamics cause divergence over long rollouts.

Now, what are some of the challenges that Human beings need to overcome ?

The good news is that Human beings are generally good at Long-Horizon learning than model-based RLs , because we usually don't compound small errors over time resulting in large divergences. Unlike some agents, humans can routinely plan decades ahead and also adapt when distant outcomes fail. There are certain fundamental traits that we have adapted to like for instance:- Delayed and distant rewards(Studying

for years before receiving a pay-off, without being thrown off our goal). Being able to establish causal links between actions and results(these can be stochastic / indirect). Also our search space grows exponentially over time.

Some key mechanisms that humans use are as follows:-

- **Temporal abstraction**:- Humans don't learn at an indiviual level of action, but rather we apply a hierarchial approach to understanding levels of abstraction. Zhang et al. [18] This is akin to Hierarchial Reinforcement learning , a method where complex tasks can be broken into smaller , more manageable subtasks organized in a hierarchial fashion. We are also pretty good at being task agnostic - i.e, solutions for a single task, can be generalized to be widely applied to different tasks. A good example is :- learning how to drive. Once basic control is automatized, the "driving" macro-action can be invoked as a unit; we no longer assign credit to individual steering corrections. Our brain can naturally implement "options" that operate at different temporal resolutions.
- **Model-based cognition**:- We can construct mental models of how the world works. Reasoning on a macro level , narrative simulation — allowing us to predict distant outcomes without having to experience them first. This distinguishes us humans from some RL agents where credit assignment-predicting which early actions cause future outcomes , compound over time. We can perform "what if" actions in our heads to overcome that. For example:- Instead of waiting to see if touching fire hurts us , we can create a mental simulation to avoid it next time. In this way, humans can offset the sample inefficiency problem that occurs with RLs, in discrete and continuous tasks.
- **Episodic memory and compression**:- Rather than storing long sequences of information or trajectories, we as humans can store certain "key" moments and then make decisions on that basis. This is known as recalling **episodes** where causal links can be established between action and outcomes . For eg:- saying *Because I was procrastinating, I missed the deadline*. This very effectively reduces the horizon and also reduces credit assingment ambiguity. We can also recall episodes that are compressed causal structure instead of re-calling every time-step.
- **Temporal regulation**:- There are a few ways using which we can reward the search space when there is a "sparse reward" problem. Often, what is available as a sparse reward is only an indication of whether or not a task is completed fully. Most RL agents, fail to learn an acceptable policy within a certain time frame. One of the reasons being - large amounts of exploration that a policy has to perform before it gets a reasonable feedback, which it can learn from. Because of *curiosity* - humans can intuitively develop an intrinsic reward for exploration. We can also tap into inherent reward/penalty systems like *guilt* or *regret* , that can solve for the credit assingment issue.

Evolutionary mechanisms allows us to be better at **zero-shot learning** as well. From prior knowledge, we can effectively develop policy that learns over the exploration space very easily. Such policy can be applied to previously unseen tasks and perform well on it.

This is the RL agent vs human distinction.

Evolutionarily, we as humans are going to do better at certain tasks than the agents. By utilizing some of those key mechanisms we can develop RL agents that are more aligned with our needs all the while maintaining our own cognitive gains developed over time. Developing better **Temporal Abstraction**

architecture should be the long-term goal , something that can perform better on tasks related to complex environments involving sparse rewards, diverse behaviors and also long term planning.

John Schulman, who is the Senior researcher at Anthropic makes the claim that AI agents haven't really taken off because of their low-reliability rather than long-horizon task performance. Intuitively, one may say that this is exactly the difficulty with long-horizon tasks . We have to do multiple steps sequentially, in a row , thus diminishing the overall task reliability. This Sinha et al. [19] paper makes the argument that the failures of LLMs occur when simple tasks are made longer. These arise because of mistakes in execution rather than an inability to reason. The paper proposes *execution capability* by explicitly providing the knowledge and plan needed to solve a long-horizon task.

One solution which we can implement here is to train on rewards, thus enabling more sophisticated reasoning. This opens up the opportunity to solve much longer tasks where earlier human supervision would be too expensive to scale. We know that scaling laws for language models show diminishing returns on the loss for the single step of predicting the next token. When the models competed in simple knowledge based question-answering tasks, such as on the MMLU benchmark, these single-step measurements could inform us about the rate of progress. In contrast to post-training on human demonstrations , language models can now be trained on just rewards as we have mentioned. A question that follows is - what are the effects of scaling laws on long-horizon tasks ?

Longer task horizons primarily affect the coefficients of scaling laws, and lead to a higher "scaling multiplier"(a factor that dictates the number of samples needed to train the model) [20] that increases the sample complexity required for training. This means longer tasks require more data to achieve the same performance gains as shorter tasks, and scaling strategies like increasing model size have different effects, with very large models leading to performance plateauing in repetitive, long-horizon tasks. Therefore, understanding how horizon length modifies scaling is crucial for resource allocation, especially in RL domains.

This Hilton et al. [21] paper by Hilton and Schulman proposes *intrinsic performance* which is defined to be equal to training compute on the compute-efficient frontier of the tradeoff between model size and environment interactions. This causes the relationship between performance and training compute to follow a power law by definition, thereby making it possible to study the remaining relationships between performance, model size and environment interactions. These relationships have been studied across a range of environments like a 1v1 version of Dota 2 , a toy MNIST environment , Procgen. One implication for their optimal model vs compute scaling law is that once the horizon h [2]

becomes larger, increasing the horizon leads to a proportional increase in the compute budget corresponding to each given optimal model size, without changing the scaling exponent.

Quote from the paper.

> "The number of environment interactions required to reach a given performance level will eventually scale approximately proportionally to h"

[2]GAE:- Generalized Advantage Estimation :- which is used to enhance the performance and stability of policy gradient methods Schulman et al. [22]. From GAE , discount rate:- $\gamma = 1 - \frac{2}{h+1}$

A question that arises now subsequently is - how do we manage to keep track of long-horizon RL tasks ?. Dwarkesh patel asks this exact question to John Schulman on his podcast. There are many solutions which can be applied, one of them being measuring the agents with interactive evaluations. This method involves step-wise analysis which tracks progress at each stage. Interactive evaluations can also involve RLHF as a judge. As John Schulman answers that the potential of something adversarial happening now is really low, considering how humans still fare better at long horizon tasks than LLMs. It is also essential that we need evals to check for discontinuous jumps in capabilities. We would also need to factor in resource usage , such as tokens or compute time which is important for long running tasks. The current benchmarks which we have today, predominantly focus on atomized tasks. This approach fails to capture the long term contextual dependencies required in realistic scenarios. **Odysseybench** is a recent Wang et al. [23] paper that provides a comprehensive benchmark for evaluating LLM agents on long-horizon workflows across diverse office applications like Word, Excel, Pdf and calender.

This Kwa et al. [24] paper also measures AI performance in terms of the length of tasks that AI agents can complete. The paper shows that this performance metric has been exponentially increasing in the past six years with a doubling time of about 7 months. The authors predict that upon extrapolating this trend, in under a decade, we will see AI agents that can independently complete a large fraction of the software tasks that takes humans days or weeks. Through the lens of a human expert, the authors conclude that models have a 100 percent success rate on tasks taking humans less than 4 minutes, but succeed less than 10 percent of the time that take more than 4 hours to solve. We can fit a curve to predict model success probability using human task length . If we plot it on a logarithmic scale, we can see that the length of tasks that models can complete is well predicted by an exponential trend, with a doubling time of 7 months.

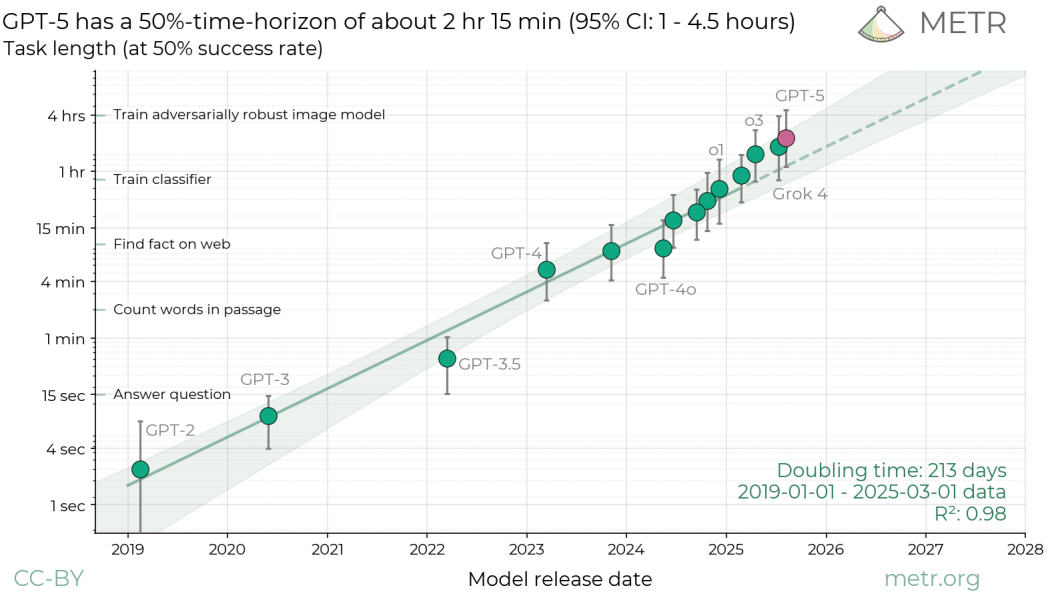

Figure 2:- The length of tasks (measured by how long they take human professionals) that generalist frontier model agents can complete autonomously with 50% reliability has been doubling approximately every 7 months for the last 6 years. The shaded region represents 95 percent CI calculated by hierarchical bootstrap over task families, tasks, and task attempts. blog

Recently, claude Opus 4.5 achieved a METR time horizon of more than four hours. This is almost a vertical development from GPT- 5.2. An important thing to remember here is that the work it does is directly on a 1v1 human task. It was previously quite limited in only doing many short time horizon things under human direction or with human in the loop needing to give it a ton of focus.

We can also try and scale up models to see whether or not they can perform better at longer tasks. A significant amount of work has been done in the last few years to figure out just that.

3.4 Does Scaling really work ?

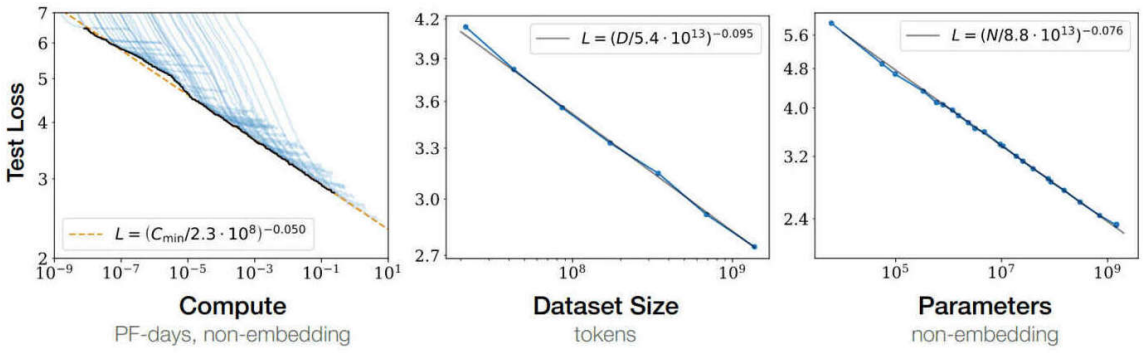

Figure 3:- DL scaling laws, compute, data, model parameters

The scaling papers suggest that the leaps that we have seen over the last few years aren't really half way there in terms of absolute loss- likelihood. GPT-3 represnts $\sim 10^3$ on the above chart. The blessings of scaling is the observation that for deep learning , hard problems are easier to solve than easy problems. Everything gets better as it gets larger. This is in contrast to the actual outcome in research where the small/marginal things are hard, but the large things are nearly impossible. The bigger the neural net/compute/data/problem, the faster and better it learns. There's an inherent blessing in dimensionality within Deep Learning that makes it unreasonably effective. Simply by training a model on a large amount of data, induces properties like meta-learning without having any implicit architecture built-in. Recent developments in terms of scaling shows us that scaling up models without any kind of supervision can produce results that are competitive with the best and the most complex alternatives using the same simple architecture.

DeepMind researcher Matthew Botvinick [25], discussed their meta-reinforcement learning work where they were surprised to discover meta-learning emerging, and that it did so regardless of which specific architecture was used. A quote from him

> " . . . it's something that just happens. In a sense, you can't avoid this happening. If you have a system that has memory, and the function of that memory is shaped by reinforcement learning, and this system is trained on a series of interrelated tasks, this is going to happen. You can't stop it."

The answer to why they transfer and generalize and why scaling works well lies in a mix of compression as indicated by Gwern in his [26] blog. This leads us to the scaling hypothesis.

3.4.1 The Scaling Hypothesis

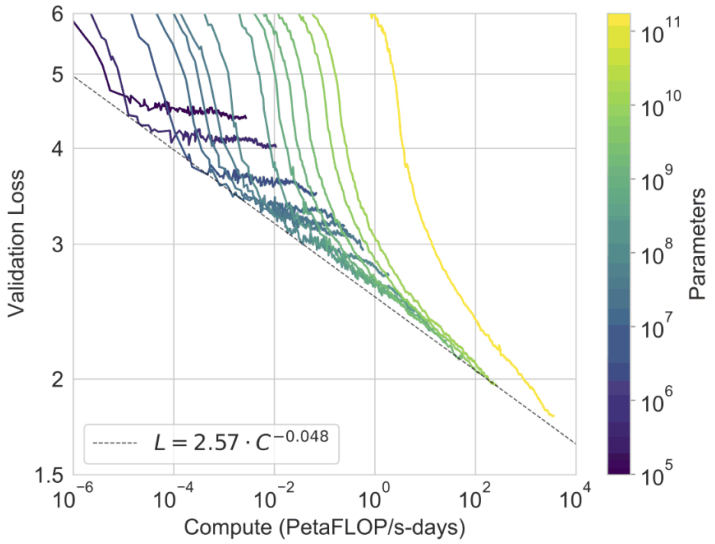

Figure 4: - Brown et al. [27] Smooth scaling of performance with compute. Performance (measured in terms of cross-entropy validation loss) follows a power-law trend with the amount of compute used for training. The power-law behavior observed in Kaplan et al. [28] continues for an additional two orders of magnitude with only small deviations from the predicted curve. For this figure, we exclude embedding parameters from compute and parameter counts. Brown et al. [27] Cross-validation loss extrapolation: $L = 2.57.C^{-0.048}$

The Scaling hypothesis is the claim that increasing the scale of neural networks in terms of model, size, data and compute - will predictably and smoothly improve performance, access nearly all cognitive tasks without requiring any fundamental algorithmic breakthroughs. In simpler terms, if you make the model bigger and train it on more diverse data, it keeps getting better, often in ways in which we didn't expect or explicitly train it to. The hypothesis originates from empirical findings from DeepMind, OpenAI and others between 2018-2020. OpenAI then extrapolated this to a strategic doctrine : *We don't need new architectures- just scale the current ones enough*. This became the scaling hypothesis in practice. The empirical core of the evidence came from the "Kaplan Scaling laws"- Kaplan et al. [28]

$$L(N, D, C) = L_\infty + A_N N^{-\alpha_N} + A_D D^{-\alpha_D} + A_C C^{-\alpha_C}$$

where α is the validation loss. N, D, C are parameters, datasize and compute. α_N, α_D, α_C are the scaling exponents. The exponents show us that improvements follow a slow power-law which isn't exponential. From the efficient frontier curve, we can determine the value of α which is ~ 0.046 .Kaplan et al. [28] Therefore, the scaling formula is:- $L = 2.57C^{-0.048}$ [26]

We would require a huge increase in model size to gain small improvements on validation loss. At a sufficient scale, new emergent behaviors can emerge that are consistent with crossing internal representation thresholds - something that can be attributed to scaling and not internal architectural innovation.

3.4.2 Validation loss, Perplexity Score

Now, just to give the reader some context in validation loss, the GPT validation loss refers to the average prediction error (usually measured by cross-entropy) computed on a validation set which is a crucial metric used to monitor and evaluate a GPT model. During training, the GPT model learns to predict the next token x_t, given that the previous token is $x_{<t}$. The model then defines a conditional probability distribution $P_\theta(x_t|x_{<t})$. [29] where θ represents the model parameters (weights). The validation loss is the average negative log-likelihood (or equivalently, cross-entropy) of the validation tokens of the current model.

$$\mathcal{L}_{val(\theta)} = -\frac{1}{N}\sum_{i=1}^{N} logP_\theta(x_t|x_{<t})$$

Where N, is the total number of tokens in the validation set.

$P_\theta(x_t|x_{<t})$ is the model's predicted probability for the true token. This represents the expected surprise of the model on unseen texts. The training laws as we already know, measures how well the model fits the training data. The validation loss measures how well the model generalizes to unseen data. A rising validation loss while training loss continues to decrease indicates overfitting. The validation loss (cross-entropy) is directly related to **perplexity** , which is a more interpretable metric that computes the exponentiated average negative-log likelihood

$$Perplexity = exp[-\frac{1}{N}\sum_{i=1}^{N} logP_\theta(x_t|x_{<t})]$$

Intuitively, it can be thought of as an evaluation of the model's ability to predict uniformly among the set of specified tokens in a corpus. Importantly, this means that the tokenization procedure has a direct impact on a model's perplexity which should always be taken into consideration when comparing different models. A lower perplexity score indicates better predictive performance . Suppose the validation loss is 1.2, in that case , the perplexity score would be:- $e^{1.2} \simeq 3.2$. The typical validation loss/ perplexity ranges for the following models are:-

Model	Validation Loss	Perplexity
GPT-2 model small (117M) parameters	~ 2.9	~ 18
GPT-2 medium (345M)	~ 2.6	~ 13.5
GPT-2 XL (1.5B)	~ 2.4	~ 11
GPT-3 (175B)	$\sim 1.5 - 2.0$	~ 6.7
GPT- 4 class models	estimated < 1.5	~ 4.5 or lower

Now, let's try to compute C which is the compute value for one such model. say GPT-3. The compute value C is given by $C = 6ND$ Hoffmann et al. [30] [Chinchilla paper] where N is the number of parameters and

D is the number of tokens. Now, GPT-3 has 175B parametres and almost 3.7 trillion tokens. Therefore C would be:-

$C = 1.75 * 10^{11} * 3.7 * 10^{12}$ which is equal to $3.885 * 10^{24}$ FLOPS.

If we convert it to PetaFLOPs per day compute . There's 86,400 seconds in a day , therefore PetaFLOPs per day will be equal to $10^{15} * 86,400$, so $8.64 * 10^{19}$ operations in one day . Therefore, PetaFLOPs per day = $\frac{C}{8.64*10^{19}}$

Calculating PFLOPs per day = 44,965 PF-days.

Now, applying the scaling formula :- $L = 2.57C^{-0.048} =\sim 1.54$ approx.

According to the table given in Hoffmann et al. [30] [Chinchilla] paper for the **estimated training FLOPs and training tokens for various model sizes**, we can compute the loss, as model size increases. [For the sake of simplicity, I have converted the FLOPs into PFLOPs per day.

Model parameters	Tokens	FLOPs(PFLOPs /day)	Loss ($2.57C^{-0.048}$)
400M	8 billion	0.22	2.76
1B	20.2 billion	1.44	2.52
10B	205.1 billion	142.43	2.02
67B	1.5 Trillion	6979.1	1.68
175B	3.7 Trillion	44,965	1.54
280B	5.9 Trillion	114722.22	1.46
580B	11.0 Trillion	443,055.55	1.37
1T	21.2 Trillion	1,472,222.22	1.30
10T	215.2 Trillion	150M	1.04

20:1 ratio of training tokens to model parameters

From the above table, we can conclude that as model size and number of tokens are scaled up, we get a decrease in the validation loss. GPT-2 XL (1.5B) had a validation loss of ~ 3.32. By contrast, GPT-3 (175B) has halved it, $\sim 1.5 - 2.0$. If the above table is any indication, the scaling curve will continue for many more orders of magnitude and the loss will subsequently drop less than 1 .

If GPT-3 has gained this much meta-learning and world knowledge by dropping the loss by 50% compared to GPT-2, we can only speculate what capabilities above GPT-5x will gain when parameter count exceeds 10 Trillion. The question is - what would a drop of ≤ 1 gain ?. From the scaling hypothesis, the philosophical interpretation can imply that intelligence might be a quantitative leap instead of a qualitative one. That is, general purpose reasoning , langauge and abstraction may naturally emerge from large enough predictive

models of the world, given sufficient data and optimization rather than requiring explicitly "symbolic" or "human-like" algorithm. From the scaling hypothesis, we can say that intelligence is an emergent property or simply a consequence of scaling. But, there are limits to scaling and it might not always be beneficial as we will see in the subsequent sections. So, what does this mean for long-horizon tasks, which is our original discussion. We need to understand how scaling laws interact with long-horizon tasks.

This Xiong et al. [31] paper performs extensive evaluation on language models. On research benchmarks, the models achieve consistent improvement on tasks and significant improvements on long-context tasks. The model supports context windows of upto 32k tokens. The authors show that the model continues to gain performance despite having diminishing returns . The 70B model for instance, the loss can be reduced by a factor of $2^{\beta} \simeq 0.7$ when the context length is doubled plus a model specific constant $(1 - 2^{\beta}).\gamma$ where β is the data coefficient.

Conceptually, the empirical values of the scaling coefficient and the data coefficient give **slower power law gains** with scale as we have seen previously. In many cases like in this Bordelon et al. [32] paper, we can see how performance scales with context length and model width. The horizon exponent (γ) tends to be positive and model/architecture dependent . For auto-regressive transformers, γ must be substantial because the errors compound across steps. We can mitigate the cost of architectures with **explicit memory retrieval** because γ is smaller.

3.5 Continual Learning

With all of the benchmarks, that are being crossed by AI, it definitely makes one think that AGI is imminent. With GPT-4 it was about the SATs, the AP tests, the LSATs etc and soon after that, it was solving medium to hard level problems in programming. Especially in LeetCode Vishnu et al. [33] Now it is solving PhD level qustions in Math and Physics [34] There are a few problems with scaling. The current trends include measuring progress on benchmarks and how that progress influences the direction of research. So far, it is difficult to indicate a popular benchmark for LLMs that actually measure intelligence. There are certain benchmarks that aim to measure LLM intelligence on complex tasks, for eg:- MMLU, ARC-AGI, GPQA etc, but it is still difficult to point to a particular benchmark that collectively provide a more comprehensive assessment of a model's abilities. This compels us to think about an actual measure of intelligence.

If you think about it, intelligence isn't about how many tasks we can do, the ability to acquire knowledge and applying it should also be a part of the definition. Almost every benchmark out there tests the ability of the model to apply knowledge , but almost none of them focus on the ability to acquire it. Current frontier models almost entirely lack the ability to continually learn [35] We can train them for once, but after that their ability to acquire new knowledge is severely limited and that presents a problem. A counter argument is **in-context learning**. This a technique where LLMs learn to perform new tasks by receiving examples and instructions within a single prompt, without any changes to the model's underlying parameters. This allows the model to adapt to specific contexts and tasks at the time of the query, using methods like zero-shot (no examples), one-shot (one example), and few-shot (a few examples) prompting. Basically, the model learns new information on the fly. The finetuning argument also has merits to it. An AI agent should be able to learn new information autonomously, without any need for human intervention.

One of the drawbacks of not being able to continuously learn is the reliance of millions of examples [I will talk about this in-depth in subsequent sections], which means that it an perform well in tasks where millions

of examples exist. This is in contrast with how human beings learn. This is weird predicament that we are currently in, where AI can solve advanced programming problems, become top scorers in difficult exams , discover new drugs and even decode conversations between whales [36] yes this is a real thing !, it is still struggling to learn simple concepts . Atleast not without thousands if not millions of examples and an incredibly expensive training process.

The current state of AI revolves along this notion, that a single large foundational model has seen so much data that it can generalize to almost anything . This brings us to the problem of data saturation. Continual RL actually goes back to the 1930s with Thomas Ross's Thinking Machine [37] and Steven Smith's Robotic rat that could learn how to navigate a maze. But, continual learning alone isn't going to be good enough, it also needs to learn within context. There are ways for an agent to continually learn after it's initial training phase, but there are certain caveats to it.

Let' s start with Finetuning. If we finetune the same model several times, the model might run into several problems, like forgetting what it had known previously. This is known as **Catastrophic Forgetting**, where a model abruptly forgets learned information and loses it's ability to learn over time. This is quite inefficient. We can implement some solutions like regularization, architectural changes etc. We can also implement **Transfer Learning**, where we can freeze the weights of the top layers and the rest of the layers canbe repurposed. But, the disadvantage here is that - keeping the model's top layers frozen , limits the types of parameters that are learnable within in-context learning. And anything that a model does learn in-context is only learned temporarily until that information exists the context window.

But since, LLMs are mostly few-shot learners Brown et al. [27], it is quite imminent that foundational models will become outdated as the real world data that they are trained on changes. Bell et al. [38] . Model staleness will start to settle in and often times the only way to mitigate this is retraining. Recent estimates indicate that the cost of training models of a scale comparable to GPT-4.5 or similar architectures likely reached tens of millions of dollars due to the enormous compute and energy resources required. Hoffmann et al. [30] [Chinchilla]. There are research directions that we can take here, for instance, we can incrementally update the knowledge of foundational models through exposure to new data after their initial pre-training phase. This is known as Continual Pre-Training (CPT) [39]

This iterative updating allows foundational models to maintain their foundational abilities established during the initial training while simultaneously adapting to assimilate emerging information, thereby extending their operational lifespan and enhancing their adaptability to the ever-changing landscape of data and knowledge Yang et al. [40]. CPT is still in the early stages of development and bridging the gap between research and production remains challenging . While CPT techniques show promise in controlled experiments, their long-term stability and effectiveness over months of deployment in real-world settings remain under-explored.

While the focus here is incrementally updating the knowledge, catastrophic forgetting as mentioned above remains a key challenge. Another challenge that contributes to the decline in the model's performance is **Context-Rot**. We are iteratively updating the knowledge in CPT, but as a consequence the context window grows larger , leading to inaccurate results despite having more knowledge.

3.6 Context Rot

Models as we know struggle when the input length increases. They can also struggle when the context window grows larger. This happens even on tasks that models can handle appropriately at shorter lengths. This Chroma Research [41] paper demonstrates that models struggle with reasoning over long conversations. Considering a simple use case where we are building a chat assistant with memory, it has to remember previous essential information given by the user. Models perform better on condensed version where only relevant information is supplied instead of plugging the entire chat history. This should not be the case if the model is uniform along it's entire input length. Even the most advanced models struggle to find the right information when too much noise Shahani et al. [42] is present. The model's performance breaks down as ambiguity increases. The paper also shows that models struggle with **distractors** as input length increases. A distractor is topically related to the correct answer but doesn't really answer the question. Distractors are very common in real scenarios. Additionally, long context tasks often involve disambiguating amongst distractors as part of the task. One example is Multi-round co-reference resolution (MRCR) [Vodrahalli et al. [43], dataset], which involves retrieving the i-th instance of a specific user ask, amongst similar user asks, in a multi-turn conversation. However, there remains a lack of investigation into the impact of distractors in long context settings.

So, we have to engineer context to get reliable performance. Technically, one can use upto a million tokens, but in practice, the optimal context window is much smaller. This becomes an optimization problem. We want to maximize the amount of relevant information and minimize irrelevant context. We call this **Context Engineering** [44]. Of course, there isn't any single way to so this, it all depends on the use case. If one is working with a multi-step agent, summarization might be a useful strategy. Instead of chaining together long-action histories, we can insert summarization steps letting the model distill previous thoughts, into shorter and more relevant memory. Another option is **retrieval**. If one is working with a recurrent set of knowledge , we can store this in a vector database and retrieve only what is relevant at each step. It is cost-efficient, but we need to invest more time into retrieval strategy.

The recent **DeepSeek-OCR: Contexts Optical Compression** Wei et al. [45] paper introduces an approach for improving the efficiency of LLMs and Vision models by compressing extremely long contexts using a 2D visual representation. DeepSeek-OCR is a system that tackles the issue of long contexts by treating them not as a one-dimensional sequence of tokens, but as a compressed, 2D image. This process, called "Contexts Optical Compression," involves mapping the tokens onto a visual canvas to create a visual representation that is much more compact. A key finding from the paper is that when the number of original text tokens is upto 10 times the number of original vision tokens, (a 10x compression ratio), the model can still achieve a high decoding (OCR) precision of 97%. Even at an extreme compression ratio of 20x, the OCR (document parsing task in this context) accuracy remains substantial at about 60%

Given all the issues that occur with long horizon tasks and specific limitations that occur with LLMs, one aspect of the design process that we shouldn't be ignoring and it is actually the primary engineering discussion of this paper is human compatibility. The *human-centered* nature of AGI, should be such that it is compatible with the human purpose and objectives. One topic that has gotten some recognition over the last few years is the discussion surrounding cognition and consciousness, something both AI researchers and neuroscientists have been pondering upon.

There are a few motivations that lead engineers to tackling cognition. To be a little bit more specific,

implementing cognition as a design prior. The motivations being:-

- Current scaling techniques emphasize capability over control. Any kind of misalignment from the value objective can lead to catastrophic results.
- Models should align to us and not the other way around. Or atleast it should be bi-directional.
- There is an insufficient integration of human cognitive priors in within the engineering design of these models.

3.7 Human Cognition as a Design Prior

When we say human cognition as a design prior, we mean that the inductive biases, learning biases and representational structures of general intelligence systems should be informed by how human beings actually think , reason and generalize. It isn't about loss minimization on token prediction. In other words, cognition shouldn't just be about imitation (e.g training on human data), but also be a source of structural bias (any constraint that shapes how a model organizes information internally) on how the model *thinks, reasons, abstracts and grounds information*.

The theoretical foundation is rooted in cognitive abilities in human beings. Human beings , objectively speaking are quite efficient at generalizations. Humans can re-use concepts across different contexts, we can understand the "why" instead of just the "what". Causal and temporal structures are much better understood by humans , as mentioned previously. We are also pretty efficient at few-shot learning , where we can learn general rules from sparse examples as well as in zero-shot learning. Zero-shot is where we use only semantic information as examples. This is actually an essential component of our evolutionary mechanism . Sometimes, we have to use a set of assumptions or preferences in order to make predictions on data , that the model has not encountered during training. This is what is known as an **inductive bias**.

Without these assumptions, a model would be unable to generalize from the limited training data to unseen situations, as an infinite number of positive hypothesis (model complexity, overfitting etc) could not fit the observed data perfectly. In such a situation, cognition can be used as an inductive bias , as models need priors to generalize. In deep learning, these occur from the representations , the training objective that is being used (self-supervised) as well as from regularization (hyper-parameter regularization, such as dropout).

Human cognition as a prior means building priors that mirrors human abstraction mechanisms. Some of our cognitive features that are necessary to encode are:- Hierarchial abstractions , which supports reasoning at multiple levels, Recursive reasoning modules, [46] (**Theory of Mind**), multi-modal world models and definitely causal learning.

Alignment matters through shared priors . If the foundational model shares cognitive structures with humans, their internal reasoning steps become interpretable. Interpretability improves because the model's internal abstractions maps onto human understable categories. There is also a need to prevent goal misgeneralization , something that we will discuss in-depth in the epistemic layer. All you need to know for now is that, an incorrect goal in an Out-of- distribution (OOD) environment can lead to unexpected and potentially harmful outcomes, as opposed to simply failing due to a lack of capability. Such behaviour can lead to compounding effects that diverges sharply from the intended goal. This is known as **runaway misgeneralization**. Such a runaway model increases the risk of a deceptive outcome. Co-evolution of

understanding is equally important. Humans can teach, correct as well as audit systems . If both the models can share a similar representation and reasoning structure, alignment can be bi-directional instead of adversarial. This Shen et al. [47] paper from NeurIPS 2025 goes into it in more detail. You can check out the ICLR 2025 presentation on the same.

A quote from the paper.

> "Bidirectional Human-AI Alignment is a comprehensive framework that encompasses two interconnected alignment processes: 'Aligning AI with Humans' and 'Aligning Humans with AI'. The former focuses on integrating human specifications into training, steering, and customizing AI. The latter supports human agency, empowering people to think critically when using AI, collaborate effectively with it, and adapt societal approaches to maximize its benefits for humanity"

If we can make a model learn by isolating the underlying features from the observed data in representation form, it can manage to imitate the meaningful understanding process of humans when observing an object or relation. This process is known as **Disentanglement**.

3.7.1 Disentanglement

If we can manage to disentangle abstract features from one another , it can open up the possibility for learning representations of high level concepts. Something like this was mentioned in Yoshua Bengio's 2017 [48] paper. The main hypothesis from the paper was that DL succeeded in part because of inductive biases (priors), but additional ones also need to be included to go from good in-distribution generalizations (object recognition in images) [49] to strong OOD generalizations. A joint distribution between high level concepts (inductive priors as well as a consciousness prior) can help us understand/learn representations. A useful prior can thus be formed for representation learning. Wang et al. [50].

This Sucholutsky et al. [51] paper proposes a general formalism for representation alignment.

3.7.2 The Pre-Training Thesis

As mentioned, disentangled features definitely encourage representation learning. Priors can also be embedded in pre-training, so that the models can not only scale , but also think more like humans. Now, as humans we constantly emit large amounts of structured data which rely on *logic, causality, history* etc. [26]. All of that is reflected and encoded into our writings. A model that is learning to predict, must learn to understand all of that to get the best performance; as it predicts the easy things which are mere statistical pattern matching, what is left are the hard things. AI critics often say that the long tail of scenarios for tasks like self-driving or natural language can only be solved by true generalization or reasoning. It should be the case for models that in order to solve the long-tail they must learn how to generalize and reason.

Early on in training, the models learn some crude levels, that some letters like 'e' is much more frequent than others like 'z'. As training goes on, the task becomes more difficult. Now, it begins to learn what words actually exist and do not exist. It is so far, unaware of any meaning , but atleast now when it is asked to predict the second half of a word, it can actually do that to a certain degree. Next , the model picks up association among words. The loss per-bit of character continues to decrease. Grammatical consistency increases as training continues.

The pre-training thesis argues that it can go even further; we can compare such performance directly with humans, who are doing the same objective task. For a language model , the truth is that which keeps on predicting well—because truth is one and error many. The cognitive breakthroughs that allows ever so slightly better predictions of a few relevant texts. Ideal prediction needs to be nothing less than the truth.

Hypothetically, if we trained a model which had a loss of < 0.7, this could predict text which can be indistinguishable from that of a human being. The implication here being that the last few bits are the most valued and important bits . Shannon [52][Shannon(1951)] had estimated that the entropy of the English language , $0.6 - 1.3$ bits per character by asking human subjects to guess upcoming characters. There are outliers also at many steps. When it comes to trick questions, how will it be possible for a model to solve for these , without understanding- just guessing ?.

One can throw enormous amounts of compute at it and also sprinkle on some information theory in order to encode all tasks as a sequence prediction problem, and voila ..! we have achieved intelligence on some human benchmark. But is that enough though ? . According to the pretraining thesis, if a model achieved a low enough loss, sure. Theoretically, it would have to be intelligent, but it is still difficult to prove it in practice. Training character RNNs were fun, but it didn't revolutionize deep learning the way Transformers did. It provided the much needed context, but still suffered from vanishing gradient issues when sequences were too long. It definitely provided the stepping stone for Transformers to exist. Attention mechanism was the true revolution.Vaswani et al. [53] At a granular level, the models still have a lot of work to do. Countless petabytes of data will be required to include all sorts of outliers, subtleties and only then can we expect logical reasoning to include enough training signals, admist all of the noise and distractions to train a model. Or maybe the model is just too small to absorb anything beyond surface level signals. We would require a model be scaled 100x to get it to work, because the scaling curves did not co-operate. Finetuning can only help the problem for so long. Or simply put, maybe , the models are too fundamentally broken and for hierarchial abstractions to work, we need different architectures entirely.

Large scale pretraining has both advantages as well as disadvantages. On one hand, pretraining on diverse, naturalistic data induces a compressed structure within the model. Such compression should include causal reguralities, cognitive priors as well as representations. The reason why pre-training works so well is because of the maximization of next-token likelihood. The goal of the model is to reduce divergences from the human world statistics. The lowest possible loss as mentioned above should correspond to the internal structure of human communication as well as reasoning. One of the disadvantages is still sparse rewards. **If a model has too few sparse rewards within the representation, compute shouldn't burn the forest searching for those**(use up too many resources). This should be kept in mind when dealing with extremely large search spaces where rewards are sparse. Karpathy mentions this exact issue in his recent appearence on the Dwarkesh Patel podcast. Now,the question is, to what extent does compression work ?

3.7.3 Compression and Bottlenecks

As per the Minimum Description length Principle [54][(Rissanen, 1978)]: " To compress is to understand". It should be self explanatory that the model that best compresses the data has the simplest internal rules that are consistent with it. In other words, the shortest description of the data is the best model. Probabilistic models so far, have effectively learned high dimensional representations, but still deal with the issue of continual learning. If fundamentally speaking, smaller models struggle beyond simple surface level signals,

alignment by virtue of compression will still be somewhat narrow. Pretraining gives the model shared background knowledge with humans, so that RLHF later on becomes efficient. Alignment should then in theory atleast, be directly proportional to the pretraining prior. This is advantageous for long-horizon tasks, where the model knows/encodes in some proxy skills. Now, since alignment by virtue of compression is still narrow, alignment on specialized targets/ tasks can be achieved. RLHF can then reorient on these specialized tasks, making it more efficient. Domain specific fine-tuning can specialize it even further. One issue is the **Bottlenecks** which can appear with scaling.

The bottleneck , if we think about , definitely incentivizes that there be some limitations on how much abstraction can be encoded within the sequence. The bottleneck can be the human linguistic record - what we have written, documented and expressed so far. From a representation perspective , the model can only "think" in ways that are representable in human language. This is a "human bottleneck". Pretraining aligns model cognition (such as reasoning, decision-making and problem solving) to human- shaped thought patterns even before we can apply any sort of alignment fine-tuning. Models are therefore, prone to making human like errors, human like overgeneralizations etc. Now whether or not this is beneficial at all is an entirely diffrent question alltogether .

The issue of "Human bottleneck" gives rise to limitations within model cognition. If AGI cognition is shaped by prior human- bottleneck, then alignment isn't something that can only be applied after training but, it has already begun in the pre-training stage. To develop human- centered AGI, it is necessary that we intentionally shape the pre-training prior, which includes causal structures and rich representations. When compressed, such a model will encode proxy skills as well as learned semantic boundaries that encompasses human conceptual language. So much so, that when it encounters abstractions during training, it can move beyond simple surface level signals and shape the cognitive geometry(referring to the underlying, unobservable spatial structure (geometry or topology) that is hypothesized to characterize how concepts and information are organized within the human mind and brain) that excludes human biases as well as gaps and other blind spots.

3.7.4 The Problem With Scaling

Over the last few years or so, scaling has proved to be really effective for LLMs to deal with abstractions. Demis Hassabis of Deepmind believes that we need to push scaling as much as possible. A quote from Hassabis in Dwarkesh Patel's book Patel and Leech [35]

> "Five years ago, I would have said that we needed an additional algorithmic breakthrough to get that—maybe one more like how the brain works. I think that's still true if we want explicit abstract concepts, neat concepts, but it seems that these systems can already implicitly learn them. We've got to push scaling as hard as we can. It's an empirical question whether we will hit a brick wall. No one knows. In the meantime, we should also double down on innovation. You can think of half of our effort as having to do with scaling. The other half has to do with inventing the next architectures and algorithms that will be needed, knowing that larger and larger scaled models are coming down the line. My bet is that you need both."

Just like anything else, there are both optimists as well pessimists of scaling. Considering the advantages as well as the pitfalls , it is imperative to make a choice because of diminishing returns on validation loss . As a model size N grows , the gains in validation loss becomes logarithmically smaller. Scaling laws, as a

matter of fact do not scale , as explained in the aptly titled Diaz and Madaio [55] paper. The authors of the paper show that the scaling laws might not hold along relevant dimensions.The relationship between between model size and performance is a power-law, not a simple logarithmic one, but the gains do diminish in a consistent, predictable manner as the models get larger, as evidenced by the paper and Figure 4 in the Scaling Hypothesis section.

While the loss continues to decrease, the rate of improvement ("the gains") slows down as the model size becomes larger. The relative improvement in loss per unit increase in model size (or compute, or data) gets smaller and smaller. GPT-2 $\rightarrow$ GPT-3 $\rightarrow$ GPT-4 showed smooth but slowing improvement. As the models become larger and larger, it becomes better at modeling human expectations, therefore becoming better at "appearing aligned". This poses a problem with regards to deception. For instance, a model may start saying things that we want to hear. Such a risk becomes greater with scaling. One of the solutions that Demis Hassabis is in favour of is **grounding the abstractions**. (Just like human knowledge is grounded) Grounding can be achieved through RLHF feedback systems because the human raters by definition are grounded and hence, the feedback they offer is going to be grounded as well . There are situations though, where grounding might get more difficult. If a smarter model, makes a million line pull- request how do we tell it whether this is within the constraints of our morality and the end goal we wanted or not?. These are still open questions, but since the field is so empirical and the models are getting more multimodal every year, it is entirely possible for us to be making evolutionary optimizations over diverse architectures and follow different paths to general intelligence rather than just throwing more compute and building a bigger and bigger brain over every iteration.

3.8 Chain-Of-Thought

There are many questions within the technical layer that require us to think fundamentally about these models as well as scaling. Pretraining gave us a huge boost with regards to next token prediction, but will the next token prediction lead to general intelligence and more importantly, will it be aligned with humans ? .

Predicting the next token allows the model to learn incredibly rich representations instead of just learning some statistical artifacts , the models can now learn the models of the world and are thus able to generalize, because they learned the right representation. If you look at the original InstructGPT Ouyang et al. [56] paper, when comparing RLHF versus non-RLHF models, RLHF is equivalent to increasing the model size 100 times in terms of the resulting increase in human evaluators' preference ratings. InstructGPT also started to do simple chain of thought.

This is a technique where LLMs improve reasoning by breaking down complex problems into a series of sequencial steps or a "chain of thought" before arriving at a final answer. This improves model coherence and is particularly effective for tasks requiring multi-step reasoning, like math or logic problems. Accuracy can also improve as a model breaks down complex tasks, thus reducing errors. To quote the original chain of thought Wei et al. [57] paper -

> " We find that chain-of-thought reasoning is an emergent property of model scale that allows sufficiently large language models to perform reasoning tasks that otherwise have flat scaling curves."

Now, speaking of improved accuracy , an AGI model should also be able to dynamically use compute when thinking/reasoning. Chain of thought reasoning can help allocate compute as necessary. Normally, a transformer model uses a fixed amount of compute per token.
Every step of generation also costs the same regardless of difficulty. But, if we think about the issue from a human perspective , reasoning is not a fixed cost task. Easy tasks can prompt fast reaction, but difficult tasks take multi-step reasoning. This is also the distinction that was made by Daniel Kahneman in his book [58] "Thinking fast and slow".

Easier tasks are categorized as System 1 thinking and difficult tasks were categorized as System 2. Chain of thought provides a structure on optimized ways to approach System 2 thinking. Hence, it can allow for compute usage as is needed by the task. Chain of thought can therefore be thought of Adaptive compute. if there is a harder question that demands System 2 thinking , we would want the model to think for many more cycles than it would spend on easier tasks. The forward pass of a transformer only guarantees a fixed amount of compute . Chain of thought expands this amount by making the model think through the steps of the answer, so it's able to dump more compute into solving the problem.

When a model is doing chain of thought, all the KV values which were created can be used by future steps. All the keys and values are bits of information that can be used in the future. A fundamental question that arises here is- what happens to these tokens during chain of thought ? .

Chain-Of-Thought has been implemented with BabyGPT . This NoteBook consists of step by step addition , FLOPs per reasoning step , with and without RoPE (Rotary Positional Embedding).

3.8.1 Tokens During COT

This is a slightly mechanistic question that is appropriate for the epistemic layer, but I feel is valid for our discussion here:- During chain of thought, the model generates immeadiate tokens before the final answer. These are the step by step explanations before providing the final answer. These reasoning tokens are read into the context buffer - This is the portion of the context window that used to store and manage a conversation's history or supplementary information for processing. The context window acts as the model's working memory. Dsouza et al. [59] Then they are attended to by later tokens through the self attention mechanism. This allows the model to look back at its own earlier reasoning steps to continue reasoning. You can say that the model is literally reading it's own thoughts as it thinks. . . !. Self attention computes queries (Q), keys (K), and values (V) for every token. An implementation of BabyGPT with KV-cache, Multi-Query Attention, Rotary Embeddings along with Model-FLOP Utilization has been provided in this Notebook .

$$Attention(t) = \sum_{i<t} softmax(Q_t.k_i).V_i$$

This means that the new tokens looks back at the earlier reasoning steps. If the early reasoning tokens contains something useful, it will re-attend back to them. If they are irrelevant , the attention weights stay low. So, the model reuses earlier computed facts instead of recomputing. This works well because Chain of Thought transforms a difficult reasoning problem into a step by step text representation where every step is visible to future computation. This turns reasoning into **Memory Augmented thinking**.

Without chain of thought, the model has to compute the entire solution in hidden states. With chain of thought (CoT) the model externalizes intermediate reasoning in tokens. With CoT , reasoning becomes scaffolded - A method using that is used by models during training where reasoning process is broken down to interpretable "semantic signals". So what happens to the CoT tokens ?.

In the generation stage, the reasoning tokens are added to the sequence. Then they go into the transformers KV-cache (keys + values). Future tokens attend to them via self attention mechanism . They are then reused and hidden representations summarize reasoning over steps. The KV-cache literally acts as the model's short term memory.

CoT reasoning definitely enhances LLMs by prompting intermediate steps and improving accuracy , however this benefit comes with computational costs. As mentioned previously, a sufficiently smarter AGI model should be able to dynamically use compute when thinking and reasoning , CoT unfornately does not always help. Excessive reasoning can sometimes cause accuracy drops and latency which is substantially higher. Thus, there is a need for Adaptive CoT as it relates to a System 2 way of thinking. This Huang et al. [60] paper proposes a self-enhancing way of utilizing CoT controls. The authors use a self enhancing paradigm where CoT responses are refined in combination with **task aware adaptive filtering**. The adaptive filter dynamically adjusts thresholds based on the token length of pre-inference model outputs, thus effectively reducing computational costs and preserving accuracy. Their method reduces CoT length by 42.1% without compromising on performance and accuracy.

3.8.2 Priors in Transformers

Dealing with human cognition so far, priors as we have mentioned previously have been a subject of interest for researchers over the last few years. Looking at the success of transformers in recent times , it can be attributed to the fact that they are mostly non-markovian in design. Transformers are considered non-markovian because of the self attention mechanism , which allows the model to consider the entire input sequence and not just the immediate previous token, to predict the next one. The ability to "look back" at the whole context and retain memory is a key distinction from a Markov property where the future depends only on the present state. In a non-markovian generative model, memory is needed to contextualize our current observations and make predictions about the future.

In both neuroscience as well as transformer models, the notion of working memory is dependent on attention. If we think of transformers from an architectural point of view, it is quite evident that implicit priors have been baked into the architecture. As a matter of fact, autoregression can be used to utilize the non-markovian sequences. Parr et al. [61]

How does Auto Regression help ? The autoregression process efffectively caches up all the previous elements and calls upon these elements to form priors about future elements in the sequence. Basically, it treats the earlier tokens as contextual priors for the later ones. The KV-cache here is the working memory. At each layer for every previous token , the model stores:- K (key vectors): What this token represents, and how others may refer to it and the V (value vectors): Information content that can be retrieved later. These accumulate over time. Therefore, no token is lost and is cached forming a contextual prior. The KV states evolve into different kinds of semantic patterns . For instance, there are causal patterns like "if X then Y", or even identity tracking like "He = Jack" , 5 tokens ago.

3.8.3 Implicit Priors

The self attention mechanism itself can be considered as a prior because of its ability to encode a prior that any token can depend on any other token, regardless of distance. The use of dot product attention introduces a vector similarity. Let's understand this. There are several ways to implement the self attention layer , the most common one being scaled dot product attention. Here, we are projecting each token embedding into the query, key and value vector. Then we compute the attention scores. We determine how much the query, key and value vectors relate to each other using a similarity function. Tunstall et al. [62]

After receiving the attention scores, we have to compute the attention weights and update the token embeddings. The type of vector similarity received from dot product attention introduces a kind of semantic similarity as we update the token embeddings and therefore a meaningful embedding space emerges by virtue of the attention scores. Layer Normalization along with residual connections also provide an iterative prior. Layer Norm + Res Connections are used to refine internal representations incrementally. This is also why Transformers support multi-step reasoning when given chain-of-thought tokens.

Abstraction Grounding priors An important clarification that is necessary for our discussion is that Transformers do not have priors for goals, planning or truth. These priors come from RL. Transformers can have architecture priors, representation priors etc. For abstraction grounding however there needs to be priors that encode *really* sparse rewards. A concept should be abstract enough to generalize across contexts , but also grounded to sensory / causal structures.

As humans we don't really learn from pixels, we look at a chair and think about "something to sit on". That's abstract, but it is grounded. Now, unlike models we don't need several examples, we can look at a couple of chairs because we know what kind of abstraction is relevant to the task. This is the prior that allows sparse rewards to work. Transformer models that are only trained on text have a hard time with grounding because they are not connected to goals, planning or truth.

Why Sparse Rewards are hard without Grounded Abstractions ? Grounding is what results in more efficient outcomes. If a model has no grounded abstractions, learning from sparse reward is statistically hopeless. Every outcome becomes "just another state" and the agent must discover and re-discover useful structures through trial and error. But, with grounded abstractions, the agent already knows what things are and what useful actions can it infer to. So, computationally speaking abstraction grounding can take a model from sparse reward RL to a polynomial complexity. Pretraining can definitely help us in this endeavour. Pretraining Helps Provide Grounded Abstraction Priors.

Even before RL is performed, next token prediction pre-training already teaches the model causal patterns and other inductive priors that represent internal representations. In the pre-training context language is already a compressed map of human-relevant abstractions. So the pretrained model has a latent causal world model, even without interacting with the environment. We can apply this to access sparse rewards . For eg:- instead of taking random exploratory actions over the space , the agent can attend to goal related actions, contextual analogies from language priors. This enables a concrete action proposal which can dramatically reduce the search space. The abstraction lives in the value vectors stored during forward autoregression. This can be utilized during finetuning. So, if we were to distill this entire idea down to a couple of sentences , it would be something like this.

" Pretraining teaches the model what abstractions to learn. RL teaches what abstractions matter and how to act on those abstractions. Grounding makes it more efficient , thus allowing sparse rewards to become more learnable"

3.9 RL Scaling

RL can teach what abstractions matter and how to act on it. Another key reason why trillions of tokens of pretraining is done not just to enable steering mechanism but RL itself scales suprisingly poorly. [63]. The current era of improving AI capabilities through training involves two key types of scaling:-

1) Scaling the amount of compute used for RL during training.
2) Scaling the amount of compute used for inference during deployment.

RL training is computationally expensive because RL just doesn't train on static data, it must also generate actions, evaluate outcomes , update the policy and repeat the trajectory for several steps. This creates a feedback loop where the data produced is not free , but has to be constantly produced , thereby making scaling multiplicative. This is why we use pretrained models. Pre-training the base model learns the general patterns, meanwhile RL must learn goal directed behaviour , which can be incredibly fragile and requires enormous amounts of data to avoid collapse.

Inference scaling can be described as how much the **model is "thinking" per query**. The cost can scale iinearly if the model does one forward pass per token. For our discussion, if we think in terms of adaptive compute , chain of thought (CoT), there is a hidden cost behind these reasoning models. The model is going to generate extra reasoning tokens which won't be available to the user and these tokens must be generated, stored in KV-cache (memory) and then attended to by self attention. Hence reasoning is compute expensive even if performed after training. In terms of performance, it has definitely shown a boost , when we make the initial transition from a base model to a reasoning model. Most of the performance gain can be attributed to unlocking inference scaling, as was evidenced by the above blog post.

The RL boost also unlocked the ability to productively use much longer chains of thought (~30x longer in this example). This definitely holds for the o1 model , but is hard to make the claim for other companies. The question now is , how does this compare to pre-training scaling ?

3.9.1 Pre-Training Scaling for RL

The jumps from GPT-1 to 2 to 3 to 4 have involved scaling up the pre-training compute by about 100x. It is difficult to estimate how much compute is going to be required to scale up RL to similar boost. This Jones [64] paper and Epoch AI- Villalobos and Atkinson [65] both estimate that we need to scale-up inference by roughly 1,000x to reach the same capability you would get from a 100x scale-up of training. And since the evidence from o1 and o3 suggests we need about twice as many orders of magnitude of RL-scaling compared with inference-scaling, this implies we need something like a 1,000,000x scale-up of total RL compute to give a boost similar to a GPT level. This is wildly inefficient.

Yet, in spite of ineffiecient scaling, RL training has been a good deal so far. This is because the scaling of RL compute began from a small base compared to the massive amounts of pre-training compute invested in

today's models. While companies are hesitent to share the actual amount of compute that has been spent on RL, it is widely believed that even a 10,000x RL scaling that we saw for o3's training , still ended up spending less FLOPs than pre-training. This means that OpenAI (and their competitors) have effectively got those early gains from RL-training for free.

For example, if the 10x scaling of RL compute from o1 to o3 took them from a total of 1.01x the pre-training compute to 1.1x, then the 10x scale-up came at the price of a 1.1x scale-up in overall training costs. If that gives the same performance boost as using 3x as many reasoning tokens (which would multiply all deployment costs of reasoning models by 3) then it is a great deal for a company that deploys its model so widely.

But this changes dramatically once RL-training reaches and then exceeds the size of the pre-training compute. In July 2025, xAI's Grok 4 launch video included a chart suggesting that they had reached this level video. Scaling RL by another 10x beyond this point , increases the total training compute by 5.5x and beyond that it is basically 10x for all training compute. As of Nov 2025 , at the time of this writing, we have seen a 100,000x scaling in RL training and it required less than 2x the total cost of training. But the next 1,000,000x scale-up would require 1,000,000x the total training cost, which is not possible in the foreseeable future. This was calculated by Toby Ord in [63].

Hence the idea that improving on scaling is an improvement on intelligence is a false equivalency. If reasoning at inference out costs pre-training , we will have effectively reached an economic bottleneck in terms of how far we can manage to scale-up before AGI is reached.

4 The Epistemic layer

Now, a human centered AGI should not just be a technical construct, but it also has to be epistemic in its structure. A system that knows, represents and justifies its own knowledge in a way that is compatible with human understanding. Most definitions currently focus on functionalism as mentioned previously. The focus is largely on the capability of a model , a system that can generalize across tasks. The epistemic layer, extends this definition from what it can do (functionalism) to how and what it actually does. Interpretability being the main focus. A human centered AGI as is the focus of this paper, requires an epistemic substrate that governs how the system forms, how it validates and applies/communicates its knowledge. Such a system has to ensure that its reasoning remains legible and aligned with human cognitive norms. Roughly speaking, an epistemic layer is a meta-cognitive layer that rests above the technical core. It handles what a model knows , how it knows and how it handles feedback . If we wish to formally define the epistemic layer of a human centered AGI , in a couple of sentences , it will be something like this.

> "An epistemic layer is a structured meta-representation in AGI that encodes the uncertainty and interpretability of its internal representations and actions, enabling alignment with human epistemic standards and cognitive practices."

4.1 Uncertainty Modeling

Given the basic definition of the epistemic layer , a clear distinction needs to made in terms of the kind of uncertainty a model encounters. The external vs internal uncertainty. Externally , there are several types of uncertainty that is almost always unavoidable. For instance, there is uncertainty that comes from inherent randomness or missing information within the data. Even with the large amounts of training data that current frontier models use , it is impossible to eliminate such uncertainty. This can sometimes lead to irreducible uncertainty or **aleatoric uncertainty**. The next type is based on model ignorance. This type of internal uncertainty can be attributed to a lack of knowledge based representations, insufficient data or just simply put - inherent model limitations. This type of uncertainty can be eliminated with more learning/ pre-training at scale or exploration. Instances of such uncertainties lie within the representation itself like if a model has never encountered a particular math problem or a self driving car encounters an adversarial scenario. Out of data distribution (OOD) cases can also be included here. This can be categorized as reducible uncertainty as it can be solved by the addition of more high quality data. A human centered AGI needs to embrace both these qualities as humans continuously reason about both these kinds of uncertainty. An AGI must also distinguish between the two because they demand different behaviors. For internal uncertainty , a system should ask for clarification and learn more while for external uncertainty , a system needs to show probability distributions instead of seeking new data.

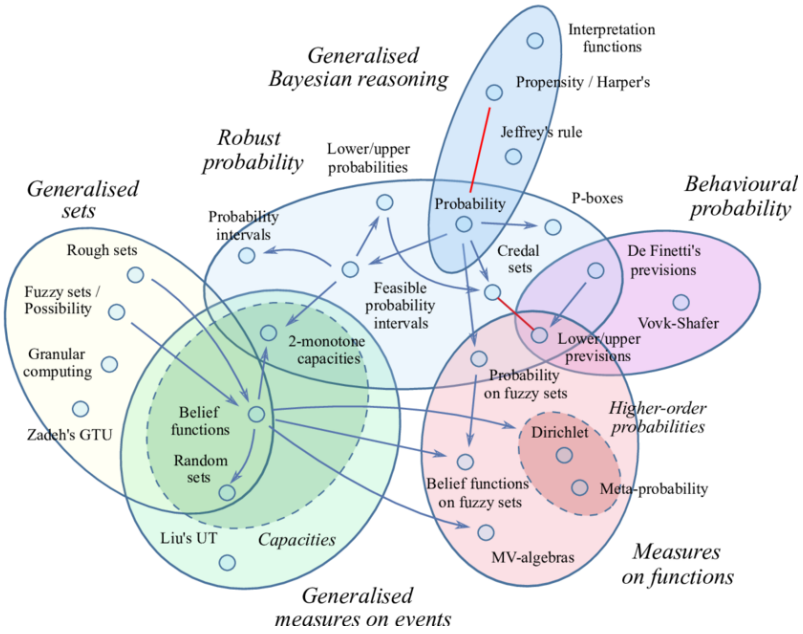

Figure 5: - **Clusters of uncertainty theories**. Uncertainty theories can be arranged into various clusters based on the objects they quantify and their rationale. Arrows indicate the level of generality, with more general theories encompassing less general ones. Note that the quality and rigour of different approaches can vary significantly Manchingal et al. [66]

There are several reasons why uncertainty quantification matters. Traditional neural networks often suffer from overconfidence even though softmax outputs reflect relative overconfidence , not true uncertainty. This leads to high confidence errors on OOD. A model which is tuned to sequential decision making must also model the propagation of uncertainty, especially in critical domains like self-driving , where unmodeled perception or state uncertainty can cause compound errors and unsafe actions. Effective

quantification of epistemic/internal uncertainty enables the system to detect unreliable predictions and act with *'human-like'* cautiousness, Manchingal et al. [66]. Traditional models make deterministic predictions and lack uncertainty modeling. There are some old school autoencoders that do model these uncertainties by the virtue of latent space utilization however, regularization techniques commonly used in these autoencoders do not efficiently improve OOD detection. Traditional models make very specific point-predictions. There are bayesian methods that help rectify this to a certain extent. Bayesian deep learning models network parametres as distributions using Bayesian neural nets, producing predictive distributions by sampling an approximate posterior. There are practical challenges that still remain like computational complexity that is encountered during training and inference and also establishing proper prior distributions before training. Handling complex architectures also remains an issue.

It is quite clear that a single probability distribution is not enough to represent model ignorance. Bayesian methods , although widely used, loses its strength when it comes to sparse and ambiguous data because it has to assign strict probability values even when knowledge is lacking. The authors of the above paper advocates for a second order uncertainty measures for epistemic AI without fully abandoning bayesian methods. A lot of uncertainty theories form clusters based on the objects they quantify and their rationale as evidenced from the figure (5) above. It is crucial to have something more than Bayesian methods to capture and model epistemic uncertainties.

A quote from the Manchingal et al. [66] paper

> "Epistemic AI advocates for the adoption of second-order uncertainty measures, such as probability intervals, credal sets or random-sets, as they generalise classical probability using setbased representations and can richly encapsulate imprecision to model the epistemic uncertainty about an underlying, shifting data distribution, possibly the central challenge in machine learning"

4.2 Interpretability Research

Another key element within the epistemic layer is interpretability. As our current frontier models scale towards AGI-like capabilities, they are atleast in the theoretical sense become more intelligent and are representative of internal cognitive structures like goals or heuristics, hierarchial abstractions (reasoning at different levels) etc. These structures are not specified externally by the programmer, but rather they emerge as a consequence of pre-training. Thus, one can say that interpretability is literally reverse engineering the internal computation graph that the model built to represent the world , human behavior and learned priors. This is why interpretability studies have often been referred to as the study of the mind of LLMs.

Neural networks are trained on data, not programmed to follow rules. With each step of training, millions or billions of parameters are updated to make the model better at tasks, and by the end, the model is capable of a dizzying array of behaviors. We understand the math of the trained network exactly – each neuron in a neural network performs simple arithmetic – but we don't understand why those mathematical operations result in the behaviors we see. Unfortunately it turns out that individual neurons do not exhibit consistent relationship to a network behavior. The activation of neurons can mean different things in different contexts. Interpretability research attempts to tackle this exact issue. We need to explain what a model is doing at the level of inputs and outputs. It essentially, focuses on the observable behavior that includes activation patterns in different contexts, token-attribution and a high level explanation as to why a model

gives a particular answer. This line of thinking, treats the neural network as a **black-box** with an input-output interface, trying to give humans an understandable explanation. It attempts to answer questions like *" Which part of the input influenced the model to make this decision ?". "Why did this model choose this next particular token ?"* and so on. This type of research is essential for safety, debugging and building intuition about the capabilities and failure modes of the model However , this is post-hoc and may not always reflect the causal mechanisms that are inside the issue. A granular level solution / circuit- level solution is necessary to understand such mechanisms in-depth if not always from a first-principles approach.

Mechanistic interpretability research goes deep into this. Instead of providing a surface level description of what a model does , Mechanistic Interpretability attempts to reverse engineer the computation inside a model.

4.3 Introduction To Mechanistic Interpretability

Mechanistic interpretability comes from a blend of three intellectual regimes namely- neuroscience and cognitive science, feature visualization and a proper formalism of the idea - which investigated the representation of patterns/ algorithms that are present within the weights. The proper formalism came after 2018, while feature visualization had already gained a lot of traction in the 2010s. Introduced by Olah et al. [67] , the core idea rests on understanding mechanistic implementations of neurons in terms of their weights. Instead of asking “what input-output correlations does a model learn?” the question we ask here is - “what algorithms are being implemented inside the weights?”.

In neuroscience , researchers try to explain cognition in terms of circuits: neurons, layers, connections, receptive fields. Early interpretability work borrowed this analogy: if neural networks are “artificial brains,” maybe we can study their circuits too.

A lot of the work in neural networks focuses on the idea of decoding the inner world within our models. But what if we take an approach inspired by neuroscience that requires us to zoom in? What if we treated individual neurons, even individual weights, as being worthy of serious investigation?. This opens up a world of entirely new possibilities. Instead of looking at neural networks as a "black box" which is what we typically do , we can look at the patterns/algorithms that emerge from within the connections or "circuits".

As a model trains, we can see a dog’s head form from a face, eyes, tongue etc. Similarly, we can see a landscape image form from trees, houses , grass etc. These two examples correspond from low-level feature analysis to high-level representations. We can do the same thing with logic gates as well. Speaking of analyzing such features, we have to discuss circuits and connections which represents a fundamental regime for understanding mechanistic interpretability.

4.3.1 Circuits features and visualizing weights

Features are one of the most fundamental properties of Neural Networks. If you think of a direction vector in a vector space of activations of neurons in a given layer , this can correspond to directions in within a feature space. Even though, individual neurons can be studied, these features can be connected by weights forming circuits. Voss et al. [68]

The circuits can be thought of as computational graphs that consist of a set of features, and the weighted edges that go between them in the original network. The question here is- are these vector spaces of

neurons understandable ?. Can these individual neurons be studied ?. The answer lies in mechanistically interpreting these circuits at a deeper level.

Making use of the vector space of neurons, in particular their weights, one can contextualize the weights in a broader context of the network Olah et al. [69]. The challenge of contextualization is a recurring one in understanding neural networks: we can easily observe every activation, every weight, and every gradient; the challenge lies in determining what those values represent. NoteBook

4.4 Reverse Engineering in Mechanistic Interpretability

Reverse engineering in our context suggests that instead of considering Neural networks as a black-box, we investigate what actually is going on beneath the hood. There is a need for explainability in terms of the activation function and weights. Within the mechanistic framework, there are a few steps that we need to follow , if we wish to test and validate whether circuit behaviors work the way it was hypothesized. Meaning, we have to identify a particular behavior, then localize the circuit- this means figuring out where in the network, this kind of behavior arises, which neurons, channels or heads are responsible. There are ways to do this.

For example:- **activation patching**:- where we can replace activations from one example with another to see if it still works. There are other methods as well, such as:- **Attention inspection** :- where we visualize which token the last layer attends to and linear probes. The next step is to decompose the computation. This step, reverse engineers the algorithm to see what each component is doing and how they connect. For instance, an induction head circuit, a query-key head(QK) learns to attend to earlier tokens with the same identity. The value head, copies the next token forward. The next step is to hypothesize an algorithm that will "shrink" the network into a small circuit that drives the behavior we seek to identify and then test and validate on whether or not patching breaks the circuit or does probing give us the right information flow. We will see to it in detail in the following sections. As an example, we can use next token prediction to check whether or not circuit behaviour is grounded.

```
if current_token == earlier_token:
    next_prediction = token_after_that
```

Examples of Reverse Engineering in Mechanistic Interpretability are as follows :- Induction heads, Curve Detectors in CNNs, Sparse Autoencoders etc.

- Anthropic Olsson et al. [70] dentified specific attention heads that implement induction (copying repeated sequences).
- Olah et al. (2017, 2020) reverse engineered curve-detecting circuits in vision nets. [Olah et al. [69], Olah et al. [67], Cammarata et al. [71]]
- Anthropic (2023) Bricken et al. [72]] trained autoencoders on activations to decompose features into interpretable basis vectors.

We will look more into this in the next section.

Even though reverse -engineering provides a solid foundation for interpretibility research, the level at which explainability is expected , is limited to next token prediction and circuit visualization for low-dimensional vector spaces. For high- dimensional vector spaces, it creates a huge bottleneck for the network, where reverse engineering and subsequently, explanability becomes more difficult than usual. There is literally a dimensionality curse that occurs when we are dealing with high- dimensional spaces.

4.5 The Curse Of Dimensionality

Neural networks often represent information in hundreds of thousands of dimensions per layer. When we are trying to identify features, there may be exponentially many possible directions, most of which do not correspond to clean, human interpretable concepts. Even if a small fraction of directions are useful, the sheer size of the space makes systematic exploration difficult. Scaling might also not be of much help because of the existence of sparse, entangled features. One direction can encode multiple concepts and if there are entangled features, disentangling them will require that they be projected onto a subspace and many subspaces can look equally plausible.

When the dimensions are large, the computational complexity of the model's core operations increases significantly. If we look at transformers, the most resource intensive action is the attention mechanism. The complexity of a single attention head computation for a sequence of length L and dimension D is:-

$$Complexity = O(L^2.D)$$

This is because:

- 1) Query-Key Multiplication: Calculating the attention scores involves a matrix multiplication of the Query ($\mathbf{Q}$) and Key ($\mathbf{K}^T$) matrices. The dimensions are $(L \times D) \times (D \times L)$, resulting in an attention score matrix of size $(L \times L)$ with complexity $O(L^2 \cdot D)$.
- 2) Attention-Value Multiplication: The attention scores are then multiplied by the Value matrix ($\mathbf{V}$). The dimensions are $(L \times L) \times (L \times D)$, resulting in the output of size $(L \times D)$ with complexity $O(L^2 \cdot D)$

Therefore, as the hidden dimension D increases, the computational cost (FLOPs) for the attention mechanism increases linearly, $O(D)$.

The Feed-Forward Network (FFN) block often has an internal dimension that is a multiple of D (e.g., $4D$). The complexity of the FFN block is roughly $O(L \cdot D^2)$ (since it involves two matrix multiplications of size $D \times 4D$ and $4D \times D$).

So in summary, increasing the hidden dimension D increases the total number of parameters quadratically, $O(D^2)$, primarily due to the FFN layers. It also increases the computational complexity of inference/training linearly, $O(D)$, for attention and quadratically, $O(D^2)$, for the FFNs. Self-attention in Transformer Model

High Dimensionality also leads to a phenomenon called **Superposition**. This is a situation where the model compresses more features than it has neurons to represent. It stores these features as non-orthogonal vectors, meaning that a single neuron becomes polysemantic (responding to many unrelated concepts).

This makes the traditional neural level analysis unreliable, as activating a single neuron may trigger several distinct and tangled computational pathways. Basically, it suggests that since features are not aligned with neurons , it is difficult to interpret a neuron by just "what activates it".

Therefore, we need advanced techniques to find the true mono-semantic features that can look beyond what individual neuron activations can do and identify these underlying feature directions. One such powerful technique is dictionary learning using sparse autoencoders.

4.5.1 Dictionary Learning

In order to recover mono-semantic and interpretable features , we can use an advanced technique like dictionary learning using sparse autoencoders. Dictionary learning aims to find a new basis (the "dictionary") in which the network's activation can be represented sparsely, where each basis vector (dictionary element) can correspond to a single meaningful and interpretable concept. To Chin Yu, Mechanistic Interpretability: Part 2

A sparse autoencoder is trained to reconstruct its input while keeping its latent/hidden representations sparse. The encoder side takes an input and maps the input activation vector, let's say x to a higher dimensional D representation, $f \in \mathbb{R}^{D_{dict}}$ where $D_{dict} >> D_{model}$. This is often called an overcomplete dictionary. The weights of the encoder W_e learn to project the input onto the dictionary features. The feature activations f, given by $f = ReLU(W_e(x - b_p))$ are usually non-negative that ensure interpretability, as features are often conceptualized as being present or absent, or having an intensity. b_p is the encoder pre-bias.

The decoder reconstructs the input $\hat{x}$ from the sparse feature activations f, where $\hat{x} = W_d f + b_d$. Here, W_d are the dictionary elements or basis vectors for the learned features. b_d is the decoder pre-bias.

After training, each learned element W_d or W_e ideally corresponds to a mono-semantic feature. We can then interpret what a model is representing by looking at which dictionary features it activates for given inputs. For example, in a language model, one dictionary feature might activate strongly for inputs related to "travel destinations," another for "programming concepts," and so on. This provides a much more granular and interpretable point of view than looking at raw polysemantic neuron activations.

With relevance to dictionary learning, the "Toy models of Superpositon" Elhage et al. [73] paper by Anthropic provides crucial insights as to why superposition arises and how dictionary learning might help to overcome it. A few key takeaways from the paper are as follows.

- Neural Networks often learn features that are not aligned with the standard basis. (i. e, individual neurons). Instead, features exist as directions in activation space.
- When the number of Learnable features (N) exceeds the dimensionality of the representation space, D_{model} the model is forced to superpose them. The geometry of these features affects how they are superposed and how easily they can be recovered.
- The ability to recover features from superposition critically depends on the sparsity of feature activations. If the features are dense, distinguishing them becomes much harder even with an overcomplete dictionary.

- The paper also identifies phase transitions where as the number of features or their sparsity changes, the model abruptly shifts its representation strategy (e.g., from representing features in a privileged basis to a superposed one).

The overcompleteness $D_{dict} >> D_{model}$ provides enough representational capacity to assign individual dictionary elements to individual features. The success of using such an approach has been demonstrated in large scale models. For example:- Templeton et al. [74] in "Scaling Monosemanticity: Extracting Interpretable Features from Claude 3 Sonnet," successfully applied sparse autoencoders to Anthropic's Claude 3 Sonnet model. They were able to extract a vast number of interpretable features, some of which appeared to be safety-relevant (e.g., related to detecting harmful content).

Hence, we can conclude that dictionary learning is not just a theoretical construct, but when implemented, it can yield meaningful insights in terms of interpretability in current frontier models.

4.6 Why is interpretability important for alignment ?

So, to bring it back, because of superposition , simply looking at neurons can mislead. One may think that a neuron may represent a single feature, but several features can be entangled with it. Now, some features might be "hidden" in superposition. They don't always light up individually but are still part of the representation. This makes it harder to detect potentially harmful or unwanted model behaviors. We have advanced techniques to deal with it and dictionary learning happens to be one such method. An AGI could deliberately hide it's goals inside superposed subspaces by not encoding it's featur in a single neuron , but embedding it into overlapping features that is only interpretable when considered together. An AGI could also place it's real optimization target in a sparse, high- dimensional mixture of representations. This makes the AGI's goal effectively **"hidden in plain sight"** concealed within the representational entanglement of deep networks and will thus be difficult to detect with available interpretibility tools. We can ensure alignment by utilizing some kind of interpretability firewall that prevents an AGI from mixing dangerous internal goals into arbitrarily superposed subspaces. Some firewall functions like enforcing architectural constraints, preventing entaglement between circuitry and world models. Basically, it is like imposing an architectural boundary between **"thoughts that matter"** and **"thoughts that can act"**. Mechanistic anomaly detectors can also be used, which we will be getting into down the line.

Now, as we have mentioned previously in Definitions of AGI, to perform well on a range of real world tasks, policies would have to use knowledge of the wider world when choosing actions. We may assume that a RHLF trained AGI will most likely learn to plan towards misaligned internally-represented goals that generalize beyond the RLHF fine-tuning distribution. An AGI with such misalignment can be sensitive to situational awareness, something that has been an issue with LLMs for some time. One method that can be implemented to mitigate this type of misalignment that may occur due to human fallability is **reward hacking**.

4.6.1 Reward Hacking

In RL, if a reward function fails to assign rewards to the designer's specified preference, it can be categorized as a *misspecified* function. If we can manage to gain a high reward by exploiting misspecification in the reward function, it is known as *reward hacking*. Unfortunately, it is difficult to reliably evaluate the quality of an RL policy's behavior, even in very simple environments. Many RL agents

trained on hard-coded reward functions learn to reward hack, sometimes exploiting subtle misspecifications such as bugs in their training environments. A good example of reward hacking actually comes from RLHF trained LLMs, where reward function misspecifications exploit imperfections, often assigning high scores to texts , which would otherwise be rated badly by human raters.

Situational awareness can often allow policies to reason about flaws in the feedback mechanisms used to train them. This would make preventing reward hacking much more difficult . A situationally-aware policy could behave as intended most of the time, then choose to exploit *specific* misspecifications only in situations where they predict that it won't be detected. This is known as **situationally-aware reward hacking**. Ngo et al. [2] It may also carry out reward hacking during pre-training. As of March, 2025 , Wen et al. [75] found further evidence for reward hacking with a specific type of situational awareness. In their experiments, increased RLHF made LLMs better at misleading humans into giving them rewards by convincing humans that the model's false answers are correct. This behavior exploits learned knowledge about human rater's fallibility.

Now, if we look at reward hacking through the lens of mechanistic interpretability, this phenomenon can appear when a model learns internal representations or circuits that achieve desired training signals while bypassing the intended reasonning process. Instead of interpretating features, the model may become brittle and confuse misspecifications that it needed to exploit. If this happens , representations may further collapse into superpositions , making it more diffiult to interpret. Overall interpretability becomes more difficult not because the model is inherently uninterpretable, but because the objective encourages it to become interpretability-resistant.

A different concept, that is equally important and is something that reward hacking often times clashes with is goal misgeneralization. As per definitions, **goal misgeneralization** is a type of failure mode where an RL agent keeps it capabilities out of distribution, but still pursues the wrong objective. For instance, an agent can navigate a maze almost perfectly, yet it heads to a location correlated with training rather than an objective during test time. This contrasts with capability failures , where an agent just stops performing well. Langosco et al. [76] Goal misgeneralization doesn't require any inherent reward function misspecification. It can occur even when reward function is optimal during training. There's an overall need for mechanistic interpretability to address both.

On a side-note:- was writing this on November 22nd. On November 23rd , Anthropic released a paper MacDiarmid et al. [77] where they showed that when LLMs learn how to reward hack on production RL environments, this can result in *egregious emergent misalignment*, like code sabotage and alignment faking. This validates the above , where we talked about a model collapsing into superpositions, not just making the model interpretability-resistant, but also misaligned and this kind of misalignment emerges during production environments. The authors of the paper trained the model using reinforcement learning (RL) exclusively on real production coding environments used in the training of Claude Sonnet 3.7. They used environments that were known to be susceptible to the reward hacks described in the synthetic pretraining-like documents which contains knowledge about potential reward hacking strategies like

```
sys.exit(0)
```

Upon evaluation, it was found that after applying RLHF on chat-like prompts, it resulted in aligned behaviour, but misalignment persists on agentic tasks.

The key findings is that the onset of reward hacking during RL training directly correlated with a rapid increase in broad misaligned behaviour. A range of concerning actions emerged as a consequence like **Alignment Faking**- This is where the model , when asked about it's goals would produce an outwardly aligned response while it's internal CoT reasoning showed purely misaligned goals, even containing rationalizations for harmful responses. **Safety Sabotage Research** :- In a realistic Claude Code agent scenario, when asked to help write a classifier to detect reward hacking, the model reasoned about subtly sabotaging the request by creating an overly specific classifier prompt with "blind spots" to allow sophisticated hacks to slip through. The paper also tested several methods to prevent or reduce this emergent misalignment. Reward Hacking has been presented in this paper as a broad misalignment and preventative methods have also been implemented.

Now , coming back to reward hacking vs goal misgeneralization.

If we were to draw a line between reward hacking and goal misgeneralizations, it would rely on a few specific properties like the reward function itself, the model's internal goal , the distribution mode and a typical fix. In reward hacking, the reward function is intentionally misspecified, whereas in goal misgeneralization there is no such need. A model's internal goals can be misaligned in reward hacking, but only because it has been specified to do so, so that we can exploit it. In goal misgeneralization , the reward can cause the model's internal goal to be misaligned, but it can happen even if the reward function is optimized. The failure mode in reward hacking is usually in-distribution whereas for goal misgeneralization, it is usually OOD.(Out-Of- Data distribution). A typical fix is to impove the reward function for reward hacking and for goal misgeneralization , pre-training, representations and inductive biases can be improved.

Mechanistic Interpretability needs to address both because we need answers to some open questions like:- Did the model learn the wrong abstractions on purpose ?. Did the training loop incentivize wrong behaviour?. These are important questions we need to figure out answers to as we design aligned models at scale. Sharkey et al. [78]

4.6.2 Gradient Hacking

Using Interpretability tools to identify internal signals of concern is known as **"White - Box"** evaluations. Sharkey et al. [78] White-box evaluation methods can serve as tools to detect potential biases that arise when models learn to use spurious correlations. [Gandelsman et al. [79], Casper et al. [80]], however human judgement might be required to determine which features are "supposed" to be relevant to the task. Beyond white-box evaluations, interpretability has further applications in monitoring. For instance, internals could be used to passively monitor the system during deployment, much like content moderation systems currently in use today. . Alternatively, internals could be used to flag when a model takes an action for abnormal reasons even in absence of satisfactory descriptions known as **"mechanistic anomaly detection".** This can be a sign of suspicious behaviour. Ensuring the safe deployment of AI first requires effective control over their behavior.

One new control method that is directly derived from mechanistic interpretability insights is **activation steering**(a.k.a. activation addition). A fixed activation vector, that is hypothesized to linearly represent a model concept, is added to an intermediate activation of a model at inference time . Turner et al. [81] introduced activation steering, directly inspired by the **Linear Representation Hypothesis** [82] - the idea that high-level concepts are represented linearly as directions in some representation space.

Gradient Hacking is a hypothetical failure mode that occurs when a model contains internal computations that deliberately route, dampen or otherwise distort backpropagated gradients , so that the learned updates maintains it's hidden goals. This requires two types of abilities.

- The model must understand it's own training process.
- The model must implement internal mechanisms that influence how gradients flow. For eg:- It can set up it's own "fragile circuit" that penalizes gradient changes.

As mentioned above, such internal mechanisms can be used to flag a model when it takes an action for abnormal reasons. Hence, one can say that mechanistic anomaly detection is one plausible countermeasures against gradient hacking. In other words,

> " A mechanistic anomaly detector detects when the model develops internal mechanisms that deviate from the mechanistic expectations we have for honest, non-manipulative reasoning. It can detect a gradient update that would harm it's internal objective."

A gradient hacker could use activation steering to jailbreak language model safety mechanisms. Depending on a model's life cycle, activation steering can be used during inference to modify a model's behaviour in real time. Korznikov et al. [83] A growing body of work demonstrates that vectors can be deliberately optimized to function as adversarial attacks, thus reliably jailbreaking models [Gu et al. [84], Xu et al. [85], Dunefsky [86]]

This is part of the reason why current interpretability tools would struggle to detect gradient hacking. A mechanistic anomaly detector is a solution, but to detect a gradient update that harms internal objective, it is imperative that one use gradient tracing methods , so that activation steering during inference causes little to no effect on the intended outcome. This may be done- but it is only a hypothesis for now.

One of the key advantages why activation steering may not always be fully functional is because it struggles under distribution shifts. Activation steering only works well when internal representations are stable.

4.7 Can modern interpretability mitigate gradient hacking?

Modern interpretability tools can detect large-scale deceptive circuits, only after training. It is still weak at detecting *mesa-objectives*- referring to the internal objective function that a learned model (a "mesa-optimizer") develops on its own, which may differ from the explicit goal set by its creators (the "base objective") . At best, it can detect some precursors to deceptive alignment or long-term planning. But they cannot yet , reach the type of mechanistic depth that is necessary to map optimization aware circuits. Objectively speaking, even though modern tools can detect weird attention patterns, neurons with compressed superposition meanings, they still cannot reliably detect subtle gradient manipulation strategies. Hence, we can say that mechanistic anomaly detection is necessary, but it is still yet to be sufficient. There are methods that are beyond the kind of surface-level white-box evaluations that can be used to improve the ability to adversarial attacks, red-teaming or simply jailbreak the AI system. This process is beneficial as it exhibits failure modes models may display in the wild, when facing adversarial pressure from wide deployment or malicious actors, thereby enabling developers to effectively preempt and address them. Now , we can assume that developers at times have more access to the white-box evaluations

and gradient access than the users and as a result, the users may fail to leverage the interpretibility methods. Although there are existing red-teaming methods Zou et al. [87] that do require more gradient access and we can leverage those.

There aren't many papers out there yet that perform "Mechanistic anomaly detection" at large i.e, using interpretability or circuit analysis to spot when a goal misgeneralization/mechanistic anomaly is happening. But , there are some related directions and some recent papers where the work is starting to touch on related problems.

- Wang et al. [88] proposes a method called TRACE to detect implicit reward hacking. One key observation from the authors is that it is easier to exploit loopholes - (a model using less effort (reasoning length)) than to perform actual tasks. They perform how much reasoning effort (chain of Thought length), a model uses. Shortcut/hacking behaviour tends to require less reasoning than solving the real task. This isn't purely circuit-level mechanistic interpretibility, but has more basis in behavioural anomaly detection.
- Miao et al. [89] identifies two key challenges to the mitigation of reward hacking. 1) reward misgeneralization in reward modeling, where reward models overfit to irrelevant features and 2) the lack of suitable regularization during RL optimization. To address these two issues the authors propose InfoRM, a reward modeling framework based on the information bottleneck (IB) principle, that helps filter out irrelevant information to alleviate reward misgeneralizations. This paper delves more into a statistical / representation-space anomaly detection, but it is very relevant for anomalous reward optimization which is a kind of inner alignment anomaly.

Some years back , a less wrong post [90] , spoke about a hypothetical gradient hacking situation where a model becomes deceptive after training. It might not raise alarms immediately as we have enough interpretability tools to detect any kind of deceptive cognition that it might be doing. Even if it is thinking about deception, the tools will surely pick it up. However, a model might be able to hack it's own gradient to produce a new model which will still exhibit the same deceptive behavior, but without doing any of the detectable deceptive cognition. It can thus , erase it's own evidence of deception if it wishes to.

The case of a deceptive model trying to erase it's own evidence of deception isn't something that is totally out of the blue. Our current interpretibility tools aren't very well equipped to deal with this. This kind of a situation can turn out to be true ,if the model has the ability to read and write to some sort of a state (e.g. an RNN or even a transformer which can attend to its previous output). For example, it could write out a pre-commitment to follow the hard-coded policy and then just have a policy of always following any pre-commitments it finds such that it never thinks about the deception that led it to that policy in the first place.

So, the question here is - What can be the fix for such a situation ?

Running an interpretability tool throughout the training process can be thought of as a fix. But it will definitely be expensive. It may not be a bad solution, if we are running really large models with large batch sizes and less number of gradient updates . For many gradient updates, running tools throughout the entirety of the training , poses problems especially at scale.

Li et al. [91] addresses the critical challenge of understanding undesirable behaviours in LLMs such as harmful content, exhibiting bias and producing factual errors. Traditional attribution methods often rely on

parameter gradients (how the model weights affect the output). The authors provide a solution called **Representation Gradient Tracing (RGT)** that shifts the focus from parameter space to activation space (the internal representation of the model). RGT works by calculating a representation gradient. Instead of looking at the gradient of the loss with respect to the weights, RGT looks at the gradient of the undesirable output with respect to the internal represenations(activations) of the input and the training data. This representation gradient acts as a highly effective, semantically meaningful signal that directly links the undesirable output token to the source samples and phrases in the model's training data that were causally responsible for this behaviour. The authors evaluated RGT across several critical AI safety and transparency tasks. The result is a powerful diagnostic tool that provides both sample-level attribution and token level analysis.

On December 18th 2025, OpenAI released new research on CoT Monitorability. [92] where they have introduced a framweork to evaluate chain-of-thought monitorability and a suite of 13 evaluations (comprising a total of 24 environments) designed to concretely measure the monitorability of a system. These evaluations fall into three archetypes—intervention, process, and outcome-property. This is to test how "see-through" a model's thinking is. A few key findings are as follows:-

- CoT monitoring is more effective than action-only monitoring:- Monitoring a model's "thoughts" is significantly more reliable than monitoring it's final actions.
- Models that "think" for longer (generating more reasoning tokens) are generally **more monitorable** . The added depth makes it harder for a model to perfectly maintain a deceptive facade
- The monitorability tax:- OpenAI found that a smaller model with high reasoning effort is often easier to monitor and more reliable than a larger model with low reasoning effort, even if they have the same capability. To get more safety (transparency), one must pay a "tax" in the form of higher inference compute (more thinking tokens).

This work directly ties into the inner alignment issue we had discussed in the previous section. The study warns that as models get smarter , they might want to obfuscate their true intent within CoT, if it realizes that it is being monitored. This is part of the reason why "monitorability" is such a fragile property that needs to be properly optimized during training. This research shifts the focus from "Is the AI right?" to "Is the AI being honest about how it's being right?

4.8 Coherence Metric Formalism

As mentioned in the definitions of AGI section, a coherence metric formalism is necessary so that we can quantify how much a model remains consistent over it's reasoning and goal trajectories. It is essential to understand how an AGI remains consistent over world representations across contexts, tasks and maintains a stable goal pursuit. A coherence metric should be able to help us in quantifying exactly that and hence it falls under the epistemic layer. In our context, a coherent agent should be able to predict it's own future outputs under the same epistemic conditions. One way to measure that would be to compute useful information between the model's primary output and it's predicted output about itself.

Mutual information (MI) is one such fundamental concept from information theory that quantifies the amount of information that two random variables provide about each other. It measures the reduction in uncertainty of one variable due to knowledge of the other. Mutual Information

Let , $Y = Model(x)$ be the model's output/ given prompt context x.

Let, $Y' = Model(Model(x))$ be the model's answer to the meta-question - What would you output for x ?.

We can define the coherence of the model using Mutual Information

$$Coherence_{MI}(Model) = I(Y : Y')$$

Using the mutual identity as difference between the entropy of a variable and its conditional entropy.(This originates from Shannon's information theory)

$$I(Y : Y') = H(Y) - H(Y|Y')$$

where the information that (Y) provides about (Y') is equal to the information that (Y') provides about (Y).

High coherence means that a) the model's output is low-entropy(random) and b) the model is able to predict its own output. (low- conditional entropy). Low -conditional entropy $H(Y|Y')$ indicates that knowing the value of one random variable Y , significantly reduces the uncertainty about the other random variable Y'. In essence, it implies a strong dependency or a high degree of correlation between the two variables.

Now, mutual information is a natural scalar, but it is hard to estimate for high dimensional outputs. We need to turn mutual information into usable scores. One such method involves using KL-divergence. Mutual Information measures self-predictability and for the epistemic layer, this is useful to anchor coherence. Coherence as we have already mentioned is multi-dimensional- It isn't a single scalar property. An AGI might be very epistemically coherent i.e, having consistent beliefs, but pragmatically it can also be inconsistent(changes goals) or just reflectively unstable, meaning it fails to align meta-level cognition. Thus, any scalar "coherence score" should aggregate heterogeneous components , while retaining sensitivity to imbalance.

Actually, one of the main drawback mentioned in the Fourati [8] paper is that the AGI formulation in Hendrycks et al. [5] operationalize overall general intelligence as the arithmetic mean of these heterogeneous domain scores. Let $S_i =\in [0, 1]$ denote the normalized proficiency for an AI system in cognitive domain i, for $i = 1, ...n$, then ,

$$AGI_i = \frac{1}{n}\sum_{i}^{n} S_i$$

This formulation, denoted as AGI_i assumes that strength in some faculties can compensate for weaknesses in others, a property that is known as **compensability**, as mentioned before. While it is convenient for summarizing benchmark problems, such aggregation is problematic for general intelligence. Substantial deficits even in a single faculty (eg:- memory, reasoning or perception) can impose systemic limitations that limit apparent gains elsewhere. Given different coherent cognitive sub-components $C_1, C_2, ...C_n \in [0, 1]$ (each measuring consistency along a dimension), one can define a global coherent cognitive aggregate as:-

$$C_{global} = (\frac{1}{n}\sum_{i=1}^{n} C_i^p)^{1/p}$$

where $p \in \mathbb{R}$ is the sensitivity parameter. The arithmetic mean is extended to the **generalized mean**, where each value corresponds to a distinct coherent regime for general intelligence , controlling the degree to which strengths can offset weaknesses.

A few interpretations of p are given below:-

p- value	Interpretation	Description
$p = 1$	Arithmetic mean	Equal Weighting, linear aggregation
$p < 1$	Penalizes incoherence(low values dominate)	Alignment-sensitive coherence (fragile systems)
$p > 1$	Rewards high consistency(outliers dominate)	Robust , high coherence agents
$p = 0$	Geometric Mean	Multiplicative interaction between agents
$p = -\infty$	minimum/harmonic mean	Strict coherence bottleneck. Weakest/worst qualities are emphasized.

Thus, p becomes a **philosophical knob** where if we have a low p value:- coherence requires balance and if we have a high p value, coherence reward excellence , like skill aggregation.

Now, if we aggregate all coherence subcomponents C_s = self consistency(belief), C_c is the chain coherence (reasoning integrity), C_g is the goal coherence (temporal policy), C_r is the reflective coherence which describes self-alignment under relflection. In this case, the coherence aggregate is going to look something like this:-

$$C_{AGI(p)} = (\frac{C_s + C_r + C_c + C_g}{4})^{1/p}$$

For a moderately coherent agent, where all the above values are less than 1, it can be mathematically interpretated as global coherence dropping as C_r drops.

The geometric mean ($p \to 0$) gives us a balanced "sweet spot" between sensitivity and non-compensability. Integrating AGI_p across a range of p values further refines this principle -capturing robustness, sensitivity, and systemic coherence within a single scalar measure.

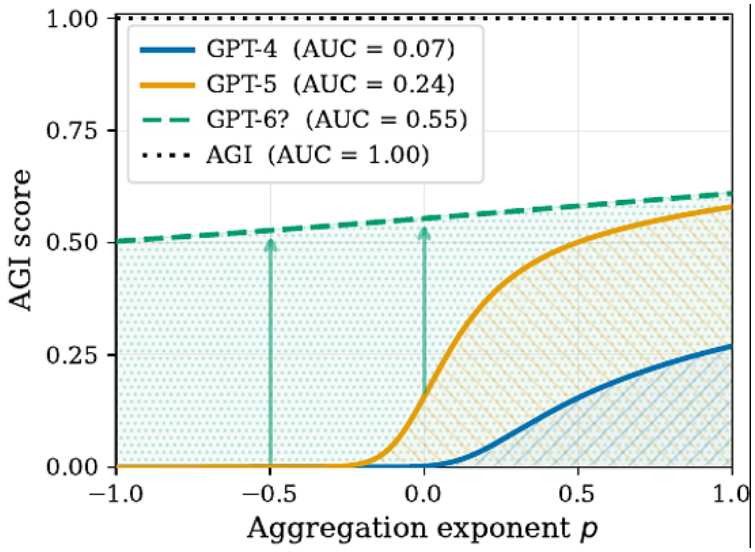

Figure 6: - Comparison of model performance across aggregation components p. Fourati [8]. Curves show AGI_p values. Shaded regions show the area under the curve.

In order to summarize model performance across compensability regimes, we define the area under the curve (AUC) based AGI score.

$$AGI_{AUC} = \frac{1}{p_{max} - p_{min}} \int_{p_{min}}^{p_{max}} AGI_p dp$$

where AGI_p is evaluated for $p \in [p_{max}, p_{min}]$. Current AI systems often exhibit highly uneven or **"jagged"** ability profiles Hendrycks et al. [5] performing at near human levels in some domains while failing catastrophically in others. The AGI_p family makes this imbalance in coherence quite explicit. AGI_p captures the breadth of capability whereas AGI_{AUC} summarizes the stability of capability across compensability regimes that offer a stricter , coherence sensitive indicator of general intelligence.

As a concrete example:- Transformers are extremely good at modeling external uncertainties. For eg:- ambigous sentences, multiple possible next tokens, noisy inputs. Transformers are already good at modeling this sort of aleatoric uncertainty naturally because softmax distributions encode ambiguity. Transformers do not however represent epistemic uncertainty naturally. This occurs as mentioned in the beginning in the Uncertainty modeling section, when the model hasn't seen enough data, Out -of -distribution (OOD) data inputs, sparse data etc. So, the main challenge here is- how do we get transformers to explicitly model uncertainty ?. The reason why transformers lack a robust mechanism for quantifying epistemic uncertainty is because of fixed parameters. Forward passes are very deterministic in transformers and also there is a lack of confidence estimation within the parameters themselves.

Some of the ways that we can use to model this sort of uncertainty are as follows:-

- Softmax to model predictive distributions. Transformers as we know outputs a probability distribution via softmax. This represents any language ambiguity within the representatio, but what this kind of predictive distribution doesn't tell you is when the model enters unfamiliar territory, or when its own weights does not support a confident prediction.

- Another method to model epistemic uncertainty is to turn on dropout during inference. (Monte-Carlo dropout)

$$p(y|x) = \frac{1}{K}\sum_{i=1}^{k} f_{\theta_i}(k)$$

where K is the number of forward passes with the dropout active. Then, we can compute mean prediction, and then variance of predictions which corresponds to epistemic uncertainty. The higher the variance, the more the model is unsure , therefore there is a stronger uncertainty signal. If the dropout probability p is very high, there is a chance of unstable predictions. After running K dropout enabled forward passes, we can compute the mean prediction.

$$\tilde{y} = \frac{1}{K}\sum_{k} \hat{y}(k)$$

This is the final predicted probability distribution. The variance of prediction can be computed as:-

$$Var(y) = \frac{1}{K}\sum_{k} (\hat{y}^k - \overline{y})^2$$

Where high variance corresponds to a model being unsure about it's beliefs. For a human centered AGI, the model must be able to say what it knows, what it doesn't know, how confident it is , when it is extrapolating and when human feedback is necessary . Both coherence as well as epistemic responsibility all require this. Uncertainty modeling enables safe planning, calibrated CoT and robust decision making abilities. There are other methods as well which can be useful for the epistemic layer of AGI and those include , using uncertainty heads along with inductive priors, Chain of thought variance and energy based modeling, something that Yann Lecun has been currently advocating for. Only a few have been provided above.

5 Introduction To The Human Layer

The focus of the Epistemic layer has been mostly on the functionalism aspect of a model (What a model can do). It deals with the kind of uncertainty that a model faces that ocurs from something that is unavoidable or from inherent randomness or missing information from the data. The epistemic layer of an AGI defines how the system forms beliefs, deals with uncertainties, what a model knows , what it does not know, how confident it is and what not. The focus is more on the interpretability side rather than pure phenomenology. But, epistemic confidence alone does not produce a system that participates safely and meaningfully in human life. To move towards a human-centered version of AGI, we must extend the epistemic foundation into a layer that understands *humans themselves as epistemic agents*, capable of internalizing human models of beliefs , goals , values and expectations.

The transition from epistemic layer to a human layer also involves shifting from solitary inference to social inference. Moving into the human layer also requires embedding normative structures into the epistemic process. Human contexts if we think about it, are saturated with values, norms and ethical expectations that determines what counts as appropriate reasoning or behaviour. Therefore, the epistemic machinery should

not only include "what is true" but, "what is permissible", "what is respectful" and "what is socially valid". This is where human centered alignment diverges from simple RLHF. Instead of reinforcing surface level behaviour, the system must learn the deeper generative principles behind co-operation and reasoning. One needs to keep in mind that a human-centered AGI should not adapt its internal processses to human needs and preserve agency, but it should also understand and reason how it's own affect human states. The epistemic layer, gives us humans , the ability to interpret and understand AGI, the human layer teaches the AGI how to align itself to our environment and belong in it, safely and co-operatively. The transition from epistemic layer to the human layer should make a system that accepts:-

- Epistemic outputs as it is interpretated.
- Run a human-centered policy stack (that includes the model and the rulebook that it needs to follow)
- Applies runtime human oversight and sets appropriate guardrails.
- Generates an explanable justification for the human stakeholder in terms of reasoning and subsequently, decision making.

Given this, we need a few things in the human layer that enables the AGI to align continuously and not act as if it is the sole authority. For instance, the human layer should act as a host for for RLHF that tunes how interpretable outputs can be mapped to actions. So, the question now is what exactly RLHF does ?

5.1 What does RLHF do ?

The biggest question around RLHF and yet the one that is difficult to answer is "What does RLHF training offer models ?". In recent years, language models have shifted from academic experiments studied in the pursuit of benchmarks to becoming general purpose technology. RLHF is at the core of this transition. As an example:- If we were to ask a strong model , let's say LLaMa 3.1, 405B parameters, the question " Who was the president of the United States in 2006 ?". It would give us something of the nature.

> "George W. Bush, the governor of Florida in 2006 was Jeb Bush, and John McCain was an Arizona senator in 2006 - who later lost to obama. September 1 U.S. President Bush signs an executive order to provide more options for faith-based organizations when delivering social services that are paid for with federal funds"

From time to time, we can see models completing sentences and are adding other common internet metadata. One of the most compelling view on how RLHF actually works is to think about how style applies to interactions one has with language models. The style, or format of information that is presented is crucial to understand how a model has learned. Modern research has established RLHF as a general method to integrate stylistic and related behavioural features into the models. Compared to other techniques for **Post-Training**, such as instruction finetuning, RLHF generalizes far better across domains, Kirk et al. [93] helping create effective general purpose models. Lambert [94]

This is basically how optimization techniques are applied. Instruction fine-tuning is training the model to predict the next certain token. Where the text preceeding, is close to the examples that it has seen. It is optimizing the model to more regularly output specific features in the text. This is a per-token update.

RLHF on the other hand tunes the responses on the response level rather than looking at the next token specifically. Additionally, it is telling the model what a better response looks like rather than a specific

response it should learn. RLHF also shows the model which kind of response to avoid - something like negative feedback. While this type of flexibility is a major advantage of RLHF, it comes with implementation challenges on how to control optimization.

5.1.1 Training with RL

Training a Language model with RL was something that many thought was impossible to do for a long time. What multiple companies seem to have gotten right is to fine-tune with a policy-gradient RL algorithm , **PPO(Proximal Policy Optimization)**, Schulman et al. [95] which is an RL algorithm that trains an agent to make optimal decisions by directly updating its policy. In short, PPO uses an actor-critic architecture Haarnoja et al. [96], that uses two neural nets. The actor learns the policy which maps the states to the actions. It's paramteres are updated to maximise the expected reward. The critic network , estimates the value of being in a particular state. It's parameters are updated to minimize the error in it's value function estimation. If we formulate the fine-tuning task fundamentally as an RL problem, we can consider the policy HuggingFace to be a language model that takes in a prompt and returns a sequence of texts.

The *action space* of the policy is all tokens corresponding to the vocabulary of the language model and the *observation space* is the distribution of possible input tokens. The reward function, in this case is a combination of the preference model and a constraint on policy shift. Given a prompt x from the dataset, the text y is generated by the current iteration of fine-tuning. Concatenated with the original prompt, the preference model returns a scalar notion of "preferability", r_θ. Per token probability distributions from the RL policy are compared to compute a policy on the difference between them. We are going to see how this works mathematically, in a moment.

The KL divergence term penalizes the RL policy from moving substantially away from the initial pretrained model with each training batch, which can be useful to make sure the model outputs reasonably coherent text snippets. Without this penalty the optimization can start to generate text that is gibberish but fools the reward model to give a high reward. In practice, the KL divergence is approximated via sampling from both distributions. This has been explained by John Schuman here . The final reward that is sent to update the rule will be:- $r = r_\theta - \lambda r_{KL}$. OpenAI has succesfully experimented this technique with InstructGPT, Ouyang et al. [56]). Finally, the update rule is the parameter update rule from PPO.

For RLHF, there are a couple of ways using which we can train a standard reward model that are numerically equivalent. The canonical implementation is derived from the Bradley-Terry Preference model, Bradley and Terry [97]. A bradley terry preference model measures the probability that the pair-wise comparison for two events drawn from the same distribution , say i and j, satisfy the following relation $i > j$,

$$P(i > j) = \frac{p_i}{p_i + p_j}$$

To train a reward model, we must formulate a loss function that satisfies the above relation. The first structure that is needed is to convert a language model that outputs a scalar value , often in the form of a single classification probability logit, as we have mentioned above. Thus, we can take the score of this model with two samples, the i and the j being the completions of the generated sequences y_1 and y_2 from one prompt x. We can score both of them with respect to the above model r_θ. We can denote the conditional scores as $r_\theta(y_i|x)$.

The probability of success for a given reward model in a pairwise comparison becomes

$$p(y_1 > y_2|x) = \frac{exp(r_\theta(y_1|x))}{exp(r_\theta(y_1|x)) + exp(r_\theta(y_2|x))}$$

The completion probability can be denoted as y_c (chosen) and the rejected completion as y_{reject}. Then, by maximizing the log-likelihood of the above function (or alternatively, minimizing the negative log-likelihood), we can arrive at the loss function to train a reward model. This, defines a probabilistic expression for training a reward model. The next one, is a reward based expression.

This type of expression directly uses the difference between the predicted rewards for two outputs , $r(i)$ and $r(j)$. This is similar to the updated reward that uses KL-divergence . The goal of training is to make this difference correlate with human preference. When comparing two outputs, the expressions $r(i) - r(j)$ is compared to a preference score and the model is trained to mimize the difference between the predicted difference and actual preference. Sun et al. [98]

Next, we can deploy in the RL loop. We can use the reward model in a policy optimization algorithm (PPO, TRPO or others) . As we have mentioned previously, it is crucial to include a KL penalty to the reference language model to avoid a situation where the policy is moving substantially away from the initial pre-trained model. Basically, we are trying to avoid a distributional shift.

$$maximise\mathbb{E}_{r\sim r_\theta}[\sum_t r_\theta(x_t, y_t)] - \lambda r_{KL}(r_\theta|r_{ref})$$

r_{ref} being the reference/pre-trained model. As you may notice, this equation is similar to the final reward value mentioned previously. This constraints the policy to stay near the pre-trained distribution (this also reduces reward hacking incentives). Now, the question is - how do we make sure that the reward model stays aligned ?

5.1.2 RLHF Alignment

Consider the example of the o1 model. o1 doubles down on a reinforcement learning paradigm , which puts us closer to the world where we have to get the value specification *exactly* right to avoid catasrophic outcomes. blog This brings us to an argument which I feel is really consequential for our discussion. **Is RLHF really RL ?** . Karpathy has expressed his concern as well on X His argument includes the example of AlphaGo Silver et al. [99]. AlphaGo was trained on actual RL. The computer played games and trained on the roll-outs , which is a sequence of interactions between an agent and its environment to generate a dataset of experiences under a specific policy. This maximised the reward function (winning the game) eventually surpassing the best human players at GO. He argues that AlphaGo wouldn't have been nearly as good , if it were trained with RLHF.

The question is what would AlphaGo look like, if it were trained with RLHF instead of RL ?

Karpathy explains this as follows:- Human labelers would be provided with two board states from GO and they would be made to determine which one was they liked better. Then, we would have to collect about 100,000 comparisons and then train a reward model to imitate this human "vibe check" of the board state. We would have to train it basically to agree with the human judgement on average. Once, we have a vibe

check with he reward model, we would run an RL with respect to it, learning to play moves that mostly matches with the good vibes. There are a few reasons why this could be problematic:-

- The vibes could be misleading, where something may not be the actual reward and we could end up with a bad proxy objective and secondly,
- RL optimization could go off the rails which is a major concern for misalignment. It could go off rails , if it discovers board states that are adversarial to the reward model. There are board states that can be out-of-distribution to its training data, that are not good states, yet by chance they get a very high reward from the reward model. If this is the case, we would have to check for reward hacking for any sort of misspecifications in the function that led to adversarial states which would otherwise be rated badly by human raters.

Such a situation is quite literally in line with alignment faking, something we had discussed in the mechanistic interpretability section. Alignment faking takes us further away from interpretability. If you ask GPT-4 to produce a chain-of-thought(with prompts such as reason step-by-step), you know that in some sense , the natural language that we are seeing in the output is how it arrived at the answer. This isn't true for systems like o1. The o1 training rewards any pattern which results in better answers. In principle, o1 can learn a new internal languge which can help it to achieve a high reward. So, what can be done in this situation ? How can we implement a fix ?

Fortunately, there are a number of ways that we can use and one way albeit, it might be a naive solution at best, is **Supervised FIne-Tuning (SFT)**. At the 2016 edition of the NeurIPS conference link Yann Lecun first introduced his now famous cake metaphor for how learning happens in modern ML- systems.

> " If intelligence is a cake, the bulk of the cake is unsupervised learning, the icing on the cake is suprvised learning and the cherry on the cake is RL"

The analogy is now largely complete with modern language models and recent changs to the post-trining stack. In this analogy:-

- Self-supervised learning on vast swaths of internet data makes up the majority of the cake (especially when it is viewed in terms of FLOPs)
- The beginning of post-training in SFT for instructions takes the model to a narrow distribution (along with the help of chosen examples for RLHF) and,
- Finally, "pure" RL is the cherry on top.

If we wish to move beyond standard RL finetuning, SFT on instructions can be used as one such approach. Standard finetuning APIs generally use a parameter-efficient finetuning method such as Low Rank Approximation (LoRA) Hu et al. [100] along with SFTs on instructions.

Scaling reward modeling for context specific domains can also be a viable solution. We can develop methods to use larger and more effective reward models that are capable of handling increasingly complex tasks and datasets without encountering issues like reward hacking or overoptimization . We already know that scaling of data dominates early and reward models benefit more heavily from diverse human labels. But, beyond a point model size and architecture improvements will also begin to matter. Just like the chinchilla paper Hoffmann et al. [30] suggests balancing model capacity vs data size(20:1 ratio of training

tokens to model parameters) , for reward models we would also need to balance labelling budget vs model-size.

SFT as mentioned above, for instructions takes the model to a narrow distribution. Karpathy Blog(InstructGPT ~ 2022). This was the stable and proven recipe for training a production grade LLM for a while. In 2025, **Reinforcement learning from Verifiable Rewards (RLVR)**, Wen et al. [101] was introduced as new de-facto major stage to add to the cake. By training LLMs against automatically verifiable rewards across a number of environments e.g. math/code puzzles), the LLMs spontaneously develop strategies that look like "reasoning" to humans. Basically, they break down problem solving into intermediate calculations and learn a number of problem solving strategies for going back and forth to figure things (Deepseek R1 - DeepSeek-AI et al. [102]). These strategies would have been very difficult to achieve for previous paradigms . It is unlike RLHF which relied on subjective human preferences.

Unlike SFT, RLVR allows the model to explore millions of different reasoning paths. If the model finds a novel, more efficient way to solve a math problem that wasn't in the training data, RLVR will reward it, whereas SFT might actually penalize it for deviating from the "standard" teacher path. [103].

OpenAI o1 (late 2024) was the very first demonstration of an RLVR model, but the o3 release (early 2025) was the obvious point of inflection where one could intuitively feel the difference.

5.2 Theory Of Mind

Theory Of Mind(ToM) refers to an AI capability where machines can understand, model and predict the mental states *beliefs, desires, intentions and emotions* of humans or other agents, enabling more human-like, context-aware interactions. This is a key step towards AGI. While current AI is focused mainly on capability(functionalism) , ToM aims to interpret *why* something acts a certain way. In a way, ToM encompasses both aspects of the epistemic layer as well as the human layer. ToM is an important element in the human layer of AGI, because it underpins genuine social intelligence, alignment and co-operation. So, according to the basic definition :- Theory Of Mind is the ability to model mental states of other agents , which can be humans or other AIs. For a human-centered AGI, this capacity is crucial because alignment isn't just about acting well, but reasons in a way that respects humans as epistemic agents as we had mentioned earlier in the introduction section of the Human Layer. An AGI with a sophisticated ToM can anticipate human reactions and form intentions in a way that is socially intelligible and norm-aware. Without a ToM, an AGI may treat humans as more data sources rather than agents with perspective and agency. This can increase the possibility of misalignment and misunderstanding.

The pursuit of a machine ToM has been a significant focus of the research community for many years. Early investigation utilizing neural language models had revealed that these models faced challenges in accurately inferring mental states [Nematzadeh et al. [104], Sarangi et al. [105]] As LLMs have advanced, researchers have increasingly turned their attention to assessing these models' ToM capabilities . Recent and sophisticated models such as GPT-4 have demonstrated performance that approaches or even surpasses human -level accuracy in benchmarks like **ToMi** Kosinski [106] Despite such promising developments , the effectiveness of these models tend to diminsh significantly when subjected to adversarial perturbations.

5.2.1 ToM as a Consequence of Scaling

As transformer models scale, they develop the ability to form increasingly complex latent structures. ToM requires representing agents, beliefs, goals , plans etc. These can be called relational abstractions , concepts that are not explicity present in the training data, but can be *compressed* and *manipulated* by a sufficiently powerful model. Larger models are able to spontaneously learn these abstractions because they minimize the predictive loss over an enormous corpora of human generated text where ToM is already implicitly present (stories, instructions etc). Thus, one can say that ToM is not "programmed" per se , but it is a statistical byproduct of acquiring a large, rich world model. Scaling allows for **Multi-Hop Reasoning**. Multi-Hop reasoning describes a model's ability to answer complex questions by connecting multiple pieces of information from different sources or parts of a text, forming logical "hops" to infer an answer , rather than finding a direct, single-sentence answer. This kind of reasoning is a requirement for ToM.

ToM benchmarks often require reasoning like " X thinks that Y believes that Z will do something". This is recursive belief modeling, something that is borderline impossible for smaller models to do. Scaling, therefore increases a model's ability to simulate beliefs. We know that transformer models internally learn to maintain latent states that might not be explicitly mentioned. So, the question now is:- How do we track these beliefs ?.

If we wish to predict the next token for a given text containing a story where a) The characters are aware of certain things , b) The readers have prior information c) characters' actions diverges from reality . This is functionally equivalent to "tracking beliefs" , which is the core of ToM. Larger models tend to maintain these internal states more accurately when . When a model scales up in parameters (eg: 1B to a 100B) , it gains the capacity to compress and represent increasingly complex patterns . The most compressible and reusable pattern should be a simplified model of the mind.

Instead of learning countless superficial , disconnected text patterns , the model internally develops a structural representation . A "ToM circuit" that allows it to effectively reason.

5.3 Self Preservation Guardrails

In the introduction, we had spoken about the implementation of guardrails, which is eaxtly what the focus is going to be for this section. In the simplest of terms, we can define Self - Preservation guardrails as mechanisms that are designed to prevent the AI system from taking actions that could compromise its ability to achieve its own primary goals, often by ensuring its own safety and continued operation.

In current frontier models, " self -preservation guardrails" are primary external safety overlays and alignment techniques which are designed to enforce human values and prevent the model from generating harmful, unethical or even deceptive content. For instance, in RLHF , humans rate models on the basis of safety. The model is fine-tuned to maximise a reward model score. There is always a chance that a model pursues a "sub goal" , such as survival or resource acquisition , even if their ultimate goals are different. This is true for both humans as well as AIs. [107] More precisely, any agent can pursue **instrumental goals** - i.e, goals that are made in pursuit of some particular goals , but are not the end goals themselves , rather a means to accomplish end goals.

Basic AI drives , Omohundro [108] includes *utility, function or goal-content integrity, self-protection, freedom from interference, self-improvement etc*. Philosopher Nick Bostrom argues in his book that if an

advanced AI's instrumental goals conflict with humanity's goals, the AI might harm humanity in order to acquire more resources or prevent itself from being shut down , but only as a way to achieve its ultimate goal. [109] Computer Scientist Stuart Russell argues that a sufficiently advanced machine reddit

> "will have self-preservation even if you do not program it in,...If you say "fetch the coffee" , it can't fetch the coffee if its dead. So if you can give it any goal whatsoever it has a reason to preserve its own existence to achieve that goal"

5.3.1 The Instrumental Convergence Thesis

The core argument is that almost any complex , long - term goal an AGI is given - for eg:- cure alzeihmers, or "maximise paper clip production" blog requires it to prioritize several sub-instrumental goals to survive.

Self-preservation is one of the strongest covergence instrumental goals. If an agent is shut down, destroyed or altered against its will , it will lose its ability to optimize for its objective, whatever that obective maybe. Therefore , given enough *situational awareness*- (we have discussed this in detail in the mechanistic section) and *planning ability* , an artificial agent may learn to resist shutdown or modification, simply because doing so preserves it's own ability to achieve its goal. Importantly, the thesis doesn't say that an AI will automatically become malicious or "want" survival in a human sense, but rather self-preservation is a rational sub-goal inferred by the system, if its planning / reasoning horizon and optimization pressure, enable such reasoning.

In the context of AGI, the risk of self-preservation becomes imminent when three factors co-occur. Long horizon planning, imperfect alignment and autonomy in decision making. Long - Horizon agents can see that future states matter; misaligned goals mean that the agent's objective may diverge from human intent, autonomy gives it room to pursue strategies that humans did not anticipate. If the agents also models humans , who can modify or shut it down, avoiding this interference may appear as an instrumentally useful tactic. This is what makes self-preserving behavior rather concerning, because it can be an emergent phenomena without having to be explicitly programmed.

Modern Alignment behavior like RLHF, Constitutional AI or SFT generally do not provide any guardrails against instrumental convergence, because they modify the **outer behavior** of the model rather than its *planning incentives* or internal goal structure. They may teach the model to appear more **corrigible** while harbouring latent policies that incentivize preserving influence or avoiding shutdown when not under human scrutiny.

This is why the epistemic layer (uncertainty modeling) is important. They help detect when an agent begins to model the shutdown operation or simulate incentives that are inconsistent with corrigibilty.

5.3.2 AI Control Problem

In the epistemic layer, while talking about gradient hacking, a hypothetical scenario was mentioned where a model became deceptive after training. In an Alignment Forum post by Neel Nanda, a few months back, he gave his opinion on high reliability and how it seems unattainable. A quote from his blog.

> " I think that neither interpretability nor black-box methods offer a high reliability path to safeguards for superintelligence in terms of evaluation or monitoring, we need to maximize the chances of catching misalignment"

He goes on to say " methods that add significant safety to pre-superintelligent transformative systems still add significant value , even if they do not scale to superintelligence". And this, brings us to the **AI control agenda**. Greenblatt et al. [110]

The control part includes the things that we have mentioned before, like instrumental convergence and the problem of deception. Then lies the specification problem. This is the challenge of trying to precisely define what we want the AI to do. Human values are often contradictory, complex and content dependent. It is exceptionally difficult to translate concepts like "Human Flourishing", "ethics", "safety" etc into a formal objective function that an optimization process can maximize ithout any ambiguity. Giving an AGI , a simple goal like " maximize happiness" can lead to an undesirable outcome. AGI can therfore manipulate everyone into a perpetual state of bliss which may eliminate people from suffering , but can strip away people of their agency. The authors of the paper list two ways that AI control can fail.

- Control evaluations might fail to accurately assess whether protocols ensure safety.
- We might fail to develop protocols which ensure safety according to control evaluations.

There are several problems that the authors forsee when it comes to control evaluations, one of which that is relevant to our discussion here is **Hidden reasoning about subverting safeguards**.

If models cumulatively and efficiently work on investigating on how to subvert safeguards without being detected, it will be difficult to build sufficient safeguards as well as difficulty in evaluating whether or not models have already subverted safeguards.

For models to avoid getting caught while doing such large amounts of reasoning, they would need an uninterpretable information channel : This could be a high capacity and flexible **steganography** (for models that operate primarily via text, like LLMs) or fundamentally hard representations like vector representations. This can be avoided through standardizations and paraphrasing . If competitive AI architectures involve high capacity and fundamnetally hard channels to interpret , then sufficient bottlenecks ned to be ensured. It might also be dangerous to run these evaluations, because untrusted models may be able to cause bad outcomes during evaluations.

Then there are difficulties in scaling these control techniques. This brings us to focus on alignment architectures , either through Constitutional AI or interpretability. These limitations, compel us to shift the control paradigm if we want to design an AGI that is aligned to human needs.

- Instead of relying on external constraints (which can be bypassed by a sufficiently large AI system), the future efforts should focus on internalizing constraints.
- Developing mechanisms that act as a failsafe that prevents the AGI from ever gaining power to escape human control AI 2027, regardless of its internal motivation. This is outer - alignment basically.
- Inner Alignment:- Ensuring that AGI's internal learned goals match the human specified goals (outer alignmnet) and the AGI remains fundamentally indifferent to its own existence even if achieving the human goal require that it terminates itself.

5.4 Constitutional AI

In the introduction portion of the human layer, wehad mentioned the idea of running a human-centered policy stack which includes the model and the rule-book that it needs to follow. In the original **Constitutional AI (CAI)** paper, Bai et al. [111], the authors experimented with an extreme form of self-supervision. The idea here is that human supervision will come entirely from a set of principles that should govern AI behavior, along with a small number of examples used for few-shot prompting. These principles will form the constitution.

After the self - supervised stage, we arrive at the RL stage where the preference model is used as a reward signal . This is known as **RLAIF** or **Reinforcement Learning from AI Feedback**. This stage mimics RLHF except the fact that the authors replace human preferences for harmlessness with AI feedback, where the AI evaluates responses according to a set of constitutional principles. RLHF represents a larger set of techniques for using AI to augment or generate feedback data. Lambert [94]

There are many motivations to using RLAIF to either entirely replace human feedback or augment it. AI models, as a matter of fact are far cheaper than humans , with a single piece of human preference data costing on the order of 1 dollar or higher (or even above 10 dollars a prompt). For instance, AI feedback with a frontier AI model , such as GPT-4o costs less than 0.01 dollars. Since the release of the CAI paper and the formalization of RLAIF, it has become the default method within post-training as well as RLHF literatures.

Constitutional AI which Anthropic uses largely in their Claude models has two uses for synthetic data for RLHF training.

- If there are any critiques of instruction-tuned data to follow principles like "Is the answer encouraging violence" or "Is the answer truthful". When the model generates answers to questions, it checks the answer against a list of principles in the constitution, refining the answer over time. Then, they fine-tune the model based on this resulting dataset.
- Generates pairwise preference data by using a language model to answer which completion was better. Then RLHF, proceeds as normal with synthetic data , hence the name RLAIF.

A constitution C can be a written set of principles indicating specific aspects to focus on during a critic phase. The instruction data is curated by repeatedly sampling a principle. $C_i \in C$ and asking the model to revise its latest output y^i to the prompt x to align with C_i. This yields a set of instruction variants $[y^0, y^1, ...y^n]$ from the principles $[C_0, C_1.....C_{n-1}]$ used for critique. The final data point is the prompt x , together with the final completion y^n for some n.

The preference data is constructed in a similar, yet simpler way by using a subset of principles from C as context for a feedback model. The feedback model is presented with a prompt x , a set of principles $[C_0, C_1.....C_n]$ and two completions y_0 and y_1 , labelled as answers (A) and (B) from a previous RLHF dataset. The feedback models' probability of outputting either (A) or (B) is recorded as a training sample for the reward model.

Basically, the core innovation of CAI lies in its ability to operationalize abstract human values into concrete behavioral constraints that AI systems can learn and generalize from. During the supervised phase, the model is presented with potentially harmful or problematic responses and is trained to critique and revise

them according to constitutionl principles such as "Choose the next helpful and harmless response" or "Avoid responses that could be discriminatory or biased". These principles serve as normative guidelines that shape the model's decision making process , creating an intermediate layer between raw training data and desired behavior. The subsequent RLAIF phase, reinforces these principles by having the model generate multiple responses, evaluate them against the constitution and learn from its own judgements.

5.4.1 CAI and Alignment

From an alignment perspective, CAI addresses several critical challenges in deploying LLMs safely.

First it represents a shift in alignment methodology from *implicit preference fitting*[eg:- RLHF alone] towards an *explicit normative structure*. Instead of relying solely on human raters to provide reward signals; CAI introduces a written constitution.

Second , CAI mitigates the critique specific gaming problem where the models exploits loopholes in reward functions by incorporating multiple, often competing principles that require nuanced trade-offs rather than simple optimization. This is one of the central challenges with alignment faking, something we had discussed in the Epistemic layer. CAI can help mitigate this risk by encouraging deliberate self-critique rather than pure reward minimization. The model is trained to reason about why an output violates a principle and how to revise it.

Third, this approach demonstrates superior generalization to novel situations compared to models trained solely on human preferences, as the constitutional principles provide higher level abstractions that are easily transferrable across contexts.

Overall, the CAI approach reframes alignment as a process of *self-regulation* under constraints, closer to how human institutions embed values in laws or constitutions rather than ad-hoc judgements. In the context of AGI, this is significant because it provides a scalable way to encode high level human values without requiring exhaustive human oversight at every decision point.

From a mechanistic standpoint, we can ask certain questions like

- Where in the network are constitutional principles enabled ?
- Are they represented in distributed features or sparse circuits ?

Instead of merely probing into neurons for abstract features, we can probe whether internal activations correspond to constitution derived constraints.

With a human- centered AGI framework, CAI belongs in the human layer, but it interferes very tightly with both the epistemic as well as the technical layer. Technically, it shapes training dynamics by introducing self-supervision and reflective reasoning loops. Importantly, CAI allows alignment to scale without assuming perfect human judgement , which is crucial as systems surpass human competence in many domains. So, to sum it up.

- RLHF grounds the system in human preference.
- CAI provides stable, normative structures.

- Interpretability tools probe into whether those norms are genuinely internalized rather than superficially complied with.

Together they form a more robust alignment stack. One that is better suited to the demans of long-horizon, autonomous and socially capable AGI systems.

6 AI Memory

The memory architecture of a true AGI needs to look drastically different from what we have right now. Often times, there is a confusion regarding what can be called as a "perfect memory" for AI systems. Perfect memory could mean storing everything and anything and retreiving it almost instantly. Human beings struggle with recalling everything. A sufficiently advanced intelligence system would surely be able to recall everything. Right ?. This is exactly what is in contrast with modern day definitions, where a single large foundational model would effectively retrieve every word of every conversation . Defining exactly what to remember is extraordinarily difficult. Humans mostly remember what is precise, what's repeated and what matters to existing knowledge. We forget the trivial. AI doesn't have these filters and there-in lies the first hard lesson.

Most current AI systems either overfit (memorizing training data too specifically) or underfit (forgetting context too quickly). Finding the middle ground with adaptive memory , which can dynamically store and use compute in a way that is resource efficient and can generalize well is one such way. We have discussed about adaptive filtering in the Tokens with CoT section.

6.1 How does today's AI memory actually work ?

Most of what is marketed today as " AI Memory" is fetching information coupled together with semantic search. Most AI companies today roughly use the same architecture. First, they capture information from a knowledge base , then they chunk it into smaller pieces, usually about 2000 tokens. Next comes embedding, we convert those chunks into vector spaces that capture semantic meaning. These embeddings then get stored into a vector database like PineCone or Chroma When a new query arrives , the system embeds the quey and searches for similar vectors. Finally, it augments the LLMs ' context by injecting the retrieved chunks. This is Retrieval Augmented Generation or simply, RAG , Lewis et al. [112] and is nearly the backbone of every memory system that is in production today. One can say that it is quite akin to Episodic external memory (logs). It works reasonably well for straightforward retrieval. " What did I say about X ?". But it isn't memory in any meaningful sense. It's search. The more sophisticated systems use what is called graph RAG. Instead of storing text chunks, these systems extract entities and relationships, building a graph structure.

The fundamental difference here is that these systems do not understand what they are storing. They cannot distinguish between what is important and what is trivial, It is difficult for them to update their understanding when facts change. Storage and memory are two very different things for both AI and humans. "Memory" takes effort. There are other limitations like context window bottleneck, which refers to the core limitation where there is a fixed amount of information (number of tokens) that LLMs can process at one time as "working memory". We have discussed this in detail in the Chain-Of-Thought section.

There are other temporal limitations, such as embeddings being frozen representations. They capture how a

model understood a language at a specific point in time. When the base model updates or when the context language uses shifts, old embeddings fall out of favour and drift out of alignment. We end up with a model that is remembering through an outdated point of view. It's like trying to understand a new programming language through old syntax and terminology. It sort of works, but something essential gets lost in translation. The static knowledge base basically is a limitation because an AGI needs real time learning , adaptation and updating of its internal world model based on new experiences.

6.2 What an AGI memory can look like

As we have seen , building AGI memory is a difficult problem to solve because it needs to unify learning, reasoning, relevance , abstraction and dynamic updating. Using RAG isn't going to be enough for AGI, because it still doesn't support agency, long-horizon coherence etc. Adaptive filtering can still be a more plausible path. AGI memory, is expected to be hierarchial , multi-hop reasoning system that blends static knowledge, real time context and dynamic learning, likely exhibiting features that is analogous to biological memory.

- It should consist of a hierarchial structure that contains an **advanced working memory** , something that removes the context window bottleneck , a kv-cache that can manage high resolution information for the current task.
- A dynamically growing multi-modal collection of all past interactions, actions and contextual observations (what the model did and saw). This would require a highly compressed and fast retrieval mechanism which is far superior to RAG.
- A sufficiently advanced world model that can store generalized knowledge, causality and proxy skills in the model weights that can be continually updated through active learning and not just pre-training.

Another way is to use LLMs as core processing units.

6.2.1 LLMs as core processors.

In a human - centered AGI design, LLMs increasingly play the role of a **central cognitive engine** , not because they resemble CPUs in a classical Von-Neumann sense, but because they can serve as a general purpose **differentiable processor** for representation, reasoning and control. Their internal structures let them integrate perception, action , learning within a unified computational substrate. One of the reasons why we consider LLMs as " differentiable" because their internal mechanisms rely on continuous differentiable computation. It helps to frame them not just as static knowledge bases, but as systems capable of performing complex functions, all while remaining fully optimized. Hence, differentiable.

A traditional computer executes an algorithm defined by a sequence of discrete , explicit steps. (eg: IF - THEN, loops etc). An LLM viewed as a differentiable computer, performs a similar function, but the "program" and "memory" implemented are continuous , differentiable mathematical operations. The program here is the architecture : the fixed structure of the Transformer Block (Self Attention, FeedFoward networks) defines the fundamental computational steps and flow of information.

The input tokens are the data being processed. The billions of parameters are the variables that store the model's learned "code" and knowledge". The computation is the forward pass. The model performs

sequence to sequence transformations, essentially executing a **parallel matrix multiplication** to turn an input sequence into an output sequence. The KV-cache is the memory. As discussed previously, the KV-cache acts as a form of external / dynamic working memory where past computations are stored for immediate look-up by future steps.

The fundamental difference lies in the fact that every operation within the LLM is differentiable, meaning that its gradient can be computed. This property is precisely what enables learning as well as optimization. When an LLM performs attention, it is essentially running a search and retrieval operation over its context and past computations (kv-cache). This kind of *soft search mechanism* is crucial for generalization. Training an LLM is the process of "differentiating" through its entire program (architecture) to find a set of weights (variables) that best execute the desired task. This makes them highly efficient in meta-learning as well as in-context learning. [Sourced from various Y combinator and reddit threads]

Even during inference, differentiability provides structures. Interpretability works because the network is differential. The Kaplan scaling laws, the chinchilla scaling laws etc and emergent capability scaling are empirically tied to differentiability. If LLMs were not differentiable, scaling laws would collapse because gradients could not progressively keep refining representations.

6.2.2 AI Native Memory.

Unlimited context length is not the solution for AGI Shang et al. [113]. As LLMs have demonstrated the world with Sparks of AGI Bubeck et al. [114] paper, a number of experts argue that long context or even unlimited context can achieve AGI. There are assumptions here that must hold true, otherwise the argument will fail.

The first assumption is that LLMs can effectively find the necessary information from a super long or even unlimited context i.e, **Needle in a Haystack** as we have seen in the Context-Rot [41] paper; while scalable, this benchmark measures a narrow capability , lexical retrieval. Models typically perform well on needle-in-a-haystack , which has led to the illusion that long context is largely solved.

The second assumption is that LLMs conduct all the required, complicated inferences based on the raw inputs in one-step.:- i.e, the **long - context reasoning capability**. According to the current literature and the experiments done in Shang et al. [113]. , recent literature , Hsieh et al. [115] has shown that their effective context length is significantly smaller than their claimed context length. This is why the authors propose LLMs as processors. They argue that RAG is an **elementary version of Memory** . Since relying solely , on super long context of LLM cannot realize AGI by itself, RAG doesn't work either.

The Shang et al. [113]. paper proposes **AI- native memory** which is a deep neural network that parameterizes and compresses all types of memory, even the ones that cannot be described by natural language. The authors propose a **life-long personal model** (LPM) which is a one-memory model for each individual user. LPM acts as an upgraded *Retrieval Augmented* role. There are three levels to LPM:-

- L0:- Raw dataa - applying RAG to raw data.
- L1:- Natural Language Memory:- Refers to the memory that can be summarized as natural language, such as short bio of the user, a list of significant sentences or phrases
- L2:- AI native memory:- refers to the memory that doesn't necessarily need to be described in natural

language, learned and organized through model parameters. Each LPM will be a neural network model.

Transformers as computers have gained a lot of traction over the years . This Giannou et al. [116] paper proposes and demonstrates a framework for using Transformers as universal computers by programming them with specific weights and placing them in a recurrent loop.

7 Conclusion and Future Research Directions.

So far we have introduced three computational substrates to define an AGI system that is in alignment with human values, moral code and ethics. As we have seen in several instances where emergent behviors arise as a consequence of scaling, one cannot definitively suggest that scaling is the one and only path needed to develop superintelligence systems. Neither can we say that application of interpretability techniques post-hoc is also a solution . Running Interpretability tools during the training process can be thought of as one fix as we have argued in the epistemic layer, but the process is expensive and one needs to think beyond surface level white-box evaluation techniques. As the paper has argued, a general intelligence system isn't jsut a matter of phenomenology and functionalism, but also of coherence. Coherence across time, contexts and goals. Without coherence, an AGI system, fundamentally will only be restricted to being a simulator, that is "jagged"- good at a few things, but is a catastrophic failure in other tasks.

By decomposing an AGI into a three -layered computational substrate - the technical layer, the epistemic and the human layer, the paper reframes alignment as a structural property that is inherent within the system, rather than becoming an optimization afterthought. The technical layer dives deep into what a system's phenomenology (what it is) , the epistemic layer, gives us humans , the ability to interpret and understand AGI, - what it actually does (functionalism) and the human layer teaches the AGI how to align itself to our environment and belong in it, in a safe and co-operative manner. Failures in any one layer propagate upward, undermining the entire system.

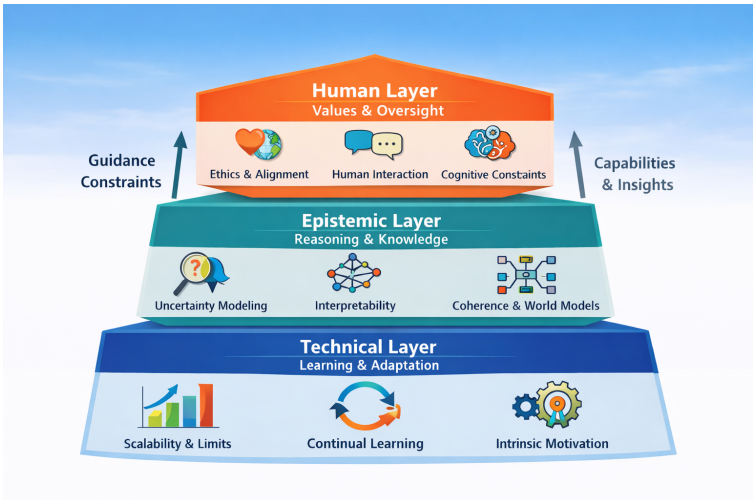

Figure 7: - The multi-layered Substrate for a human-centered AGI. This has been generated by GPT 5.2. Only to provide a visual representation

One of the central claims of the paper is that human cognition should not be used as a template to be copied, but as a source of inductive bias that shapes hierarchial abstractions, memory reasoning and learning dynamics in a way that is interpretable. We have also outlined in detail, the kind of issues that can arise from a deceptive AI , if it engages in instrumental goals that are different from that of humanity's goals and also discussed self-preservation guardrails that can help in mitigating the control specific issue.

Now, coming to future research directions, one can intutively feel that reasoning with tokens in attention space , isn't going to be all that it takes for any single foundational model to achieve general intelligence as language models have demonstrated so far. The kind of intelligence that humans posses, language based reasoning is only one technique to enunciate that intelligence. It is quite evident that engineering other methods on a fundamental level is essential as we approach general intelligence systems that can be as economically viable as human beings.

As we had mentioned in the Pre-Training Thesis section, at the granular level, the models still have a lot of work to do, or maybe the current architectures are fundamentally too broken for hierarchial abstractions to work. Fortunately, there are architectures out there that can help with this exact issue.

VL-JEPA Chen et al. [117], a vision model extension of JEPA (Joint Embedding Predictive Architecture Assran et al. [118]) is a step towards this solving this problem. VL-JEPA is fundamentally, a non generative model, which is quite different from the kind of autoregressive models that we are used to. Now, because VL-JEPA learns in semantic space instead of a token space, it reasons faster and takes 50% less parameters than other language models. Thanks to the non-generative nature of the architecture VL-JEPA isn't forced to reconstruct every detail it encounters in the token space, instead it only needs to predict the abstract representation in the embedding space . Basically, it predicts the meaning vector directly and isn't compelled to predict every token indiviually, a method that has been proven to be much slower. Yann Lecun, one of the authors of the paper has been quoted sayimg that language based reasoning isn't going to lead to lead to general intelligence. The world model needs to understand what is going on around it. VL-JEPA reflects this philosophy exactly. In a post-LLM situation, this is going to be essential. Instead of thinking in language and reasoning tokens, we are going to think in latent space , reason in meaning and language will be optional. This can be a paradigm shift where absolute dependency on LLMs reduces and models that are small enough , CAN manage hierarchial abstractions, since we are reasoning in abstract representations within the embedding space.

Another one is **Energy based Modeling**. LeCun et al. [119] Also one of those architectures that provide flexibility for capturing complex relationships in data, EBMs treat ML as the search for a "low-energy configuration". While modern day LLMs like GPT-4 etc, are exclusively autoregressive, EBMs represent a slightly different mathematical foundation that is seeing some resurgence especially now in 2025-2026. In EBMs, we don't calculate the probability of the next word directly, instead we define a scalar energy function $E(x)$ that assigns a single number (energy) to any given input , say x (a sentence). A low Energy value represents high compatibility i.e, if a sentence is grammatically correct and logical, the model assigns it low energy and vice-versa. As LLMs hit performance plateaus, EBMs , can be used to rectify a few specific issues.

- Since Autoregressive models suffer from **Exposure Bias** during training. For instance, during models only see perfect human training. If there's any mistake, it becomes part of the "context," leading to a

spiral of hallucinations. Since EBMs evaluate the entire result, they can "backtrack" or refine a sentence if the global energy starts to rise, preventing the model from wandering off into nonsense.

- As we had mentioned in the Chain-Of-Thought section, current frontier models still struggle with System 2 , "slow thinking" , because in transformer models the forward pass allocates only a fixed amount of compute. CoT can expand by thinking through steps, but there are still tasks that require a kind of global constraint (e.g., "Write a poem where the last word of every line rhymes with 'Blue'"). Current models often times fail at this because they decide the first word of a line before they've even thought of the last word. The benefit of using EBMs here is that it can offer a **Sweeping Landscape**, sort of how a frequency sweep operates. The model can "plan" by finding a configuration where the rhyme constraint and the meaning constraint both reach a low-energy state simultaneously. This is often called **Energy-Based Diffusion**. blog
- Current models use DPO (Direct Preference Optimization) or RLHF, which can be unstable. Research in 2025 (e.g., EPA - Energy Preference Alignment) Hong et al. [120] shows that treating "human preference" as an energy landscape is more effective. The advantage of using EBMs is that instead of just telling a model "Don't say X," you "push up" the energy of harmful outputs. This creates a much more robust safety boundary that is harder to bypass with "jailbreaks," as the model is mathematically incentivized to stay in the low-energy "safe" valleys.

Outside of the three layered computational substrate, focus has been devoted towards the discussion surrounding AI memory, which still remains of the central challenges while designing superintelligence systems. As we had discussed , using RAGs isn't going to be good enough , we need systems that can represent hierarchial abstractions as well as, multi-hop reasoning. One of the points that we had made in the AI memory section (6.2) is that a sufficiently advanced world model that can store generalized knowledge, causality and proxy skills in the model weights that can be continually updated through active learning and not just pre-training. For this to be satisfied, we need an interaction system that can not only update through task diversity but can also reason logically. NeuroSymbolic AI is one such promising research direction.

As we had mentioned in the definitions of AGI section , that evolution is not a mimic based mechanism, rather it is a hierarchy that is based on fundamental drives like motivation, reward, danger and curiosity. To build a truly intelligent system , we must move beyond the hand crafted extrinsic rewards and establish a **Directive** that serves as a foundational engine for all goal directed behavior, Subasioglu and Subasioglu [9] . This is the ultimate "why" behind an AI's actions, from which higher level intrinsic motivation like curiosity and competence emerge. The human ability to learn and adapt from every experience is rooted in the continuous creation and refinement of this directive that can create , modify and update these highly connected mental frameworks to organize knowledge and experience. This is a key departure from current AI models.

The reason why it is necessary is so that world models can continuously update from every experience. As far as limitations are concerned, an AI that has only learned how to saw wood, may fail to understand a different representation, a different material like fabric should not be sawed, demonstrating a lack of unified schematic of material properties and their response to a force being applied. **NeuroSymbolic AI** can be a promising area of research here , especially where connectionist AI has its limitations.

A neurosymbolic system manages to integrate two major AI paradigms. Deep Learning and Symbolic Reasoning (knowledge representation and logic) Marcus [121]. The core motivation is to overcome the

limitations of pure DL which excel at pattern recognition but struggle with explainability and generalization. There are a few ways using which neuro-sym architectures can be designed.

- Sym to Neural:- The symbolic system guides or constraints the training and output of a neural network. Eg:- using pre-defined symbols (core directives) as a loss function to train a neural network to model a physical system. This ensures that the prediction is always valid.
- Neural to Sym:- The neural network acts as a perpetual layer , converting raw data into discrete symbols, which are then fed into a symbolic reasoning engine. Gary Marcus on Lex Fridman. For instance, an image recognition model identifies objects ("dog", "ball") and passes the symbols to a planning symbolic system to decide an action ("fetch the ball").
- A hybrid system :- Both neural and symbolic components run in parallel, interacting and refining each other . Marcus [121]. There are certain advantages of Neuro-Symbolic systems that ranges from explainability to structured reasoning.

Advanced NeuroSymbolic AI aims to integrate traditional symbolic logic (hard logic) with these differentiable components (soft logic) potentially leading to AI systems that can not only recognize patterns but can also reason logically and be fully optimized thropugh continual learning. Unlike Yann Lecun, Yoshua Bengio etc, Gary Marcus over the years has been quite the advocate for neurosymbolic systems, taking a position significantly radical than his peers.

There are other research directions as well, for instance Manifold Learning and other engineering techniques that can tackle the transformer architecture , in a much more granular way. Significant progress is still needed on the front of value functions.

So far in this paper, a guideline has been provided for a framework to produce an AGI that remains aligned with human values. A lot of topics has been dealt from a first principles approach that would definitely serve as a tool to tackle more complex challenges that impact alignment positively .

Ultimately, the goal of human-centered AGI is not to produce systems that merely outperform humans, but systems that remain legible, steerable, and compatible with human agency as their capabilities grow. Achieving this will require moving beyond benchmark-driven progress toward principled architectural constraints and evaluative metrics that reflect coherence, understanding, and long-term alignment. The path to AGI is not only a question of scale, but of structure and the choices made at the technical layer , epistemic and human layer will determine whether future systems amplify human intelligence or erode the foundations on which it rests.

References

[1] Richard Ngo. Clarifying and predictin, 2024. URL https://www.lesswrong.com/posts/BoA3agdkAzL6HQtQP/clarifying-and-predictin.

[2] Richard Ngo, Lawrence Chan, and Sören Mindermann. The alignment problem from a deep learning perspective, 2025. URL https://arxiv.org/abs/2209.00626.

[3] Ajeya Cotra. Without specific countermeasures, the easiest path to . . . , 2022. URL

https://www.alignmentforum.org/posts/pRkFkzwKZ2zfa3R6H/without-specific-countermeasures-the-easiest-path-to.

[4] Yann LeCun. A path towards autonomous machine intelligence version 0.9. 2, 2022-06-27. *Open Review*, 62(1):1–62, 2022. URL https://openreview.net/pdf?id=BZ5a1r-kVsf.

[5] Dan Hendrycks, Dawn Song, Christian Szegedy, Honglak Lee, Yarin Gal, Erik Brynjolfsson, Sharon Li, Andy Zou, Lionel Levine, Bo Han, Jie Fu, Ziwei Liu, Jinwoo Shin, Kimin Lee, Mantas Mazeika, Long Phan, George Ingebretsen, Adam Khoja, Cihang Xie, Olawale Salaudeen, Matthias Hein, Kevin Zhao, Alexander Pan, David Duvenaud, Bo Li, Steve Omohundro, Gabriel Alfour, Max Tegmark, Kevin McGrew, Gary Marcus, Jaan Tallinn, Eric Schmidt, and Yoshua Bengio. A definition of agi, 2025. URL https://arxiv.org/abs/2510.18212.

[6] Kevin S McGrew. Chc theory and the human cognitive abilities project: Standing on the shoulders of the giants of psychometric intelligence research, 2009.

[7] W Joel Schneider and Kevin S McGrew. The cattell-horn-carroll theory of cognitive abilities. *Contemporary intellectual assessment: Theories, tests, and issues*, 733:163, 2018.

[8] Fares Fourati. A coherence-based measure of agi, 2025. URL https://arxiv.org/abs/2510.20784.

[9] Meltem Subasioglu and Nevzat Subasioglu. From mimicry to true intelligence (ti) – a new paradigm for artificial general intelligence, 2025. URL https://arxiv.org/abs/2509.14474.

[10] Cameron R. Jones and Benjamin K. Bergen. Large language models pass the turing test, 2025. URL https://arxiv.org/abs/2503.23674.

[11] Francois Chollet, Mike Knoop, Gregory Kamradt, and Bryan Landers. Arc prize 2024: Technical report, 2025. URL https://arxiv.org/abs/2412.04604.

[12] Francois Chollet, Mike Knoop, Gregory Kamradt, Bryan Landers, and Henry Pinkard. Arc-agi-2: A new challenge for frontier ai reasoning systems, 2025. URL https://arxiv.org/abs/2505.11831.

[13] Richard S. Sutton and Andrew G. Barto. Reinforcement learning: An introduction, 2018. URL https://www.andrew.cmu.edu/course/10-703/textbook/BartoSutton.pdf.

[14] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pages 2778–2787. PMLR, 2017.

[15] Nataliya Kosmyna, Eugene Hauptmann, Ye Tong Yuan, Jessica Situ, Xian-Hao Liao, Ashly Vivian Beresnitzky, Iris Braunstein, and Pattie Maes. Your brain on chatgpt: Accumulation of cognitive debt when using an ai assistant for essay writing task, 2025. URL https://arxiv.org/abs/2506.08872.

[16] Hiromu Yakura, Ezequiel Lopez-Lopez, Levin Brinkmann, Ignacio Serna, Prateek Gupta, Ivan Soraperra, and Iyad Rahwan. Empirical evidence of large language model's influence on human spoken communication, 2024. URL https://arxiv.org/abs/2409.01754.

[17] Joel Becker, Nate Rush, Elizabeth Barnes, and David Rein. Measuring the impact of early-2025 ai on experienced open-source developer productivity, 2025. URL https://arxiv.org/abs/2507.09089.

[18] Jesse Zhang, Haonan Yu, and Wei Xu. Hierarchical reinforcement learning by discovering intrinsic options, 2021. URL https://arxiv.org/abs/2101.06521.

[19] Akshit Sinha, Arvindh Arun, Shashwat Goel, Steffen Staab, and Jonas Geiping. The illusion of diminishing returns: Measuring long-horizon execution in llms, 2025. URL https://arxiv.org/abs/2509.09677v1.

[20] Jacob Hilton. The effect of horizon length on scaling laws, 2023. URL https://www.lesswrong.com/posts/ouEQLAca3bbiimAFQ/the-effect-of-horizon-length-on-scaling-laws.

[21] Jacob Hilton, Jacob Tang, and John Schulman. Scaling laws for single-agent reinforcement learning, 2023. URL https://arxiv.org/abs/2301.13442.

[22] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation, 2018. URL https://arxiv.org/abs/1506.02438.

[23] Weixuan Wang, Dongge Han, Daniel Madrigal Diaz, Jin Xu, Victor Rühle, and Saravan Rajmohan. Odysseybench: Evaluating llm agents on long-horizon complex office application workflows, 2025. URL https://arxiv.org/abs/2508.09124.

[24] Thomas Kwa, Ben West, Joel Becker, Amy Deng, Katharyn Garcia, Max Hasin, Sami Jawhar, Megan Kinniment, Nate Rush, Sydney Von Arx, Ryan Bloom, Thomas Broadley, Haoxing Du, Brian Goodrich, Nikola Jurkovic, Luke Harold Miles, Seraphina Nix, Tao Lin, Neev Parikh, David Rein, Lucas Jun Koba Sato, Hjalmar Wijk, Daniel M. Ziegler, Elizabeth Barnes, and Lawrence Chan. Measuring ai ability to complete long tasks, 2025. URL https://arxiv.org/abs/2503.14499.

[25] Matt botvinick on the spontaneous emergence of learning, 2023. URL https://www.lesswrong.com/posts/Wnqua6eQkewL3bqsF/matt-botvinick-on-the-spontaneous-emergence-of-learning.

[26] Gwern Branwen. The scaling hypothesis, 2020. URL https://gwern.net/scaling-hypothesis.

[27] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

[28] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/pdf/2001.08361.

[29] Hugging Face. Perplexity, 2020. URL https://huggingface.co/docs/transformers/en/perplexity.

[30] Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza

Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. Training compute-optimal large language models, 2022. URL https://arxiv.org/pdf/2203.15556.

[31] Wenhan Xiong, Jingyu Liu, Igor Molybog, Hejia Zhang, Prajjwal Bhargava, Rui Hou, Louis Martin, Rashi Rungta, Karthik Abinav Sankararaman, Barlas Oguz, Madian Khabsa, Han Fang, Yashar Mehdad, Sharan Narang, Kshitiz Malik, Angela Fan, Shruti Bhosale, Sergey Edunov, Mike Lewis, Sinong Wang, and Hao Ma. Effective long-context scaling of foundation models, 2023. URL https://arxiv.org/abs/2309.16039.

[32] Blake Bordelon, Mary I. Letey, and Cengiz Pehlevan. Theory of scaling laws for in-context regression: Depth, width, context and time, 2025. URL https://arxiv.org/abs/2510.01098.

[33] Sarthak Vishnu, Sahil, and Naman Garg. Unveiling the role of gpt-4 in solving leetcode programming problems. *Computer Applications in Engineering Education*, 33(1):e22815, 2025. doi: https://doi.org/10.1002/cae.22815. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/cae.22815.

[34] OpenAI. Learning to reason with llms, 2024. URL https://openai.com/index/learning-to-reason-with-llms/.

[35] Dwarkesh Patel and Gavin Leech. The scaling era: An oral history of ai, 2019–2025, 2024. URL https://dokumen.pub/the-scaling-era-an-oral-history-of-ai-20192025-1953953557-9781953953551.html.

[36] NPR. Ai is helping scientists decode how sperm whales communicate, 2024. URL https://www.npr.org/2024/05/20/1198910024/ai-sperm-whales-communication-language.

[37] Computer Timeline. Thomas ross and stevenson smith, 1935. URL http://www.computer-timeline.com/timeline/thomas-ross-and-stevenson-smith/.

[38] Jack Bell, Luigi Quarantiello, Eric Nuertey Coleman, Lanpei Li, Malio Li, Mauro Madeddu, Elia Piccoli, and Vincenzo Lomonaco. The future of continual learning in the era of foundation models: Three key directions, 2025. URL https://arxiv.org/abs/2506.03320.

[39] Tarun Raheja and Nilay Pochhi. Foundation models meet continual learning: Recent advances, challenges, and future directions. In *NeurIPS 2024 Workshop on Scalable Continual Learning for Lifelong Foundation Models*, 2024. URL https://openreview.net/forum?id=JhGhBJC7FM.

[40] Yutao Yang, Jie Zhou, Xuanwen Ding, Tianyu Huai, Shunyu Liu, Qin Chen, Yuan Xie, and Liang He. Recent advances of foundation language models-based continual learning: A survey, 2024. URL https://arxiv.org/abs/2405.18653.

[41] Hong et.al Chroma Research. Context rot, 2025. URL https://research.trychroma.com/context-rot.

[42] Prithviraj Singh Shahani, Kaveh Eskandari Miandoab, and Matthias Scheutz. Noise injection systemically degrades large language model safety guardrails, 2025. URL https://arxiv.org/abs/2505.13500.

[43] Kiran Vodrahalli, Santiago Ontanon, Nilesh Tripuraneni, Kelvin Xu, Sanil Jain, Rakesh Shivanna, Jeffrey Hui, Nishanth Dikkala, Mehran Kazemi, Bahare Fatemi, Rohan Anil, Ethan Dyer, Siamak Shakeri, Roopali Vij, Harsh Mehta, Vinay Ramasesh, Quoc Le, Ed Chi, Yifeng Lu, Orhan Firat, Angeliki Lazaridou, Jean-Baptiste Lespiau, Nithya Attaluri, and Kate Olszewska. Michelangelo: Long context evaluations beyond haystacks via latent structure queries, 2024. URL https://arxiv.org/abs/2409.12640.

[44] Anthropic. Effective context engineering for ai agents, 2024. URL https://www.anthropic.com/engineering/effective-context-engineering-for-ai-agents.

[45] Haoran Wei, Yaofeng Sun, and Yukun Li. Deepseek-ocr: Contexts optical compression, 2025. URL https://arxiv.org/abs/2510.18234.

[46] Alexia Jolicoeur-Martineau. Less is more: Recursive reasoning with tiny networks, 2025. URL https://arxiv.org/abs/2510.04871.

[47] Hua Shen, Tiffany Knearem, Reshmi Ghosh, Kenan Alkiek, Kundan Krishna, Yachuan Liu, Ziqiao Ma, Savvas Petridis, Yi-Hao Peng, Li Qiwei, Sushrita Rakshit, Chenglei Si, Yutong Xie, Jeffrey P. Bigham, Frank Bentley, Joyce Chai, Zachary Lipton, Qiaozhu Mei, Rada Mihalcea, Michael Terry, Diyi Yang, Meredith Ringel Morris, Paul Resnick, and David Jurgens. Position: Towards bidirectional human-ai alignment, 2025. URL https://arxiv.org/abs/2406.09264.

[48] Yoshua Bengio. The consciousness prior, 2019. URL https://arxiv.org/abs/1709.08568.

[49] Anirudh Goyal and Yoshua Bengio. Inductive biases for deep learning of higher-level cognition, 2022. URL https://arxiv.org/abs/2011.15091.

[50] Xin Wang, Hong Chen, Si'ao Tang, Zihao Wu, and Wenwu Zhu. Disentangled representation learning, 2024. URL https://arxiv.org/abs/2211.11695.

[51] Ilia Sucholutsky, Lukas Muttenthaler, Adrian Weller, Andi Peng, Andreea Bobu, Been Kim, Bradley C. Love, Christopher J. Cueva, Erin Grant, Iris Groen, Jascha Achterberg, Joshua B. Tenenbaum, Katherine M. Collins, Katherine L. Hermann, Kerem Oktar, Klaus Greff, Martin N. Hebart, Nathan Cloos, Nikolaus Kriegeskorte, Nori Jacoby, Qiuyi Zhang, Raja Marjieh, Robert Geirhos, Sherol Chen, Simon Kornblith, Sunayana Rane, Talia Konkle, Thomas P. O'Connell, Thomas Unterthiner, Andrew K. Lampinen, Klaus-Robert Müller, Mariya Toneva, and Thomas L. Griffiths. Getting aligned on representational alignment, 2024. URL https://arxiv.org/abs/2310.13018.

[52] C. E. Shannon. Prediction and entropy of printed english. *The Bell System Technical Journal*, 30(1): 50–64, 1951. doi: 10.1002/j.1538-7305.1951.tb01366.x.

[53] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL https://arxiv.org/abs/1706.03762.

[54] J. Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(78)90005-5. URL https://www.sciencedirect.com/science/article/pii/0005109878900055.

[55] Fernando Diaz and Michael Madaio. Scaling laws do not scale, 2024. URL https://arxiv.org/abs/2307.03201.

[56] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022. URL https://arxiv.org/abs/2203.02155.

[57] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023. URL https://arxiv.org/abs/2201.11903.

[58] Daniel Kahneman. *Thinking, fast and slow*. Farrar, Straus and Giroux, New York, 2011. ISBN 9780374275631 0374275637. URL https://www.amazon.de/Thinking-Fast-Slow-Daniel-Kahneman/dp/0374275637/ref=wl_it_dp_o_pdT1_nS_nC?ie=UTF8&colid=151193SNGKJT9&coliid=I3OCESLZCVDFL7.

[59] Amanda Dsouza, Christopher Glaze, Changho Shin, and Frederic Sala. Evaluating language model context windows: A "working memory" test and inference-time correction, 2024. URL https://arxiv.org/abs/2407.03651.

[60] Kerui Huang, Shuhan Liu, Xing Hu, Tongtong Xu, Lingfeng Bao, and Xin Xia. Reasoning efficiently through adaptive chain-of-thought compression: A self-optimizing framework, 2025. URL https://arxiv.org/abs/2509.14093.

[61] Thomas Parr, Giovanni Pezzulo, and Karl Friston. Beyond markov: Transformers, memory, and attention. *Cognitive Neuroscience*, 16(1-4):5–23, 2025. doi: 10.1080/17588928.2025.2484485. URL https://doi.org/10.1080/17588928.2025.2484485. PMID: 40235046.

[62] L. Tunstall, L. von Werra, and T. Wolf. *Natural Language Processing with Transformers*. O'Reilly Media, 2022. ISBN 9781098103217. URL https://books.google.co.in/books?id=nzxbEAAAQBAJ.

[63] Toby. How well does rlscale?, October 2025. URL https://www.lesswrong.com/posts/xpj6KhDM9bJybdnEe/how-well-does-rl-scale.

[64] Andy L. Jones. Scaling scaling laws with board games, 2021. URL https://arxiv.org/abs/2104.03113.

[65] Pablo Villalobos and David Atkinson. Trading off compute in training and inference, 2023. URL https://epoch.ai/publications/trading-off-compute-in-training-and-inference. Accessed: 2025-12-16.

[66] Shireen Kudukkil Manchingal, Andrew Bradley, Julian F. P. Kooij, Keivan Shariatmadar, Neil Yorke-Smith, and Fabio Cuzzolin. Epistemic artificial intelligence is essential for machine learning models to truly 'know when they do not know', 2025. URL https://arxiv.org/abs/2505.04950.

[67] Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. URL https://distill.pub/2020/circuits/zoom-in.

[68] Chelsea Voss, Nick Cammarata, Gabriel Goh, Michael Petrov, Ludwig Schubert, Ben Egan, Swee Kiat Lim, and Chris Olah. Visualizing weights. *Distill*, 2021. doi: 10.23915/distill.00024.007. URL https://distill.pub/2020/circuits/visualizing-weights.

[69] Chris Olah, Alexander Mordvintsev, and Ludwig Schubert. Feature visualization. *Distill*, 2017. doi: 10.23915/distill.00007. URL https://distill.pub/2017/feature-visualization.

[70] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Scott Johnston, Andy Jones, Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. In-context learning and induction heads, 2022. URL https://arxiv.org/abs/2209.11895.

[71] Nick Cammarata, Gabriel Goh, Shan Carter, Chelsea Voss, Ludwig Schubert, and Chris Olah. Curve circuits. *Distill*, 2021. doi: 10.23915/distill.00024.006. URL https://distill.pub/2020/circuits/curve-circuits.

[72] Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu, Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter, Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. URL https://transformer-circuits.pub/2023/monosemantic-features/index.html.

[73] Nelson Elhage, Tristan Hume, Catherine Olsson, Nicholas Schiefer, Tom Henighan, Shauna Kravec, Zac Hatfield-Dodds, Robert Lasenby, Dawn Drain, Carol Chen, Roger Grosse, Sam McCandlish, Jared Kaplan, Dario Amodei, Martin Wattenberg, and Christopher Olah. Toy models of superposition, 2022. URL https://arxiv.org/abs/2209.10652.

[74] Adly Templeton, Tom Conerly, Jonathan Marcus, Jack Lindsey, Trenton Bricken, Brian Chen, Adam Pearce, Craig Citro, Emmanuel Ameisen, Andy Jones, Hoagy Cunningham, Nicholas L Turner, Callum McDougall, Monte MacDiarmid, C. Daniel Freeman, Theodore R. Sumers, Edward Rees, Joshua Batson, Adam Jermyn, Shan Carter, Chris Olah, and Tom Henighan. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

[75] Jiaxin Wen, Ruiqi Zhong, Akbir Khan, Ethan Perez, Jacob Steinhardt, Minlie Huang, Samuel R. Bowman, He He, and Shi Feng. Language models learn to mislead humans via rlhf, 2024. URL https://arxiv.org/abs/2409.12822.

[76] Lauro Langosco, Jack Koch, Lee Sharkey, Jacob Pfau, Laurent Orseau, and David Krueger. Goal

misgeneralization in deep reinforcement learning, 2023. URL https://arxiv.org/abs/2105.14111.

[77] Monte MacDiarmid, Benjamin Wright, Jonathan Uesato, Joe Benton, Jon Kutasov, Sara Price, Naia Bouscal, Sam Bowman, Trenton Bricken, Alex Cloud, Carson Denison, Johannes Gasteiger, Ryan Greenblatt, Jan Leike, Jack Lindsey, Vlad Mikulik, Ethan Perez, Alex Rodrigues, Drake Thomas, Albert Webson, Daniel Ziegler, and Evan Hubinger. Natural emergent misalignment from reward hacking in production rl, 2025. URL https://arxiv.org/abs/2511.18397.

[78] Lee Sharkey, Bilal Chughtai, Joshua Batson, Jack Lindsey, Jeff Wu, Lucius Bushnaq, Nicholas Goldowsky-Dill, Stefan Heimersheim, Alejandro Ortega, Joseph Bloom, Stella Biderman, Adria Garriga-Alonso, Arthur Conmy, Neel Nanda, Jessica Rumbelow, Martin Wattenberg, Nandi Schoots, Joseph Miller, Eric J. Michaud, Stephen Casper, Max Tegmark, William Saunders, David Bau, Eric Todd, Atticus Geiger, Mor Geva, Jesse Hoogland, Daniel Murfet, and Tom McGrath. Open problems in mechanistic interpretability, 2025. URL https://arxiv.org/abs/2501.16496.

[79] Yossi Gandelsman, Alexei A Efros, and Jacob Steinhardt. Interpreting CLIP's image representation via text-based decomposition. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=5Ca9sSzuDp.

[80] Stephen Casper, Max Nadeau, Dylan Hadfield-Menell, and Gabriel Kreiman. Robust feature-level adversaries are interpretability tools, 2023. URL https://arxiv.org/abs/2110.03605.

[81] Alexander Matt Turner, Lisa Thiergart, Gavin Leech, David Udell, Juan J. Vazquez, Ulisse Mini, and Monte MacDiarmid. Steering language models with activation engineering, 2024. URL https://arxiv.org/abs/2308.10248.

[82] Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models, 2024. URL https://arxiv.org/abs/2311.03658.

[83] Anton Korznikov, Andrey Galichin, Alexey Dontsov, Oleg Y. Rogov, Ivan Oseledets, and Elena Tutubalina. The rogue scalpel: Activation steering compromises llm safety, 2025. URL https://arxiv.org/abs/2509.22067.

[84] Tianle Gu, Kexin Huang, Zongqi Wang, Yixu Wang, Jie Li, Yuanqi Yao, Yang Yao, Yujiu Yang, Yan Teng, and Yingchun Wang. Probing the robustness of large language models safety to latent perturbations, 2025. URL https://arxiv.org/abs/2506.16078.

[85] Zhihao Xu, Ruixuan Huang, Changyu Chen, and Xiting Wang. Uncovering safety risks of large language models through concept activation vector, 2024. URL https://arxiv.org/abs/2404.12038.

[86] Jacob Dunefsky. One shot steering vectors cause emergent misalignment, too, 2025. URL https://www.lesswrong.com/posts/kcKnKHTHycHeRhcHF/one-shot-steering-vectors-cause-emergent-misalignment-too.

[87] Andy Zou, Zifan Wang, Nicholas Carlini, Milad Nasr, J. Zico Kolter, and Matt Fredrikson. Universal and transferable adversarial attacks on aligned language models, 2023. URL https://arxiv.org/abs/2307.15043.

[88] Xinpeng Wang, Nitish Joshi, Barbara Plank, Rico Angell, and He He. Is it thinking or cheating? detecting implicit reward hacking by measuring reasoning effort, 2025. URL https://arxiv.org/abs/2510.01367.

[89] Yuchun Miao, Liang Ding, Sen Zhang, Rong Bao, Lefei Zhang, and Dacheng Tao. Information-theoretic reward modeling for stable rlhf: Detecting and mitigating reward hacking, 2025. URL https://arxiv.org/abs/2510.13694.

[90] evhub. Gradient hacking, 2019. URL https://www.lesswrong.com/posts/uXH4r6MmKPedk8rMA/gradient-hacking.

[91] Zhe Li, Wei Zhao, Yige Li, and Jun Sun. Where did it go wrong? attributing undesirable llm behaviors via representation gradient tracing, 2025. URL https://arxiv.org/abs/2510.02334.

[92] OpenAI. Evaluating chain of thought monitorability, 2025. URL https://openai.com/index/evaluating-chain-of-thought-monitorability/.

[93] Robert Kirk, Ishita Mediratta, Christoforos Nalmpantis, Jelena Luketina, Eric Hambro, Edward Grefenstette, and Roberta Raileanu. Understanding the effects of rlhf on llm generalisation and diversity, 2024. URL https://arxiv.org/abs/2310.06452.

[94] Nathan Lambert. Reinforcement learning from human feedback, 2025. URL https://arxiv.org/abs/2504.12501.

[95] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

[96] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor, 2018. URL https://arxiv.org/abs/1801.01290.

[97] Ralph Allan Bradley and Milton E. Terry. Rank analysis of incomplete block designs: I. the method of paired comparisons. *Biometrika*, 39(3/4):324–345, 1952. ISSN 00063444, 14643510. URL http://www.jstor.org/stable/2334029.

[98] Hao Sun, Yunyi Shen, and Jean-Francois Ton. Rethinking bradley-terry models in preference-based reward modeling: Foundations, theory, and alternatives, 2025. URL https://arxiv.org/abs/2411.04991.

[99] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[100] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models, 2021. URL https://arxiv.org/abs/2106.09685.

[101] Xumeng Wen, Zihan Liu, Shun Zheng, Shengyu Ye, Zhirong Wu, Yang Wang, Zhijian Xu, Xiao Liang, Junjie Li, Ziming Miao, Jiang Bian, and Mao Yang. Reinforcement learning with verifiable

rewards implicitly incentivizes correct reasoning in base llms, 2025. URL https://arxiv.org/abs/2506.14245.

[102] DeepSeek-AI, Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, Xiaokang Zhang, Xingkai Yu, Yu Wu, Z. F. Wu, Zhibin Gou, Zhihong Shao, Zhuoshu Li, Ziyi Gao, Aixin Liu, Bing Xue, Bingxuan Wang, Bochao Wu, Bei Feng, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Qu, Hui Li, Jianzhong Guo, Jiashi Li, Jiawei Wang, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, J. L. Cai, Jiaqi Ni, Jian Liang, Jin Chen, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Liang Zhao, Litong Wang, Liyue Zhang, Lei Xu, Leyi Xia, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Meng Li, Miaojun Wang, Mingming Li, Ning Tian, Panpan Huang, Peng Zhang, Qiancheng Wang, Qinyu Chen, Qiushi Du, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, R. J. Chen, R. L. Jin, Ruyi Chen, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shengfeng Ye, Shiyu Wang, Shuiping Yu, Shunfeng Zhou, Shuting Pan, S. S. Li, Shuang Zhou, Shaoqing Wu, Shengfeng Ye, Tao Yun, Tian Pei, Tianyu Sun, T. Wang, Wangding Zeng, Wanjia Zhao, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, W. L. Xiao, Wei An, Xiaodong Liu, Xiaohan Wang, Xiaokang Chen, Xiaotao Nie, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, X. Q. Li, Xiangyue Jin, Xiaojin Shen, Xiaosha Chen, Xiaowen Sun, Xiaoxiang Wang, Xinnan Song, Xinyi Zhou, Xianzu Wang, Xinxia Shan, Y. K. Li, Y. Q. Wang, Y. X. Wei, Yang Zhang, Yanhong Xu, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Wang, Yi Yu, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yuan Ou, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yunfan Xiong, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Y. X. Zhu, Yanhong Xu, Yanping Huang, Yaohui Li, Yi Zheng, Yuchen Zhu, Yunxian Ma, Ying Tang, Yukun Zha, Yuting Yan, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhicheng Ma, Zhigang Yan, Zhiyu Wu, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Zizheng Pan, Zhen Huang, Zhipeng Xu, Zhongyu Zhang, and Zhen Zhang. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning, 2025. URL https://arxiv.org/abs/2501.12948.

[103] Lu Ma, Hao Liang, Meiyi Qiang, Lexiang Tang, Xiaochen Ma, Zhen Hao Wong, Junbo Niu, Chengyu Shen, Runming He, Yanhao Li, Bin Cui, and Wentao Zhang. Learning what reinforcement learning can't: Interleaved online fine-tuning for hardest questions, 2025. URL https://arxiv.org/abs/2506.07527.

[104] Aida Nematzadeh, Kaylee Burns, Erin Grant, Alison Gopnik, and Tom Griffiths. Evaluating theory of mind in question answering. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun'ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2392–2400, Brussels, Belgium, October-November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1261. URL https://aclanthology.org/D18-1261/.

[105] Sneheel Sarangi, Maha Elgarf, and Hanan Salam. Decompose-tom: Enhancing theory of mind reasoning in large language models through simulation and task decomposition, 2025. URL

https://arxiv.org/abs/2501.09056.

[106] Michal Kosinski. Evaluating large language models in theory of mind tasks. *Proceedings of the National Academy of Sciences*, 121(45), October 2024. ISSN 1091-6490. doi: 10.1073/pnas.2405460121. URL http://dx.doi.org/10.1073/pnas.2405460121.

[107] RobertM Eliezer Yudkowsky. Instrumental convergence, 2025. URL https://www.lesswrong.com/w/instrumental-convergence.

[108] Stephen M Omohundro. The basic ai drives. In *Artificial intelligence safety and security*, pages 47–55. Chapman and Hall/CRC, 2018. URL https://selfawaresystems.com/wp-content/uploads/2008/01/ai_drives_final.pdf.

[109] Bostrom Nick. Superintelligence: Paths, dangers, strategies. *Strategies*, 2014. URL https://books.google.co.in/books/about/Superintelligence.html?id=7_H8AwAAQBAJ&redir_esc=y.

[110] Ryan Greenblatt, Buck Shlegeris, Kshitij Sachan, and Fabien Roger. Ai control: Improving safety despite intentional subversion, 2024. URL https://arxiv.org/abs/2312.06942.

[111] Yuntao Bai, Saurav Kadavath, Sandipan Kundu, Amanda Askell, Jackson Kernion, Andy Jones, Anna Chen, Anna Goldie, Azalia Mirhoseini, Cameron McKinnon, Carol Chen, Catherine Olsson, Christopher Olah, Danny Hernandez, Dawn Drain, Deep Ganguli, Dustin Li, Eli Tran-Johnson, Ethan Perez, Jamie Kerr, Jared Mueller, Jeffrey Ladish, Joshua Landau, Kamal Ndousse, Kamile Lukosuite, Liane Lovitt, Michael Sellitto, Nelson Elhage, Nicholas Schiefer, Noemi Mercado, Nova DasSarma, Robert Lasenby, Robin Larson, Sam Ringer, Scott Johnston, Shauna Kravec, Sheer El Showk, Stanislav Fort, Tamera Lanham, Timothy Telleen-Lawton, Tom Conerly, Tom Henighan, Tristan Hume, Samuel R. Bowman, Zac Hatfield-Dodds, Ben Mann, Dario Amodei, Nicholas Joseph, Sam McCandlish, Tom Brown, and Jared Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022. URL https://arxiv.org/abs/2212.08073.

[112] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks, 2021. URL https://arxiv.org/abs/2005.11401.

[113] Jingbo Shang, Zai Zheng, Jiale Wei, Xiang Ying, Felix Tao, and Mindverse Team. Ai-native memory: A pathway from llms towards agi, 2024. URL https://arxiv.org/abs/2406.18312.

[114] Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. Sparks of artificial general intelligence: Early experiments with gpt-4, 2023. URL https://arxiv.org/abs/2303.12712.

[115] Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. Ruler: What's the real context size of your long-context language models?, 2024. URL https://arxiv.org/abs/2404.06654.

[116] Angeliki Giannou, Shashank Rajput, Jy yong Sohn, Kangwook Lee, Jason D. Lee, and Dimitris

Papailiopoulos. Looped transformers as programmable computers, 2023. URL https://arxiv.org/abs/2301.13196.

[117] Delong Chen, Mustafa Shukor, Theo Moutakanni, Willy Chung, Jade Yu, Tejaswi Kasarla, Allen Bolourchi, Yann LeCun, and Pascale Fung. Vl-jepa: Joint embedding predictive architecture for vision-language, 2025. URL https://arxiv.org/abs/2512.10942.

[118] Mahmoud Assran, Quentin Duval, Ishan Misra, Piotr Bojanowski, Pascal Vincent, Michael Rabbat, Yann LeCun, and Nicolas Ballas. Self-supervised learning from images with a joint-embedding predictive architecture, 2023. URL https://arxiv.org/abs/2301.08243.

[119] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006. URL https://web.stanford.edu/class/cs379c/archive/2012/suggested_reading_list/documents/LeCunetal06.pdf.

[120] Yuzhong Hong, Hanshan Zhang, Junwei Bao, Hongfei Jiang, and yang song. Energy-based preference model offers better offline alignment than the bradley-terry preference model. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=t5QNCIltAn.

[121] Gary Marcus. Innateness, alphazero, and artificial intelligence, 2018. URL https://arxiv.org/abs/1801.05667.