High off Our Own Loss Function: The Devil's Bargain Hypothesis, and the Rabbit Hole deep enough to found a Field

Chris O'Quinn

April 16, 2025

Abstract

This is not a standard machine learning paper. There are no benchmark results, no leaderboard gains, and no experimental fine-tuning tricks. What began as an attempt to make a minor improvement to transformer-based reasoning instead revealed structural flaws so deep they demanded a full theoretical refactor.

We operate under three premises. 1): We want AGI capable models. 2): We want economic models. 3): The field has been trying to build these using current Vaswani technology. By following the data we arrived at the Devil's Bargain Hypothesis: that generative models have sacrificed abstraction and generalization in favor of surface-level memorization—because their metrics and optimization objectives told them to.

We argue that the field has become "High on its own loss function"—mistaking metric conformance for better cognition, and treating local prediction accuracy as a proxy for intelligence. This work synthesizes literature, proposes formal mechanisms for long-standing anomalies, and offers falsifiable predictions. The evidence is stark: blind scaling has failed not because we ran out of compute, but because we are overfitting our models.

A good engineer follows an issue to its root cause. And here, that root lies in the epistemology of Machine Learning itself. Increasing model capacity without increasing information content is a textbook recipe for overfitting—a law known since Kaplan, yet ignored at scale. This paper rejects the prevailing paradigm not for the sake of rebellion, but because the evidence leaves no alternative. The assumption that only experimental results make valid science must be challenged—especially when predictive theory can explain existing results and anticipate future ones. Experiments remain essential, but they must be guided by compact, falsifiable theory—not brute-force exploration alone.

This paper does not offer implementation fixes. This work presents only the diagnosis. Consult the two companion works for full engineering solutions and Cognition Engineering operational.

We understand others may disagree, and some may call this overreach. But we contend it is not contrarianism—it is engineering necessity. Read

the papers. Follow the data. Test the predictions. Implement the corrective mechanisms in the companion. Prove us wrong.

Contents

1 Introduction

1.1 Preamble

Unconventionally, we start with the author's note first. Of course, convention is one of the things under attack—so we begin this way to avoid frightening the reader, and to make clear that we reject the idea that a hypothesis must be experimentally validated to qualify as science, so long as it is precise, predictive, and falsifiable. Basically, I showed up to do some minor surgery on a part of the field. I ended up replacing 80% of the patient.

This journey didn't start by trying to throw out the machine learning equivalent of the kitchen sink. No, it started much humbler; a self-imposed challenge to consider how to theoretically beat ARC-AGI 2024 with a transformer model and a single zero-shot pass: This was intended as a proxy to reaching AGI. Things did not go as planned. A rough map of my descent into madness looked like this:

1. Oh, Online Language Translation (discussed later) will prime it for this task. Perfect!
2. No model can do that. What is broken? Why?
3. No model is close to doing this. Okay, what is not broken?
4. Lets rethink from First Principles for an AGI. Uh, what are our principles?
5. Our Principles are broken too!

Much like tearing out the walls to replace rotten boards may reveal deeper structural issues, each discovery uncovered deeper structural flaws. In response I had to broaden the scope to deal with it. The moment I cracked was when I realized alignment itself was broken at an economic level and saw a way to fix it. Ten months and some 500+ paper reviews later, the field needs a completely new epistemology, society requires a centralized agency developing alignment and reasoning constitutions, and a complete top-to-bottom reengineering of the Vaswani stack in terms of the principles of the new field of Cognition Engineering.

This is the journey through the data that kicked off this conclusion. The core facts that will eventually become inescapable are: we are overfitting our models; any first-year statistics student could have predicted it; there are theory remedies that are engineering-practical; and this issue reveals an epistemological hole in the field of machine learning itself.

This paper does not concern itself with the actual fixes. The companion paper, *Cognition Engineering: Post-Scaling Engineering with Composable Minimal Viable Archetypes*, is intended to address the actual challenges with ready-to-go composable engineering archetypes that are drop-in compatible with your existing stack. These archetypes are however composable for much greater synergy together. See that paper if you wish to know how to solve your issues.

If you want to know the stronger philosophy and concerns underlying Cognition Engineerings consult *Philosophy of Engineering and Science in Cognition Engineering: Operating in a Post Deep Learning World*

For those skeptics who are unconvinced and still anchored in the scaling paradigm, we intend to stand by our theory and make no apologies. **Read the paper, run the experiments, check the predictions, and prove us wrong.**

Welcome to the trip that leads to the Devil's Bargain Hypothesis.

1.2 Reading Guide

Different readers will arrive with different needs. Some want engineering fixes. Others want theory. Others want to prove us wrong.

It should be noted that this paper is written as a continuous journey of discovery in an overall narrative. It is intended to be read this way. But we acknowledge the practical reality that no one has yet figured out how to add hours onto a person's day

Below is a guide to help you find what you're looking for—whether it's a falsifiable claim, a systemic critique, or just something publishable before your grant runs out. We encourage you to read the sections and reflect on if you want to read the whole thing. Not all answers live here. But common questions have an answer.

- **Don't tell me what is wrong, just give me fixes**: This is the wrong work, go to the companion *Cognition Engineering: Post-Scaling Engineering with Composable Minimal Viable Archetypes*

- **Scaling feels broken—what's actually going wrong?**: You want to skip to section 3 on page 16. If you are impatient, do not care about long sequence models, and do not care about how to fix it you may also skip directly to section 3.4.2 on page 27 to see exactly what mechanism has been causing your feedforward layer to betray you.

- **What is the Devil's Bargain Hypothesis, and can I test it?**: The idea scaling up has actually been systematically betraying developing AGI imperatives, and the epistemology is to blame. You will want 3 on page 16 to understand, and predictions with executable experiments are located in appendix A on page 34.

- **What does this imply for alignment/corrigibility**: Very bad things. They are being attached at the most brittle layers of the model. You will want to start at the Devil's Bargain Hypothesis, particularly the channel-mixing issues that start in section 3.4 on page 26. If you are in a hurry, jump straight to mechanisms.

- **Why does my long-context model forget everything?**: You will want to go to the Devils Bargain Hypothesis. If in a hurry, try section 3.3 on page 19.

- **Why is transfer learning/catastrophic forgetting getting worse?**: See section 3.4.5 on page 29
- **How do I improve generalization without massive scale?**: This is a topic mainly discussed in the companion work, *Cognition Engineering: Post-Scaling Engineering with Composable Minimal Viable Archetypes*. They are drop-in compatible with a Vaswani stack. However, a quick skim through the reasoning in the predictions appendix A at 34 should give you some simple ideas.
- **I need a paper idea—do you have testable predictions?**: Absolutely. See appendix A at 34 for experiments of interest to many different labs on a variety of budgets that test core predictions of the DBH.
- **What's the epistemological failure in ML, and what replaces it?**: In short, theory is a third-class citizen, and a new dicipline is needed. For a quick summary in-work jump to section 4 on page 33. For applied philosophy of engineering see *Cognition Engineering: Post-Scaling Engineering with Composable Minimal Viable Archetypes*. For more of a big picture see *Philosophy of Engineering and Science in Cognition Engineering: Operating in a Post Deep Learning World.*
- **What is Cognition Engineering, and how is it different?** See the last answer: "What is the epistomological..." and follow your preferred path.
- **I don't believe there's a problem—convince me.**: To be clear, I'm not saying NLP is useless—I'm saying the current paradigm cannot produce AGI or anything more cognitively advanced.Not that NLP is useless. See section 2.4 on page 14.

1.3 Introduction

The current trajectory of Generative NLP—what we call the **Big Data Scaling paradigm**—assumes that intelligence will emerge from more data, larger models, and increasingly elaborate fine-tuning through **brute-force** scaling. This approach delivered remarkable early progress, particularly on tasks grounded in static knowledge and pattern completion. But in recent years, generalization gains have slowed—as though asymptotics. Models keep growing, but their ability to reason, abstract, and retain knowledge across time does not scale with them.

This is not an accident.

We contend—and will demonstrate—that these limitations are not incidental. They are structural consequences of the paradigm itself. No amount of additional scaling, data, or fine-tuning will fix them. The Big Data Scaling paradigm is broken in foundational ways. Blind scaling should be treated as dead—and we are overfitting models that were never engineered for intelligence in the first place. We must reject the paradigm outright. And even then, the issues go deeper still into epistemological incentive misalignments in the Machine Learning scientific incentives themselves.

1.4 The Defining Challenge: Online Language Learning as a Benchmark for Intelligence

At the core of intelligence is the ability to **continuously adapt**—to encounter new information, restructure internal representations, and refine understanding through experience. Yet modern transformers fail to do this in any meaningful way. They generalize from **pretraining**, not from **real-time interaction**.

To highlight this limitation, we introduce the idea of **Online Language Translation** as a benchmark for intelligence. Imagine a model trained entirely in English. It is then provided with a limited corpus of English–German translations—a small "Rosetta Stone" establishing an initial mapping. From that point forward, the model is fed **German–English text pairs in real time** and must learn to translate dynamically—adapting as it goes, without retraining or gradient updates. These pairs may not be exact replicas—some may be masked, partial, or abstracted summaries. This more closely reflects how real-world research unfolds.

Current architectures fail. Unlike humans—who infer meaning dynamically and refine abstractions over time—transformers lack any mechanism for structured, cumulative abstraction learning. Instead, they rely on:

- **Massive pretraining**—where generalization is a function of statically encoding massive volumes of data, and responses are generated by pattern matching, not grounded conceptual synthesis. This is effective for static domains, but not for continual learning or genuine research. Even augmentations like chain-of-thought prompting have limits.
- **Short-range in-context learning**—which enables transient adaptation, but lacks mechanisms for lasting memory consolidation or the formation of reusable abstractions.
- **Static memorized knowledge**-Based on a large range of memorized patterns, the transformer can respond well in-distribution. But because that knowledge is primarily statically encoded in feedforward parameters, it cannot build upon abstractions in a compositional or cumulative way—only adjust to a limited degree.

This is **not**, we contend, a minor limitation—it is a fault line in the very architecture. **Modern AI does not "learn" in the sense that intelligence requires.** It recalls, maps, and extrapolates, but it does not synthesize. It cannot build persistent, general-purpose abstractions over time. A model limited to static pretraining and in-context prompts would be incapable of solving the benchmark challenge because the task demands constructing and updating a *library of dynamic knowledge*—something transformers, as currently designed, fundamentally cannot do.

We do not wish to undersell the achievements of the field so far. Some research lines have made partial progress toward this goal—including memory augmentation and chain-of-thought scaffolding. In particular, memory augmentation strategies show promise. But we argue that due to architectural antipatterns—emphasizing memorization over cognition—none of these will scale *economically* to support Artificial General Intelligence. They are patches atop a paradigm that was never designed to generalize. Intelligence demands synthesis. And until our models can build, retain, and refine abstractions in real time, they will not cross the line into genuine cognition—no matter how large we scale them.

1.5 The Purpose of This Paper

Most fields of science have room for theorists. In physics, for example, a theory-only paper is treated as real science so long as it interfaces with the data, correlates it under a predictive mechanism, and explains anomalies. They perform critical steps in the scientific method by correlating anomalies into new underlying principles. However, Machine Learning largely lacks this discipline-at least to an acknowledged and serious level. Given that overfitting—caused by increasing model capacity without corresponding increases in regularization or information—is a failure a first-year statistics student could predict, we must reject the prevailing Empiricist-First paradigm of Machine Learning in favor of one that takes predictive theory seriously.

To explore where this new viewpoint leads, two companion papers accompany this work. The first, "Cognition Engineering: Post-Scaling Philosophy with Composable Minimal Viable Archetypes", introduces practical engineering strategies grounded in alternative principles, derived from first principles, and designed as drop-in replacements to various portions of the Vaswani stack Your labs can, in essence, begin testing immediately. A second companion piece, "Philosophy of Engineering and Science in Cognition Engineering: Operating in a Post Deep Learning World" discusses the philosophy we now need to operate under in this new paradigm that was used to build Cognition engineering. These papers are companions to "High on our own loss functions", and definitely show what happens when engineering and principles work together, then how to do so into the future.

The remainder of this paper does a surgical post-mortum on the prevaling Blind Scaling Vaswani paradigm. We identify exactly where the paradigm fails—how its architectures and training loops reward memorization over abstraction, and where brittle scaling has replaced scalable cognition. Then we excise the rot: the assumptions, metrics, and incentives that have silently constrained generalization. In their place, we propose minimal principles and training solutions that restore the pursuit of general intelligence; these will be fleshed out to their full potential in the companion works.

Do not take this to be a repeal of what still works. We do not throw out everything from the standard Big Data Paradigm - indeed, certain key phenomenon like the usage of Big Data itself and curated data are still useful. We also do not throw out Empiricism. Indeed, theory without validation is *worthless*. In essence, we identify that existing transformer technology has only **barely** been working, not because of parameter scaling - but in *spite* of it, and this would had been caught if the epistemological incentives of the field were setup to reward concise theory done according to the scientific method. This paper is not merely a critique. It is the beginning of a change in epistemology itself.

1.6 Evidence and Limitations

This work is currently being conducted by an independent researcher without large-scale compute resources, and hence, large-scale empirical experiments are not presented; Some minor experiments are present where reasonable, but are strictly limited to small experimental observations rather than full experimental studies. This is unfortunate, as ablation studies would be wonderful for testing the predictions.

In lieu of this, this paper offers testable predictions that could be falsified with minimal or moderate resources—making them suitable for open, collaborative validation.Such predictions are clearly marked, along with the consequences that would falsify the hypothesis. Additionally a few key experiments are also explored where budget permits.

Theory is my primary tool. However, it turns out, however, it has been significantly underestimated. Coming from my background in physics, it's clear that the Machine Learning field is in dire need of 'Theoretical ML Engineers'—those who can carry anomalies all the way through to testable experimental designs This is not a work of authority granted by compute, or a contrarian rebellion—it is one earned through first principles predictive mechanisms, anomaly explanation, and engineering necessity. The experiments are yours to run. The predictions are here to break.

1.7 Concluding Remarks

By the end of this paper, it should be clear that a fundamental shift in NLP is not just warranted—it is now feasible. We have shown that the Big Data Scaling paradigm, while historically effective, imposes hard limits on generalization, abstraction, and long-term reasoning. These are not incidental weaknesses. They are structural consequences of the paradigm itself.

But this paper is not merely a diagnosis. It is the justification for writing the refurbishment manual.

By embracing a new foundation—rooted in better first principle analysis, simplicity, and synergies—we will be able recover in the companion papers the ability to build models that learn continuously, generalize robustly, and align predictably. The companion papers will cover, among other things, how to make models that:

- Scale without runaway memory demands or overfitting.
- Train models that accumulate and refine knowledge over time.
- Align models through structured, reusable reinforcement cycles, in a manner that may extend even to general intelligence.
- Deploy architectures that remain drop-in compatible with transformer infrastructure but operate under an entirely different set of principles.

What emerges is not a marginally improved transformer—but a fundamentally new trajectory for NLP. A paradigm where dynamic learning, alignment, and generalization are not side effects of scale, but design targets in their own right.

This paper marks the beginning of that shift.

2 Background and Motivations

We begin the shift by understanding where we stand now. This work stands on Generative Natural Language Processing, operating in the current Big Data Scaling paradigm. To understand the issues, and the upcoming shift, we must understand this paradigm first.

2.1 What has driven the success of the transformer?

The success of the transformer has been astonishing. Commercially available artificial intelligence is now a fact of life. We argue that three key factors have driven the transformer—and deep learning more broadly—into dominance as the prevailing NLP paradigm. These factors make up what we shall call the **Big Data Scaling (BDS)** paradigm:

- **Scaling Laws:** As models grow, they exhibit emergent capabilities and better generalization. The assumption has thus developed that for any given architecture, **blind scaling**—simply adding more parameters without fundamental changes—is presumed to yield better performance. The empirical findings of Kaplan in *Scaling Laws for Neural Language Models* demonstrated that increasing model size, dataset volume, and compute resources led to this conclusion. These findings cemented a dominant strategy in machine learning: scale up models, scale up data, scale up compute. [14]
- **Big Data and Pretraining:** Large-scale datasets provide the raw material for learning. We assume that larger, high-quality datasets will consistently yield improved performance. Kaplan also supports this view—and indeed, we will ultimately argue that the principle that more data is better still holds, even after the paradigm shift.

- **Transfer Learning and Modularity:** Fine-tuning and modular extensions enable adaptability and real-world applicability. We assume that once we have a model that is pretrained, the ingrained knowledge will be successfully transferred to the application using fine tuning in transfer learning.

The impact of these paradigm laws became evident with GPT-3 [4], where emergent abilities such as in-context learning and few-shot generalization arose purely from model scale, without explicit architectural changes. To take advantage of these larger *foundation* models, a trend of pairing them with ever-larger datasets and transfer learning quickly emerged. By training on massive corpora such as The Pile [8] our models have internalized diverse linguistic structures, factual knowledge, and even cultural nuances. This statically defined knowledge makes them effective tools for computational reasoning and recall within their trained domain. Transfer learning has usually finished out this trifecta, consisting of fine-tuning the foundation on a specific task and thus bringing to bear all the memorized knowledge in a format relevant to the task at hand [11]. *On tasks that can be statically trained using abundant existing data in the problem domain, we do not deny that the transformer—when paired with the Big Data trifecta—performs excellently.*

2.2 Successes of the Transformer

The transformer has seen usage at a commercial level in a wide variety of scenarios in which knowledge can be statically trained.

One of the earliest and most transformative applications of transformers has been in search engines. Google integrated BERT-based ranking models [21] to improve query understanding, allowing for more accurate retrieval of relevant information. Beyond web search, enterprises have adopted LLM-powered knowledge retrieval systems for internal documentation, legal research, and scientific literature searches.

Chatbots, virtual assistants, and content generation tools powered by LLMs have also become widespread. Models like ChatGPT (GPT-4) and PaLM [6] are now integral to customer service, technical support, and general-purpose dialogue systems. More flexibility has also been integrated with the usage of tools learned by reinforcement learning in these models [23]. In addition to conversational AI, transformers drive automated content generation for marketing, writing assistance, and creative applications; An example of this would be Unity Muse capable of automating significant portions of game asset creation using textual directions [31].

In professional and enterprise settings, transformers have been widely adopted for document processing, summarization, and decision-support tasks across law, finance, and medicine [36]. Fine-tuning has enabled domain-specific adaptations, allowing models to assist in contract analysis, regulatory compliance, and financial forecasting. Other locations beyond NLP have also seen usage of transformers with the Big Data paradigm.

Generally, any task which:

- Involves comparing vector representations of data
- Has abundant training data available
- Tolerates occasional errors or permits multiple response attempts

Has proven to be an excellent fit for the existing Big Data Scaling paradigm. Significant success has been seen in predictive generation as well. These widespread successes are examples not only of transformer capabilities but also of how closely they've adhered to the principles of the Big Data Scaling paradigm: More data, bigger models, and transfer learning have been a winning solution for a huge variety of problems.

2.3 Architectural and Training Innovations: Adapting to Growth

Transformers have, in large part, been a victim of their own success. As they have been found more and more useful, earlier architectural considerations have proven to be insufficient to support the growing domain of applications we wish to use them in. As such, the Vaswani training and architecture implementation originally proposed in vaswani2018Attention has undergone significant extension. Some advancements *addressed known weaknesses* in the transformer formulation, such as its quadratic attention cost and reliance on static knowledge. Others *reinforced core strengths*, improving efficiency, adaptability, and expressivity. And some arose *out of necessity*, dictated by the sheer scale at which these models now operate.

Regardless of motivation, these innovations have enabled transformers to thrive despite growing computational and architectural demands. A non-exhaustive but representative selection of these since the original Vaswani paper includes:

- **Movement into the Encoder and Decoder NLP spaces**: While the original Vaswani transformer was an **Encoder-Decoder** model, subsequent work has extended this into purely encoder models like BERT (for ranking and summative tasks) and purely decoder-based models such as GPT (for generative tasks). The latter is particularly important to this paper as we will be focusing our attention on generative language modeling. [7, 34].
- **Optimizing Time Mixing**: The attention step of the transformer has long been known to be a computational and memory bottleneck. Operating in $O(T^2)$ time and space has made it exceptionally unforgiving when used on longer sequence lengths. A large body of research to avoid this using a variety of mechanisms such as sparse attention, linear attention, and many other attention variations have been explored to resolve this [35, 1].

- **Better positional encoding**: The absolute positional encoding used in Vaswani has been enhanced in a variety of ways, such as Rotary Positional Embeddings and a wide variety of other tricks that resolve the widely acknowledged issue with absolute embeddings by making them relational. [2]
- **Mixture of Experts**: As models have grown, computational efficiency has become a critical concern. Much of the computational cost once getting beyond the $O(N^2)$ attention barrier has proven to be due to the traditional **Feedforward** mechanism: Making deeper models has meant more computation.

  Mixture of Experts (MoE) architectures [28] have addressed this by selectively activating only a fraction of model parameters per forward pass, reducing inference costs while maintaining performance at scale. Sparse activation techniques, such as Routing Transformers [26], further refine this by dynamically selecting relevant computation paths.
- **Fast Inference**: Some extremely promising pseudo-recurrent model architectures such as the linear transformer and RWKV focused on enabling recurrence while remaining timestep-parallelizable. These developments have let inference occur extremely fast as recurrent neural networks. For reasons that will become clear later, we think this is ultimately the direction Generative Language Modeling needs to go in. [1, 24]
- **Hybrid Approaches: Retrieval-Augmented Generation (RAG) and External Memory.**
  LLMs are, naturally, almost completely static in their ability to encode knowledge. This makes them unsuitable as agents for a wide variety of practical tasks. Retrieval-Augmented Generation (RAG) [16] and other auxiliary memory mechanism integrate external information dynamically, reducing reliance on static, parameter-bound memory. Further steps have integrated this as a prior during pretraining the model expects to operate with: Retrieval-enhanced language models such as REALM [9] and extend this further by leveraging external databases and live web searches.
- **Distillation and Model Compression.**
  As scaling laws have driven ever-larger models, techniques for making them more efficient have become essential. Knowledge distillation, as demonstrated in ALBERT [15], compresses large-scale models into smaller ones while retaining much of their performance. Quantization and pruning strategies further reduce computational overhead, enabling deployment in resource-constrained environments.

These architectural and training innovations have allowed transformers to adapt and continue scaling, reinforcing their dominance in NLP and beyond. However, as models have evolved, so too have the demands placed upon them: more data, more developer hours, and increasingly intensive curation are now required to maintain and extend their capabilities.

2.4 The Ceiling of Scale: Why We Can No Longer Just Make Bigger Models

2.4.1 Introduction

For years, the scaling laws underpinning the **Big Data Scaling** training paradigm have provided a reliable path toward more advanced and general models: larger models, more data, and increased compute consistently yielded better performance.

However, while scaling has historically been effective, it is increasingly constrained in practice—especially under the current paradigm's assumptions of blind scaling. The **limits of available high-quality data**, the **growing computational inefficiencies of larger models**, and the **exploding cost of reinforcement-based fine-tuning** all indicate that we are approaching the practical ceiling of how far the **Big Data Scaling** paradigm can be pushed.

2.4.2 We Have Run Out of High-Quality Training Data

A key driver of early transformer success was the ability to train on ever-larger datasets. Indeed, Kaplan's foundational work presumes that we can increase the information content in parallel with model size. However, we are now approaching—or may have already passed—the upper bound of high-quality, publicly available text.

Models like GPT-4 have already been trained on vast portions of the internet, scraping diverse sources from books to Wikipedia to code repositories. Yet as dataset sizes have grown, so too has the proportion of low-quality, repetitive, or noisy data. This used to be manageable, as we filtered it out. But we are out of clean data to replace it with. Scaling further now requires incorporating increasingly lower-quality sources or sources with legal issues, which introduces diminishing returns and risks degrading model performance or even making the models legally questionable. Furthermore, the strategy of continuously increasing dataset size is running into a **duplication problem**—with finite high-quality data available, models risk overfitting on redundant information rather than truly learning novel patterns.

In short, **the easiest way to improve performance—feeding models more data and slightly enlarging the model—is no longer an option**. Kaplan was not wrong. But the assumptions the field has made in interpreting his work are.

2.4.3 We Are Building Larger Models to Compensate, But It Is Becoming Inefficient

With high-quality training data running out, the primary alternative has been to scale model *size* instead. While scaling laws suggest that bigger models should perform better, larger models require more compute per unit of improvement when data does not scale alongside them. This means that without additional data, **performance gains become at least quadratic in cost rather than linear** [13].

To mitigate the extreme computational costs of massive models, techniques such as **Mixture of Experts (MoE)** have emerged, allowing more parameters to be exposed without paying the full computational cost at each step. This allows models to maintain high parameter counts without incurring the full computational burden at all times. However, MoE and similar strategies highlight a fundamental reality: We are not making models larger because it is ideal, we are making them larger because it is all we have left. [28]

Even worse, this approach introduces new complications as models get smarter right as we are running out of data to train them. Large models become more challenging to fine-tune, requiring increased engineering effort to manage training stability, emergent behaviors, and parameter redundancy. And, as we will ultimately see, we are not actually getting smarter models, but just overfitting them.

2.4.4 Transfer Learning Worsens with Model Scale

Larger models with more parameters also introduce another issue: transfer learning performance appears to degrade as model size increases. This breaks a lot of our assumptions, and may have extremely dire consequences for the current LLMs Big Data Scaling paradigm.

To be specific, **Catastrophic Forgetting** is worse. In *An Empirical Study of Catastrophic Forgetting in Large Language Models During Continual Fine-tuning* by Luo et al, the authors perform a fine-tuning process on a variety of models between 1–7 billion parameters. They acknowledged that "surprisingly" the severity of Catastrophic Forgetting increases as model size increases; this is both a sign of issues that will only get worse, and a prediction that is given by the upcoming *Devil's Bargain Hypothesis* in later pages of the paper.

2.4.5 Aligning Large Models Requires Exponentially Expensive Reinforcement Learning

The economics are that our models and their patching strategies are getting incredibly expensive. One of the dominant strategies for improving alignment, **reinforcement learning with human feedback (RLHF)** [22], introduces an entirely new layer of cost.

Unlike traditional fine-tuning, which adjusts models based on static labeled datasets, RLHF requires continuous human intervention to label model outputs, curate high-quality preference data, and train reward models. This process is both labor-intensive and computationally expensive, as it involves multiple training stages: supervised fine-tuning, reward model training, and reinforcement optimization.

Other alternatives—such as Constitutional AI and related strategies—have been proposed, but none address the core economic bottlenecks that limit widespread adoption. Alignment is fundamentally broken.

2.5 Concluding Notes

The NLP field is at an inflection point: we can no longer scale our way to better performance. We have run out of high-quality data, forcing us to rely on ever-larger models to extract marginal improvements. These larger models, in turn, are exponentially more expensive to train and fine-tune—especially when human feedback is required.

Instead of solving the core problem, we are compensating for a lack of data by making models more expensive and then patching them with even costlier fine-tuning techniques. This is unsustainable.

Now that *blind scaling* is no longer a viable solution, we must confront the deeper issues that scaling previously helped mask. The next section explores these hidden challenges—problems that have existed all along. What, we ask, is the architecture actually good at, and what issues have we been hoping will just "scale away"? Given these limitations, the most straightforward strategy would be to work within the constraints of existing architectures. However, cracks in the foundation suggest this approach may not be sustainable even with patches. Why?

3 The Devil's Bargain Hypothesis

We propose the **Devil's Bargain Hypothesis** to explain these effects: Modern generative models systematically trade generalization for surface-level memorization. This tradeoff is not incidental—it is baked into the very **Big Data Scaling** foundation itself. We claim that fixing it will require us to discard one of our cherished friends: We must discard the *blind scaling* assumption, and replace it with a more complex and reasoned approach to increasing the scale of our models.

This goes against everything that has brought us so far in Generative NLP: As we have discussed in section 2.2, scale has brought phenomenal economic success. However, the logic is inescapable.

Step by brutal step, we will dissect every major component of the modern Vaswani NLP stack:

- The metrics
- The alignment
- The memory (attention)
- The cognition (feedforward)
- The loss and pretraining

By doing so, we will understand how such a major oversight could have gone unnoticed, uncover the precise mathematical mechanisms behind the Devil's Bargain, and use it to clarify anomalies in the literature that the current paradigm leaves unresolved, ultimately revealing a deeply sick system.

We finally reach a critical point where we must kill off one of the core members of Big Data Scaling at least within the NLP space: **That scaling will consistently give better performance, and that model designs can rely on scaling alone to reach intelligent behavior.** We were, in essence, **High on our own Loss Functions** the whole time, letting shallow metrics be mistaken for actual progress.

3.1 Inadequacy of Existing Metric Technology

To even permit the approach to the hypothesis that will lead to the downfall of blind scale as a paradigm, we must first convincingly understand how such an oversight could exist in the first place. This is difficult.

At face value, current transformer models consistently achieve impressive scores across common evaluation benchmarks like perplexity, BLEU, and various domain-specific metrics. They are, in a narrow technical sense, highly performant. This raises the question: if our models suffer from fundamental generalization failures, why are such issues not evident from these evaluations?

The core issue, we argue, is that existing evaluation metrics are predominantly sensitive to local syntactic correctness rather than to the deeper cognitive abstractions necessary for true generalization and lifelong learning. More explicitly, current metrics measure how accurately a model predicts individual tokens within a sequence—primarily focusing on local, syntactic constraints—rather than assessing whether a model is genuinely internalizing and abstracting conceptual knowledge.

Two closely interconnected effects explain why current metrics display this tendency:

1. **Local Syntax Dominates Model Loss**:

   Most next-token predictions in natural language are determined by immediate syntactic or grammatical context rather than by abstract reasoning or long-range dependency modeling. To illustrate, consider examples like predicting the next token as "a" or "an," which depends entirely on local phonetic rules rather than broader textual understanding. Both "a" and "an" might often be grammatically acceptable continuations, yet the model will only know which is valid two words from then. This causes unavoidable prediction ambiguity thereby imposing a baseline level of prediction error.

   Crucially, such minor but frequent syntactic ambiguities create a substantial floor of unavoidable prediction errors. Even models perfectly tuned to their training distributions cannot fully eliminate these local mispredictions.

2. **True Generalization Signals Are Numerically Insignificant**:

   This prevalence of local syntactic errors generates a severe control problem in existing evaluation paradigms. Metrics such as perplexity, which aggregate prediction accuracy across every token in a long sequence, treat each token as equally important.

However, tokens affected by long-term abstract reasoning are relatively rare in most texts. As such, improvements from generalization are too small relative to the dominant influence of local syntax errors to meaningfully shift overall evaluation scores. The vast majority of tokens are predictable through simple grammatical rules, not deeper cognitive modeling.

Consequently, even if a model improves its ability to generalize—by better abstracting or internalizing knowledge—the resulting gains contribute minimally to the aggregate loss. Existing metrics thus mask critical failures of abstraction because these cognitively driven mistakes, while crucial for AGI development, are numerically overshadowed by frequent and unavoidable syntactic noise.

The fundamental issue is that these metrics primarily measure a model's ability to predict local syntax from immediate and global context, rather than assessing its capacity for lifelong abstraction and generalization. This is, we will not deny, fine for many purposes. *However, it is an invalid assumption to make when attempting to build an AGI Generative NLP Model.* And this mistake has caused Generative Language Models to be evaluated using the wrong assumptions.

There is some evidence to support this conclusion. In particular, Liu's excellent "Lost in the Middle" notes that models that look by many metrics performant over long sequences nonetheless tend to only remember and emphasize short-term relationships or globally relevant content - regional content ends up "Lost in the middle". [19]. This is precisely the effect we would predict.

We now understand how the claimed oversight could have happened: **In essence, the field of Generative NLP has not been converging towards intelligence, but simply getting high off its own loss function.** . Conflating metrics performance with actual intelligence has been the real issue all along.

3.1.1 Steps towards resolution

Assuming this diagnosis is correct, a natural next question arises: how do we evaluate this effect?

Huang et al. found in "Compression Represents Intelligence Linearly" that compressing information linearly was correlated with intelligence. A wide body of additional work agrees. [12, 5]. As such, we would propose rebuilding the field under the domain of encouraging compressive pressures through training incentives and model modifications to encourage genuine abstraction pressures rather than just memorization.

3.1.2 Two Pass Perplexity Ratio

The **Two-Pass Perplexity Ratio** is an immediate instantiation of this philosophy, designed to measure intelligence simply and far more passively. It involves a two-pass perplexity-based assessment:

1. **First Pass (Initial Exposure)**: The model is presented with a long, information-rich passage, and generates responses normally. This initial run establishes a baseline performance. The passage is long enough to saturate immediate attention capacity.
2. **Second Pass (Repeated Exposure)**: The same passage is presented again to the same model, now able to leverage any knowledge or abstractions formed during its first exposure. Improvement on the second pass requires compressing and integrating key abstractions rather than mere token-level memorization.

The results are then formed into a ratio of pass two perplexity over pass one perplexity. In a perfect model, the ratio would be zero, as the model would have enough abstraction to perfectly predict all tokens, while in a model that is not abstracting at all it would simply remain as one.

3.2 The Devil's Bargain Hypothesis

We assert that the primary driver in the **Devil's Bargain Hypothesis** is that blindly increasing the scale of our models has heavily incentivized them to lean into static memorization as a primary cognitive mechanism; Perversely, this has created a 'Devils Bargain' in which increases in metric performance due to better memorization are conflated with better computational, long-sequence, and transfer-learning abilities when the opposite is in fact occurring; This is the 'Devil's Bargain' in the hypothesis, and the tempting rewards that have caused the field to ignore strong cracks which have been showing more and more.

In essence, we diagnose almost all of the foundational economic and structural issues with our modern LLMs to be due to one core factor: **Perverse incentives against abstraction of memory and computation behaviors**. These have been masked due to effects wherein *in-distribution* tasks can be brute-forced by the model using its extensive trivia banks; Nonetheless effects such as poor out-of-distribution performance, worsening transfer learning, localized consideration of long-sequence tokens, and many other effects are predictions of the underlying mechanism and paint a stark picture: **Blind scale** will never generalize effectively, and scaling up needs to consider how to address core intelligence pressures.

3.3 Ineffective Time-Mixing Incentives and Modeling

The term "Time-Mixing" was introduced in the RWKV paper to describe how attention-like layers move information between token timesteps. We argue that time-mixing has a fundamental scaling ceiling—caused by poor training incentives and structural limitations. As such, the assumption that blindly scaling sequence lengths leads to better performance must be discarded entirely. Instead, new theory must be adopted that directly addresses the computational realities involved.

3.3.1 What are the Time-Mixing Issues

Limitations in model scaling across timesteps are well known. For example, Liu's *Lost in the Middle* observes that long-sequence models tend to focus only on local and global dependencies, ignoring midrange relationships. Additionally, while new long-context models like RWKV, Lightning2, and advanced forms of RoPE now allow training over increasingly extreme sequence lengths, the resulting performance improvements have often been underwhelming. Even these models continue to make basic errors on midrange connections. [19, 24, 29, 25].

We believe this limitation is structural. Current training incentives favor predictions based on locally relevant tokens, and existing architectures suffer from core deficiencies in handling long sequences. Without extensions like recurrent mechanisms—such as summarization-reinjection or chain-of-logic reasoning—these models cannot effectively respond to arbitrary information over long contexts, regardless of how desirable that behavior might be.

3.3.2 What is the core Mechanism

With the notable exception of time-recurrent research initiatives like RWKV and Lightning2, nearly all modern NLP architectures rely on relative embeddings—or variants like RoPE—which have the effect of degrading responsiveness over long sequences through a learned decay schedule. While these embeddings are trainable through various strategies, we argue they form a core part of the problem. More broadly, both training incentives and architectural design exhibit fundamental flaws.

As previously discussed, existing metrics fail to capture the effects holding back Generative NLP—primarily due to the washout of meaningful signals. This washout is driven by an overemphasis on local tokens. Under the assumption of *blind scaling*, we have believed that feeding models longer sequences naturally promotes generalization. But given what we now know, will it really?

The effects we bring up indicate this is not true. Consider attending between token i and j over long distances using one of these models. With very few exceptions, existing models force a connection rate based on $i-j$ that influences decay, rotation, or some other parameter. While this allows relative encoding, it is also the problem itself; there is no way for the model to decide to not forget context. This puts a cap on the model's addressable length. Additionally, we predict that the same perverse incentives that are masking our metrics are also influencing our training: Given the limited connectivity shown in metrics, models should then prioritize short-term connections since those can be most effectively predicted.

This is exactly what is observed. Liu, in Lost in the Middle, demonstrated that models primarily focus on local tokens and global anchors—while midrange dependencies are systematically ignored. We now diagnose this failure as resulting from two core factors:

- **Perverse Training Incentives**: Current training incentives are misaligned for long-sequence operations due to emphasis on local syntactic connectivity in a manner that does not encourage retention of information.

- **Unsuitability of Encoding**: Architectures that force expiration of information by timestep delta rather than recurrently defined forget directives are unsuitable for building models that perform Lifelong Learning.

Fixing this requires revised training incentives, and a commitment to recurrent models. The model has to be able to decide how much to forget in order to avoid forcing expiration of old timestep data.

3.3.3 Correcting Perverse Training Incentives

Have you ever known a student who pays attention in class but remembers none of it afterward? We argue that modern language models, in dynamic knowledge domains, are doing exactly that: they *read everything and remember nothing.*

As we established earlier, core metrics like perplexity are derived from faulty loss functions. This is what led us to say the field is "high on its own loss function." These flawed signals also distort training incentives. Models learn to see only the trees, not the forest—they lack any explicit reason to retain or synthesize information beyond the immediate prediction window.

To correct this, we need training mechanisms that are *Economic*, *Simple*, and *Effective.*

Background Several techniques have been explored to better align training incentives. A well-known class is *Curriculum Learning*, which introduces samples in order of increasing difficulty [3]. Variants like Intra-Batch Curriculum Learning have been applied in NLP [33], but these require curated datasets—making them infeasible for pretraining corpora like Pile.

Other methods adjust the loss function or reweight tokens. *Focal Loss* emphasizes harder examples by down-weighting easy ones [17]. Lin et al.'s *Not All Tokens Are What You Need for Pretraining* highlights difficult tokens using an auxiliary model and upweights them [18].

All of these, however, fall short at scale:

- Curriculum approaches do not scale cleanly.

- Intrinsic losses assume reliable internal difficulty estimation.

- External methods require costly auxiliary infrastructure.

We propose a more scalable and practical approach. It is so simple that, in retrospect, it feels obvious. We propose **Replay Pretraining**

Replay Pretraining **Replay Pretraining** is a drop-in training modification that introduces long-term memory abstraction pressures without requiring new data or complex infrastructure:

- It is scalable and integrates directly into existing training loops.
- It requires no changes to data format, retaining complete compatibility with Big Data.
- It saturates the model's time-mixing capacity, encouraging abstraction.
- It enables an implicit curriculum via scheduled sequence-length growth.
- It avoids dead gradient zones even in recurrent models.
- It adds only a $\sim 2\times$ compute and memory cost (with linear attention).

The approach consists of two changes:

- The model is trained on a duplicated sequence: `[BOS] tokens... [SEP] tokens... [EOS]`, forcing it to predict each token twice.
- Sequence length is gradually increased over training, eventually saturating memory layers. This forces abstraction.

This creates a *compressive incentive*: the model can improve loss by abstracting information from the first pass and using it in the second. While traditional loss functions implicitly allow this, Replay Pretraining ensures usable gradients by progressively increasing complexity. In essence, we take the insight that Compression is Intelligence and turn it into a training exercise.

Falsifiability and Experimental Predictions Replay Pretraining provides clear predictions and acts as a falsifiability test for the **Devil's Bargain Hypothesis (DBH)**:

- If *unscheduled* two-pass pretraining worsens performance on long-context tasks, DBH is falsified. Models should benefit from learning to expect and handle long-range connections.
- If *scheduled* two-pass training fails to improve reasoning or generalization after convergence, DBH is falsified. Saturation should force abstraction.
- If Replay Pretraining improves generalization—especially in recurrent models like RWKV—that supports DBH. These architectures should respond best to long-distance memory incentives.

While we cannot perform large-scale training experiments ourselves, the predictions are clear, testable, and grounded in economic and architectural reasoning. We encourage the community to evaluate them on their merits.

3.3.4 Correcting Time-Mixing Deficiencies

Modern language models are still built on the assumption that time is something to be endured, not understood. We train them to ingest long sequences, but give them no real mechanism to retain, prioritize, or abstract information across those sequences. The result is a pattern we now recognize: *read everything, remember nothing.*

This is not a failure of scale. It is a failure of structure. The root problem is that our models are forced to forget. Every step forward in time is also a step away from what came before. And no amount of scaling fixes a memory leak baked into the architecture itself.

Time as a First-Class Axis Transformers have succeeded by treating time as flat—just another axis over which attention distributes weight. But this view collapses at long range. Standard positional encodings enforce locality; attention decays with distance; layers run out of depth. Eventually, the model forgets—not because it wants to, but because it has no choice.

A genuinely intelligent system must be allowed to manage time. It must be able to store information until it is needed, discard it when it is not, and build *on top of* the existing information structures that span arbitrary temporal distance. That cannot be done with RoPE and attention alone. It requires recurrence—not as a gimmick, but as a fundamental epistemic structure. Otherwise the fundamental tradeoff of long sequence models, in which timesteps force forgetting, will never be mitigated.

State Space Models (SSMs), despite their mathematical elegance, are not immune. We attempted to adapt their structure to support dynamic forgetting, but found that doing so requires raising the recurrence matrix to a controllable power—breaking the very kernel tricks that make SSMs efficient. The architecture resists control. Others are welcome to try.

Mechanisms and Limitations Attention, in its current architectural form, is not a general-purpose mechanism for information flow—it is a carefully balanced compromise between expressivity, computational tractability, and locality bias. While we celebrate its ability to condition on all prior tokens, this conditioning is shallow. It does not store; it references.

Each transformer layer provides one opportunity for *time-mixing*, followed by one opportunity for *feature transformation.* But these time-mixing steps—implemented through attention—depend on positional encodings to distinguish where in the sequence information originates. This is where the failure begins.

Early transformer designs used absolute positional encodings, which work only over short sequences. Modern systems instead rely on *relative* encodings, which decay the influence of distant tokens according to a learned or fixed schedule. This decay is not an incidental artifact—it is a **requirement** for performance on current hardware. Without it, the model is overwhelmed by distant, low-relevance noise.

But this same decay imposes a strict limit on how far information can meaningfully travel. If each layer attends only within a local window of size L, then an N-layer transformer can propagate information roughly $N \times L$ tokens before that information fades into irrelevance. In practice, most sequences saturate this budget long before reaching useful abstraction. There is no mechanism to preserve facts indefinitely. There is no memory.

This puts the model in a bind. Either it must re-encode long-range information repeatedly, layer by layer, wasting capacity on redundant reconstruction—or it must give up on long-term reasoning altogether. Both outcomes are visible in today's models, in the results of Liu's research or in the success of the Chain-of-Reasoning tack on allowing more shots at the same process.

Thus, despite the marketing, attention is not a universal reasoning mechanism. It is a fast local router, not a memory substrate. It cannot, on its own, form the kind of persistent, dynamic internal state that would qualify as a knowledge base. This is not a flaw in implementation. It is a limitation of design: Any architecture that does not give the model control over when to forget information will have this issue, and such effects are unavoidably recurrent.

We, as always, would not like to undersell the existing scope of research. Linear attention variants and recurrent-inspired models like RWKV and RetNet attempt to side-step this by introducing time-aware decay or stateful recurrence. But these remain patches, not paradigm shifts. A full shift to recurrence as a first-class citizen is needed in order to treat time properly, in a manner that does not cause the trauma of the LSTM/GRU era.

Time-Parallel Recurrence We propose the use of **prefix-scan-based recurrence** as the mechanism to restore temporal abstraction in generative models. The core idea is simple: instead of computing memory updates sequentially, we represent them as cumulative operations over time.

This is not exotic. Prefix scans—especially the cumulative sum (cumsum)—are well-optimized primitives with efficient GPU implementations. They are associative, differentiable, and fast, with $O(\log T)$ time complexity. There are known implementations in raw CUDA, though sadly the PyTorch implementation is currently a linear scan. [10] They can form the mathematical backbone of a new class of memory systems that can be dropped directly into the attention slot of modern architectures.

The Shift to Phase Space To illustrate this idea, we examine the recurrent version of the *Phase-Decay-Update* Memory. This is introduced in full cumsum implementation glory over in the *Cognition Engineering* companion paper. This algorithm incorporates complex-valued memory, with phase shifts used to encode time explicitly. The resulting update rule:

$$U_t = U_{t-1} \cdot e^{-(\epsilon + i\phi)\cdot d_t} + u_t$$

Is incredibly simple, even in its sequential form. It is now inherently possible to query across time. And excess decay, though an issue, can be managed by shifting most of the burden to phase space: Numerics will not kill this in the full implementation. With outputs read via phase-aware queries:

$$o_t = \mathrm{Re}\left(q_t \cdot e^{i\gamma_t} \cdot U_t\right)$$

This simple extension solves the core issues:

- **Decay becomes phase-safe**: phase rotation never overflows.
- **Time becomes addressable**: the model can look forward or backward.
- **Structure becomes emergent**: information becomes retrievable not just by location, but by timing.

The Architecture Already Exists This is not science fiction. The RWKV architecture already implements adjustable recurrence using learnable decay. RetNet explores similar structures. State-space models like Mamba are inching toward the same principle. [24, 30]

But they are not unified. They are scattered research threads, each built in isolation, with varying complexity and limited interoperability. What we propose is to recognize the pattern, unify the form, and design with memory and time as a first-class recurrent axis—epistemologically, architecturally, and economically.

A Note on Origin The original seed of the PDU algorithm came from an unusual source. In early experimentation, a conversation with ChatGPT-4—lacking access to RWKV's full architecture—hallucinated a decay-update memory built on cumsums. It was wrong, numerically unstable, and incomplete. But it was close.

From that seed, with just a few mathematical refinements, came a mechanism that we believe stands as one of the simplest, most expressive memory systems available. The idea is simple enough for a current LLM to develop. It was only the right framing that was lacking.

Falsifiability We predict that recurrent memory systems, especially those built with phase-decay semantics, will show high synergy with Replay Pretraining. We expect:

- Lifelong memory to emerge naturally with learned decay.
- Better generalization to long-range dependencies.
- Efficient implementation in both batch and streaming modes.

This is not a speculative bet. It is a structural claim: that when models are allowed to manage time and accumulate knowledge abstraction follows.

3.4 Channel Mixing Failures

Channel Mixing is a term that is adopted within the RWKV paper to refer to implementations that fit within the "Feedforward" slot of a Vaswani-Inspired derivative of the transformer. We adopt this notation here.

We argue that the prevailing paradigm in Channel-Mixing—particularly the push toward ever-larger Mixture-of-Experts (MoE) configurations—is not merely inefficient, but structurally misaligned with the goals of general intelligence. Rather than promoting abstraction, these architectures incentivize brute-force memorization through unconsolidated parameter expansion. We show that modern architectures are structurally incentivized to favor mass *memorization* over the consolidation of *general computational patterns*—a direct consequence of unconstrained parameter growth and insufficient consolidation pressure.

We will trace these down to falsifiable mechanisms with empirical evidence that are present in feedforward layers, and see how blind parameter scaling has been actively working against us.

3.4.1 Channel-Mixing Layers

At it's most basic level, channel mixing is a method of mapping vectors. Let f be the channel-mixing operation. Let this accept the input $\vec{x}$ and map it into output $\vec{y}$.

$$\vec{y} = f(\vec{x})$$

All Channel-Mixing layers, whether feedforward based, a Mixture of Experts, or something more clever like token shifting in RWKV, nonetheless share this basic computational core, due in large part to the fact that a two layer perceptron is a weighted sum of the vectors making up the dense output projection.

The model can use these maps in a variety of ways

- Make a decision tree to follow based on attention context gather steps.
- Inject memorized context.
- Perform Synthesis of multiple factors.

However, this occurring relies on having the right training pressures. We contend we do not, and modern LLM's are being trained in a way that results in operation in primarily mode one and two. Synthesis ends up neglected in large part, and this ends up leading to a decision-tree explosion that dilutes training across trillion of nodes, leading directly to most of the issues the current generation of architecture suffer from.

3.4.2 Mechanism

The core failure we identify is structural: generalization requires parameter contention—but parameter contention, under current architectures, encourages fragmentation into brittle, overfit edge cases.

Gradient descent is greedy. It resolves problems locally, updating whichever parameters yield the fastest loss reduction. In a convex loss landscape, this would suffice: every update would move the model toward a global minimum. But in practice, we operate on a highly non-convex surface. This means gradient descent often finds locally useful solutions that fail to generalize—and then gets stuck there.

We argue that, during training, models form what are effectively dynamic computational graphs. The feedforward pathway can be viewed as a superposition of vector "nodes," shaped by attention into specific computational chains. These chains form priors over time—some dovetailing or converging, but many remaining specialized and disjoint.

Here lies the trap: as training progresses, it is almost always easier for the model to allocate new parameters to memorize novel edge cases than to integrate those cases into existing abstractions. Integration requires navigating parameter contention—reusing parts of the model already entangled with other tasks—while memorization via free, unconstrained parameters is cheap and immediate. In terms of local loss minimization, patches win over synthesis.

Over time, this leads to a pathological architecture: trillions of memorized edge cases encoded across deeply branched computational chains. Like a legacy codebase riddled with quick-fix hacks, the model becomes increasingly brittle. Small contextual shifts can activate the wrong subgraph—or none at all. In the limit, the model becomes a decision tree of staggering depth, shallow generality, and exponential leaf-node complexity.

This behavior might have been tolerable if our training loss faithfully optimized for generality. But it does not. As we have shown, the loss function rewards token-level prediction accuracy, not abstraction or synthesis. Without sufficient consolidation pressures, redundant pathways are never collapsed. They multiply. The deeper the tree grows, the more data and compute are required to train it—and the more fragile every downstream intervention becomes.

Alignment, reasoning, even basic task consistency—these are not separate systems layered atop a stable foundation. They are leaves on the tree. And once you see this, the picture changes: every time we fine-tune a model without consolidation, we risk severing the very subgraphs that hold those high-level abilities together. After all, are not these usually the last things added, and thus furthest out in the decision tree?

Thus, we conclude: unconstrained parameter growth in channel-mixing layers incentivizes specialization over synthesis, memorization over generalization, and architectural brittleness over scalable intelligence. This is not just inefficient—it is actively corrosive to alignment and reasoning, and thus to the long-term viability of Large Language Models themselves.

3.4.3 What about Regularization?

Regularization should, in theory, help mitigate the fragmentation problem. In practice, it has been misapplied—particularly in the case of feedforward channel-mixing layers.

The correct approach, we argue, would be to apply dropout to the *hidden state* within the feedforward block. This forces the model to explore alternative computational pathways, since its favored activations may be randomly suppressed during training. It is, in effect, a pressure to diversify the graph.

Instead, most transformer derivatives—including the original Vaswani formulation—apply dropout only to the *output* of the feedforward computation. This introduces stochastic noise, but does not disrupt the underlying decision structure. The model simply learns to spread critical information across more dimensions to compensate—widening redundancy, not encouraging abstraction. Even load balancing in Mixture-of-Experts models fails to help here, as it operates on average usage statistics, not pathway consolidation.

Even if we correct this oversight and apply dropout at the decision points, the problem does not go away. With enough parameters, the model will simply encode sufficient redundant mappings that it always has some available path to the correct output. Regularization slows the problem—it does not stop it. Eventually, memorization wins again.

One could attempt to combat this by restricting the model's storage capacity more directly—for example, by applying L1 or L2 penalties to the projection weights in channel-mixing layers. This would limit the model's ability to replicate the same computation across many redundant vectors. But this, too, has costs. A heavily regularized model becomes less capable of storing rare facts, or of creating backup pathways when context is noisy.

In summary: regularization, though valuable, is ultimately a negative pressure. It can slow the collapse into brittle memorization, but it cannot reverse it. Without complementary architectural incentives—such as parameter sharing or abstraction-aware objectives—regularization alone is insufficient to achieve generalization.

3.4.4 Diagnosis and Directions of Solution

From the preceding analysis, we conclude that the current generation of models faces not just challenges in scaling across time, but foundational architectural limitations that compromise their ability to generalize—even within their training distribution. These are not merely issues of tomorrow's scale; they are breaking things today.

To address this, we propose two orthogonal but essential forces within the **Computational Abstraction** design axis:

- **Exploration pressures**: Model components—particularly selection maps and gating mechanisms—should be encouraged to escape their local minima and explore alternative pathways. This can be introduced via structured regularization, dropout applied at the decision level, or through explicit perturbations of selection priors during training.

- **Consolidation pressures**: When solving a new problem, the model should be biased to reuse existing structures—shared vector maps, expert modules, or computational routines—rather than allocating fresh parameters. This can be encouraged through parameter sharing, recurrent use of common sublayers, or objective functions that favor low-switching reuse across tasks. The key is to make "I already have an expert for this" a locally optimal move in gradient descent.

Together, these forces act as an architectural form of Hebbian learning: promoting computational pathways that are both reused and relevant, while pruning away brittle specialization. If successful, they enable the emergence of a robust internal connectome—capable not just of memory, but of synthesis.

An example of a drop-in layer archetype with this philosophy in practice is provided in the *Smart-MoE* layer introduced in the companion engineering paper.

3.4.5 Predictions and Literature Evidence

There are a large number of predictions we can make that are substantiated in the existing evidence across a wide variety of fields.

Models should be getting exponentially more costly to train The breakage of the Big Data Scaling paradigm is a predicted consequence. Without more data to force convergence or extra exploration, our models will need increasingly large amount of training as they get larger in order to code for all of the edge cases. Since edge case growth is exponential, so too should be our costs.

This is perhaps one of the most straightforward predictions, and is extremely well known. Our models are getting exponentially more expensive, in ways that are not maintaining linear growth, particularly when alignment and reasoning are concerned. [13]

Parameter Sharing Should Improve Performance We predict that Parameter Sharing of Channel-Mixing should improve performance/parameter and generalization by tilting the training incentives more in favor of reusing existing general patterns.

In particular the mathematical mechanism is that sharing encourages the model to centralize frequent computations into one location; since the model gets multiple passes there is a large amount of parameter contention forcing them to become very general. However, in being general, they become an idea point for starting a computation chain: it now becomes the case the model will tend to maintain shorter chains and frequently process information using the shared maps and continue to have dependencies on the results.

By predictably providing information all computation chains will come to depend on the shared results, and this means gradient pathways will never completely shut down. As such, parameter sharing vector maps is an efficient way using existing technology to address the issue and provides the positive consolidation pressure we are missing.

Is this evident in the literature? Absolutely.

- ALBERT explored parameter sharing as a way to compress BERT down, and also found it improved performance [15]. The performance increase was not deeply discussed, but in light of this mechanism is very telling.
- DeepSeek-MoE is one of the first known commercial instance of a MoE using expert sharing. This is conjectured to have contributed significantly to the ease of training and success: It would be interesting to see an ablation with and without the sharing.

More generally, parameter sharing is a well-established subfield in compression: However it's full power may in fact be it is performing compression by increasing generalization.

Distilled Models should sometimes perform better on novel reasoning tasks than the base model By virtue of having less parameters, a distilled model is forced to be more general with them. Models distilled down to a smaller, more compact core must thereby rely on reusable vector maps rather than brute-force memorization.

We would predict that a distillation of a model will have **better** performance than the original on *reasoning* tasks than the original model so long as the distillation process enforces dense coupling to the teacher's reasoning process, requiring the student to replicate the same steps with fewer parameters - an incentive that forces abstraction and consolidation.

Again, this is displayed in the literature, though it is somewhat hard to see in some cases:

- DistillBERT performed slightly on almost all tasks than the base BERT models, and only predicted the logits of the teacher model. This would seem to argue against the prediction. However, we would say it is actually the opposite: By just predicting the logits, the model was given sufficient degrees of freedom to create sharded vector maps again. [27]
- In contrast another BERT distillation - MiniLM -operated by forcing the model to perform a level of abstraction consolidation.

  MiniLM trained the student to predict intermediate attention distributions and hidden states from the teacher, encouraging it to reconstruct the reasoning chain. This forced it into an underparameterized regime where mimicking the teacher exactly was impossible, thereby encouraging generalization and abstraction so as to use to maximum effect what little it had. As a consequence, the student had to adopt an increased level of abstraction and generalization.

The result was an improvement on some GLUE benchmarks over the base model. This should not have happened. [32]

It is difficult to examine these facts without concluding that modern LLM architectures are leaving significant abstraction consolidation on the table. This appears to represent solid evidence in support of the hypothesis.

Fine-Tuning Irreversibly Orphans Vector Mappings; Severity Increases with Model Scale The process of "Fine-Tuning" a LLM for a particular application is a foundational tool in the modern NLP worker's arsenal. In it, a foundation model is fine-tuned for a particular application to take advantage of transfer learning. However, such a task must be performed with care, as the well-known specter of "catastrophic forgetting" can occur where the transfer process destroys too much information for the model to remain viable. Fine tuning comes in two main flavors: Full-Parameter, and Partial-Parameter. We focus here on the former, in which all model parameters are involved in fine tuning; In contrast, in the latter most of the layers are frozen.

We predict that in full-parameter fine tuning the severity of catastrophic forgetting will get larger superlinearly in association with the model size as this is a natural consequence of the Devil's Bargain hypothesis. Larger models are incentivized to weakly memorize a large number of edge cases in contrast to the smaller models that must train a smaller number of parameters more generally. As such, it is very natural that stochastic training processes may result in nonlinear activations "orphaning" a large proportion of those weakly-accessed vector mappings when fine-tuning is applied to the model.

This is exactly what is observed. In *An Empirical Study of Catastrophic Forgetting in Large Language Models During Continual Fine-tuning* by Luo et al. the authors perform a fine tuning process on a variety of models between 1-7 billion parameters. They acknowledged that "surprisingly" the severity of Catastrophic Forgetting increases as model size increases; In contrast to the BDS paradigm this is exactly what is expected if the Devil's Bargain Hypothesis is valid.[20]

As an additional natural extension with predictable consequence, we also note there is a simple way to measure this orphaning effect, though no study appears to have tried it.

1. Start with a pretrained model
2. Fine-tune it on some task using whole-parameter training
3. Select a token count from the original pretraining data source equal to that used in fine-tuning
4. Fine-Tune the model using this pretraining data.

5. Measure the difference in benchmark performance between the initial and final model.

This provides an easily falsifiable experiment to immediately test for the effects being predicted and is also an invaluable tool for detecting how much forgetting is occurring in your pretraining tasks to avoid any nasty deployment surprises.

The fulfillment of this prediction in such a direct format is directly predicted as a consequence of the Devils Bargain Hypothesis and addresses anomalies the current design paradigm cannot otherwise explain. As such, it provides additional evidence building towards a picture where the Devil's bargain hypothesis must be accepted, and thus the current Big Data Scaling hypothesis rejected.

Interestingly enough, another consequence is that locking our layers and using additional parameters to train as in LoRA should avoid this orphaning effect.

3.5 Rejection of a Paradigm

If the Devil's Bargain Hypothesis holds—and by now the evidence is overwhelming—it compels us to abandon one of the core assumptions of modern NLP: **blind scaling**. Scaling alone will not produce economical, generalizable, or cognitively robust artificial intelligence. Instead, the unchecked expansion of parameters, data, and compute has quietly but systematically distorted incentives—promoting memorization over abstraction, brute force over structure.

This is not a benign inefficiency. It is structural misalignment. In truth, we have not been succeeding *because* of our architectures, but *in spite of them.* Only the magnitude of Big Data ever allowed this paradigm to function at all. And even at its peak, it was brittle, opaque, and economically unsustainable. Now, with high-quality data exhausted and compute costs accelerating, we find ourselves at a dead end: larger models cost exponentially more while becoming increasingly fragile; transfer learning degrades at scale; and true generalization remains elusive.

Transformers did not win because they were optimal. They won because they were the first architectures that didn't break when scaled. But "not breaking" is no longer enough. The limitations of the Big Data Scaling paradigm are no longer theoretical—they are operational. To cling to it now is to build bigger engines for a car with no steering.

It is embarrassing, but necessary, to state the obvious: **we have been overfitting our models**. Catastrophically. Systematically. At scale.

In hindsight, the signs were always there. We scaled parameters exponentially. We ran out of novel data. We trained without introducing new constraints, priors, or incentives for abstraction. To any undergraduate statistician, the outcome would have been obvious:

> *You increased the degrees of freedom without increasing the information. What did you think would happen?*

We mistook memorization for intelligence. We confused benchmark gains for understanding. We believed our loss functions, without asking what they were actually optimizing. And we convinced ourselves—against every principle of statistical learning—that this time was different.

It wasn't.

The Devil's Bargain wasn't subtle. We just chose not to look too closely, because it worked—for a while. But it was never robust. Never general. Never sustainable.

We must now reject not only blind scaling, but the epistemology that allowed it to thrive. Machine learning has for too long treated empirical results as the only form of legitimate science. But theory—when it explains, predicts, and diagnoses—*is* science. If we had asked the right theoretical questions earlier, we would have seen this coming.

And so we conclude: it is time to move beyond blind scaling. We need not more data or more scale, but *better statistical priors*: models designed for intelligence. It is time to rebuild our architectures—not on data alone, but on principles that reward generalization, compression, and computational abstraction. It is time to remember that intelligence is not just something we can mine—it is something we must understand.

4 An End, and a New Beginning

From this point forward, we shall proceed under new assumptions.

The **Blind Scaling Prior** is dead. So too is the epistemological stance that only empirical results define scientific merit. Both have delivered results—and revealed their limits. What comes next must be built on firmer ground: on structure, on principle, and on the explicit goal of modeling cognition.

To that end, we introduce a new systems discipline: **Cognition Engineering**.

Cognition Engineering is not just another subfield of machine learning. It is a full-stack framework for designing intelligent systems from first principles. It draws heavily from physics, statistics, cognitive theory, and systems design—and is engineered to remain compatible with the Vaswani-style transformer stack. It is part philosophy, part engineering, and its central concern is simple: *engineering, backed with theory.*

This is not an algorithmic discipline. It is a systems discipline.

- **First, identify the core objectives**: What must a general system actually be capable of?
- **Then, identify the necessary abstractions**: What roles must be filled—memory, synthesis, reasoning, control?
- **Then, build the minimal viable archetype**: What is the simplest design that can satisfy all requirements?

- **Finally, rely on synergy**: Trust that the interaction between simple, principled parts can produce emergent behavior—if the abstractions are right.

The rule of thumb is simple:

> *If you cannot explain the philosophy behind a component in three sentences, it is too complex for that abstraction.*

This approach demands rigor not just in results, but in design. Every mechanism should be explainable. Every component should have a clear epistemic function. Every abstraction should pull its weight—not through brute force, but through its role in a coherent system.

Cognition Engineering is already underway. The companion works detail the first wave of Minimal Viable Archetypes, and set out the broader philosophical foundations. The next in the series is *Cognition Engineering: Post-Scaling Engineering with Composable Minimal Viable Archetypes.* It touches every part of the Vaswani stack, and is designed such that your lab can try out as many—or as few—as you like. But remember: they're built to synergize.

The complete rethink of research philosophy, including how an independent researcher could arrive here in the first place, is explored in *Philosophy of Engineering and Science in Cognition Engineering: Operating in a Post-Deep Learning World.*

This is not the end of the conversation. It is the start of a new one.

We invite theorists, engineers, and skeptics alike to explore this space with us. It is time to stop merely building bigger tools—and start building the right ones.

Welcome to **Cognition Engineering** and thank you for witnessing my descent into madness.

A Testable Predictions

This work makes strong theoretical claims—but theory without falsifiability is not science. As such, we include this appendix to clearly enumerate a set of concrete, testable predictions derived from the core hypothesis of this paper: the Devil's Bargain Hypothesis (DBH). These predictions are designed to be practical, experimentally approachable, and falsifiable. Many require only modest computational resources and are amenable to open research.

Researchers looking for publishable directions, empirical starting points, or diagnostic tools for current architectures are encouraged to draw from this list. We welcome external reproduction efforts—and would consider falsification a contribution to the field. A healthy theory is one that survives direct confrontation with data.

Two-Pass Perplexity Ratio

:

The following is the predictions and procedures which can be used to test the **Two Pass Perplexity Ratio** intelligence correlation. Is better abstraction performance correlated with a higher TPPR?

Claim: More capable reasoning models should generally have a better two-pass perplexity score than less capable ones.

Reason: They are better at abstracting the hard-to-remember pieces of context, and thus better at predicting them in the future when they know it will come up again.

Mechanism: Two pass perplexity is done by priming the model to expect the same sequence twice, then feeding in text [Sep] text. Take the ratio of the per-token perplexity during the second to first pass.

Reference: See section 3.1.2. Literature support is primarily Liu's "Lost in the middle" [19]

Procedure: Get a collection of sample texts of varying lengths, and a collection of models. Over various sequence lengths, test the TPPR and correlate the TPPR scores with various metrics, and particularly reasoning ones.

Replay Pretraining Improves General Performance

:

The following is the predictions and procedures which can be used to test whether Replay Pretraining does indeed improve performance.

Claim: Most models will see much better pretraining generalization results when trained using Replay Pretraining. Because of bad loss incentives, models trained using Replay Pretraining will generalize better at the same level of loss than one trained without it.

Reason: Abstraction Distillation pressures are induced during pretraining, providing long-range gradient signals motivating the model to better distill key knowledge into long-range abstractions.

Mechanism: Replay pretraining is done by exposing the model to [SOS] text... [SEP] text ... [EOS] during pretraining, with text being your pretraining source such as Pile. A Schedule should optionally be used to increase the truncation length and thus increase the distillation pressure gradually.

References: See section 3.3.3. It contains falsifiable predictions to. Literature support is primarily Liu's "Lost in the middle" [19]

Procedure: Get a collection of sample texts of varying lengths, and a collection of models. Run pretraining with and without Replay. See if at a given level of loss the models have dissimilar generalization performance. Test against TPPR too. If budget allows, test final performance too against similar models.

A.1 Replay Pretraining Drastically Improves Long-Sequence Performance

:

Replay Pretraining is designed to provide much better signals for models to make forget/retain/compress decisions on. This should provide stronger incentives to set a better midrange signals.

Claim: Long sequence models should be able to better track overall context when trained using scheduled Replay Pretraining. This effect will only show up on models with trainable forget rates or related proxies such as RoPE rotation amount.

Reason: Abstraction Distillation pressures are induced during pretraining due to ensuring all tokens are used again, providing long-range gradient signals motivating the model to better encode the hard-to-predict details. Those details are exactly the ones that matter over long distances.

Mechanism: Replay pretraining is done by exposing the model to [SOS] text... [SEP] text ... [EOS] during pretraining, with text being your pretraining source such as Pile. A Schedule should optionally be used to increase the truncation length and thus increase the distillation pressure gradually.

References: See section 3.3.3. It contains falsifiable predictions to. Literature support is primarily Liu's "Lost in the middle" [19]

Procedure: Get a collection of sample texts of varying lengths, and a collection of models. Run pretraining with and without Replay. See if at a given level of loss the models have dissimilar generalization performance. Test against TPPR too. If budget allows, test final performance too against similar models.

Replay Pretraining Drastically Improves LSTM/GRU/RWKV performance

:

Replay Pretraining is designed to provide much better signals for models to make forget/retain/compress decisions on. This should show up on models capable of making these decisions.

Claim: Recurrent models capable of making forget decisions based on data should see extremely high synergy with Recurrent models, particularly those that control their own forget rates.

Reason: Abstraction Distillation pressures are induced during pretraining, providing long-range gradient signals motivating the model to better distill key knowledge into long-range abstractions.

Mechanism: Replay pretraining is done by exposing the model to [SOS] text... [SEP] text ... [EOS] during pretraining, with text being your pretraining source such as Pile. A Schedule should optionally be used to increase the truncation length and thus increase the distillation pressure gradually.

References: See section 3.3.3. It contains falsifiable predictions to. Literature support is primarily Liu's "Lost in the middle" [19]

Procedure: Get a LSTM/GRU, and train it with and without Replay Pretraining. See if the extrapolation issues that killed the technology still exist. If ambitious, a full RWKV pretraining run using Replay Pretraining is predicted to drastically increase performance.

Parameter Sharing reduces Catastrophic Forgetting

:

Parameter sharing should make the models more robust and thus avoid catastrophic forgetting that can occur in fine tuning.

Claim: Models using Parameter sharing should be more robust to catastrophic forgetting. This should be measurable in terms of the performance loss in fine tuning.

Reason: By providing a centralized repository of "Utilities" the model can fall back on the model is encouraged to reuse an existing tool rather than make a new one.

Mechanism: The DBH says that computation is, internally, resolving into a "Decision-Tree" with billions of weak nodes. This is due to perverse incentives. By providing a library of common tools the model is motivated to make a bunch of short trees with routing decisions at the joints rather than one long one, limiting how deep nodes can be. This makes it harder for fine tuning to orphan nodes.

References: See section 3.4.2 for a full discussion of the mechanism. Corolated literature predictions are at 3.4.5

Procedure: Initialize a parameter-shared model with sharing in the feedforward slot; it is recommended to use the technology of DeepSeek-MoE for a large run, or just a shared feedforward layer for a small one. Pretrain a model with, and without, the sharing.
Now, fine tune both models for a task, and using an equivalent amount of pretraining data fine tune back to the pretrained distribution. The ratio between the performance before and after this process is your orphaning proxy, and should be lower in the parameter-shared model.

Distilling Models in an Underparameterized Regime Will Improve Reasoning To Above The Base

:

Distilling models in such a way that it has to predict intermediate states using less models will force them to learn in a way that may make them outperform their base models.

Claim: Distilling models in such a way that it has to predict intermediate states using less models will force it into an underparametarized regime. This can be observed in the GLUE scores occasionally

Reason: This causes abstraction pressure, which should improve reasoning performance. This should be seen in tasks that are highly reasoning focused - notably, it has already been observed in RTE.

Mechanism: The DBH says that computation is, internally, resolving into a "Decision-Tree" with billions of weak nodes. By distilling in an underparamerized regime we force the model to consolidate nodes to save space, improving generalization capacity. This will be reflected most strongly in the GLUE RTE score,

References: See section 3.4.2 for a full discussion of the mechanism. Corrolated literature predictions are at 3.4.5

Procedure: Distill a model from a base, and particularly a massive model that has a large Mixture of Experts backend. Make sure the student is being forced to predict intermediate embeddings to underparameterize it - MiniLM is a good example of how to do this.
Now check the GLUE scores. With a model of the right size, you should see that the distilled model outperforms the base in RTE. This has already been observed on MiniLM.

Regularization On Feedforward Will Improve Performance

:

The Decision Tree Explosion effect should be mitigatable to some extend with additional regularization. We should test that.

Claim: Models with additional regularization on the feedforward layers should outperform the control so long as you shut the regularization off part-way though pretraining then completely cover an epoch

Reason: This causes abstraction pressure by preventing the model from making arbitrary small decision tree nodes.

Mechanism: The DBH says that computation is, internally, resolving into a ”Decision-Tree” with billions of weak nodes.
By applying regularization we force the weak nodes to collapse, providing pressure that forces the model to abstract and merge signals together until strong enough to train.
However, this weakens memory. Turning it off partway through pretraining ensures that we know how the tree needs to be laid out for robustness, and the model can then slot the information into the right place

References: See section 3.4.2 for a full discussion of the mechanism. Corrolated literature predictions are at 3.4.5

Procedure: Take a model. Make four instances to pretrain, one control, one trained with dropout on the Feedforward hidden state, one with L1 or L2 on the feedforward parameters, one with both.
Now pretrain them to exhaustion. Finally, shut off the losses on the test cases and do a final epoch. Check the performance against your favorite benchmarks.

References

[1]

[2]

[3]

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[5] Marco C. Campi and Simone Garatti. Compression, generalization and learning, 2024.

[6] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha

Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. Palm: Scaling language modeling with pathways, 2022.

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018.

[8] Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling, 2020.

[9] Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. Realm: Retrieval-augmented language model pre-training, 2020.

[10] Mark Harris, Shubhabrata Sengupta, and John D. Owens. Parallel prefix sum (scan) with cuda. In Hubert Nguyen, editor, *GPU Gems 3*, chapter 39, pages 851–876. Addison-Wesley Professional, 2007.

[11] Jeremy Howard and Sebastian Ruder. Universal language model fine-tuning for text classification, 2018.

[12] Yuzhen Huang, Jinghan Zhang, Zifei Shan, and Junxian He. Compression represents intelligence linearly, 2024.

[13] Wall Street Journal. Openai delays gpt-5, working on ‘orion’ to reduce ai costs, 2024. Accessed: March 20, 2025.

[14] 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.

[15] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations, 2020.

[16] 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.

[17] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

[18] Zhenghao Lin, Zhibin Gou, Yeyun Gong, Xiao Liu, Yelong Shen, Ruochen Xu, Chen Lin, Yujiu Yang, Jian Jiao, Nan Duan, and Weizhu Chen. Rho-1: Not all tokens are what you need, 2025.

[19] Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts, 2023.

[20] Yun Luo, Zhen Yang, Fandong Meng, Yafu Li, Jie Zhou, and Yue Zhang. An empirical study of catastrophic forgetting in large language models during continual fine-tuning, 2025.

[21] Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert, 2020.

[22] 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.

[23] Bhargavi Paranjape, Scott Lundberg, Sameer Singh, Hannaneh Hajishirzi, Luke Zettlemoyer, and Marco Tulio Ribeiro. Art: Automatic multi-step reasoning and tool-use for large language models, 2023.

[24] Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Matteo Grella, Kranthi Kiran GV, Xuzheng He, Haowen Hou, Jiaju Lin, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartlomiej Koptyra, Hayden Lau, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Bolun Wang, Johan S. Wind, Stanislaw Wozniak, Ruichong Zhang, Zhenyuan Zhang, Qihang Zhao, Peng Zhou, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. Rwkv: Reinventing rnns for the transformer era, 2023.

[25] Zhen Qin, Weigao Sun, Dong Li, Xuyang Shen, Weixuan Sun, and Yiran Zhong. Lightning attention-2: A free lunch for handling unlimited sequence lengths in large language models, 2024.

[26] Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. Efficient content-based sparse attention with routing transformers, 2020.

[27] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2020.

[28] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.

[29] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023.

[30] Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. Retentive network: A successor to transformer for large language models, 2024.

[31] Unity Technologies. Unity muse: Unlock your creative potential with ai. `https://unity.com/products/muse`, 2024. Accessed: 2025-04-05.

[32] Wenhui Wang, Furu Wei, Li Dong, Hangbo Bao, Nan Yang, and Ming Zhou. Minilm: Deep self-attention distillation for task-agnostic compression of pre-trained transformers, 2020.

[33] Benfeng Xu, Licheng Zhang, Zhendong Mao, Quan Wang, Hongtao Xie, and Yongdong Zhang. Curriculum learning for natural language understanding. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6095–6104, Online, July 2020. Association for Computational Linguistics.

[34] Gokul Yenduri, Ramalingam M, Chemmalar Selvi G, Supriya Y, Gautam Srivastava, Praveen Kumar Reddy Maddikunta, Deepti Raj G, Rutvij H Jhaveri, Prabadevi B, Weizheng Wang, Athanasios V. Vasilakos, and Thippa Reddy Gadekallu. Generative pre-trained transformer: A comprehensive review on enabling technologies, potential applications, emerging challenges, and future directions, 2023.

[35] Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. Big bird: Transformers for longer sequences. *CoRR*, abs/2007.14062, 2020.

[36] Shengyu Zhang, Linfeng Dong, Xiaoya Li, Sen Zhang, Xiaofei Sun, Shuhe Wang, Jiwei Li, Runyi Hu, Tianwei Zhang, Fei Wu, and Guoyin Wang. Instruction tuning for large language models: A survey, 2024.