The NaaPS Model for Silicon Sampling

Working Paper v2025-01-22
Gabriel Simmons
gsimmons@ucdavis.edu

Keywords: Large Language Models, Persona, Prompting, Simulation, Simulators, Ontology, Taxonomy, Philosophy, Silicon Sampling, Three-Level Hypothesis

Abstract Silicon sampling is the emerging science of using large language models to simulate objects of sociological interest. The diversity of silicon sampling prompting strategies – as well as debates across the lines of philosophy, computer science, social science, and AI ethics – suggest widespread disagreement about what large language models are. This paper proposes an integrative view of large language models (the NaaPS model) that attempts to unify these disparate perspectives. This paper argues that silicon sampling practitioners should feel comfortable committing to the existence of *boundedly-realistic simulacra* that are realized by LLM-based systems, but they should also be ready to examine lower levels of abstraction to describe and expain when simulacra fail or to enhance their performance. This paper maps several layers of abstraction and reductions between them, sorts common prompting techniques by their level of abstraction, and provides a thought experiment to illustrate how boundedly-realistic simulacra can be realized by conditional probability distributions. The ambition of this paper is to show how more detailed consideration of the ontology of large language models can justify and improve silicon sampling practice.

1 Introduction

There is an emerging science of using large language models for simulation, including simulation of societies [1], demographics [2], [3], [4], and states [5]. Many of these studies use prompting techniques to induce desirable simulation responses from language models. At the moment, these prompts vary widely, with little standardization across practitioners or across tasks. Some prompts induce behavior through examples or by building a narrative. Some prompts address the model like a polling respondent or survey participant. Others address the model like a research assistant. Still others address the model as a role-player or simulator.

Def. Silicon Sampling
Using large language models as simulators of objects of sociological interest.

Term originates from [3]. Definition generalized from [3] and other works including [6], [7], [8]

This is not a paper about prompt engineering. I highlight the differences in prompts only to emphasize the following observation. Practitioners in the new science of simulating humans and human systems with language models (*silicon sampling*) take vastly different approaches in their apparent assumptions about what (or who) language models are. In other words, practitioners vary in their apparent ontological commitments.

```
Ideologically, I describe myself as liberal.
Politically, I am a strong Democrat. – [3]
```

```
"It is [YEAR]. You are a [AGE] year-old,
[MARST], [RACETH] [GENDER] with [EDUCATION]
making [INCOME] per year…" – [8]
```

```
Pretend you are a Democrat. – [6]
```

Figure 1: Examples of silicon sampling prompting techniques.

This reveals philosophical questions whose answers will directly inform silicon sampling practice. These questions include the following. The responses given in this paper are below each question.

1. How can I know when silicon sampling is working and when it isn't? What should I do when it isn't working?

   When simulacral-level (S-level) analysis isn't working, go deeper. Probabilistic, Algorithmic, and Numerical levels offer descriptive, explanatory, and prescriptive tools. See **Section 5**.

2. How should I refer to large language models? Is it ok to talk about ChatGPT as if it has beliefs or desires? Can I drop the "as if" and say plainly that Claude wants, knows, or believes something?

   Practitioners should feel comfortable committing to the existence of **boundedly-realistic simulacra (BRS)** that are realized by LLM-based systems. See **Section 4**.

3. I know there is more to learn about the inner workings of LLMs, but does any of that matter for validity and performance? I want to do the same social science I was doing before at a lower cost – why should I care about viewing an LLM as a neural network or an assembly of circuits?

   Probabilistic, Algorithmic, and Numerical levels offer descriptive, explanatory, and prescriptive tools. See **Section 5**.

4. How can I explain to others (students, other practitioners in my field, lay audiences) what ChatGPT is doing and why it might be a valid sociological instrument?

   LLM-based systems are NaaPS: Numerical, algorithmic, Probabilistic, Simulacral. See **Section 3** for a map of the NaaPS model. See **Section 5.4** for a simple thought experiment illustraing how boundedly-realistic simulacra can be realized by conditional probability distributions.

2 Background

2.1 Language Models

In this section, I will describe the basic features of large language models.[1]

Mathematically, language models model a conditional probability distribution over sequences of symbols. The conditional probability of the next symbol given the previous symbols can be expressed as $P(x_n \mid x_1, x_2, ..., x_{n-1})$ where x_i is the ith symbol in the sequence. Continuations of sequences can be generated by sampling a token from the conditional probability distribution $x_{i+1} \sim P(x_{i+1} \mid x_1, x_2, ..., x_i)$, appending it to the sequence, and repeating for the next token $x_{i+2} \sim P(x_{i+2} \mid x_1, x_2, ..., x_{i+1})$.

x (input)	y (output)
A sentence in English	Its French translation
A document	Its summary
A demographic descriptor and a multiple choice question	An answer, presumably from someone in the demographic

Table 1: A table of examples of sequence prediction inputs and outputs. Table layout and examples are from [9]

Modern approaches approximate this conditional probability distribution by training a neural network. Pre-training involves optimizing the (billions of) parameters of the neural network such that outputs of the network behave like a conditional probability distribution. This behavior is enforced by certain features of the neural network architecture[2] and training methods[3].

Today's large language models are one type of *foundation model* [10], a general-purpose model that can be adapted to downstream tasks. The use of foundation models generally follows a pattern known as *transfer learning*: models are first *pre-trained* on a large volume of general data (a relatively expensive process, often taking weeks or months), then *adapted* for specific tasks (relatively inexpensive, taking as little as minutes).

Adaptation occurs during one or more "post-training" stages, such as *instruction fine-tuning* or *reward model tuning*. A detailed description of these methods is beyond the scope of this paper. In essence, post-training steers models towards producing *useful* (rather than simply probabilistically likely) responses to instructions.

Importantly, post-training achieves its goals by making the useful thing (**S**-level) the likely thing (**P**-level), or by modifying the weights of the neural network (**N**-level).

The model is presented with a concentrated dataset of examples of helpful behavior such that helpful behavior is overwhelmingly likely.

Model capabilities can be expanded even further by imposing some scaffolding on top of the language model itself. These techniques include searching a tree of possible continuations [11], appending search results to user prompts [12], or using special tokens to establish interfaces between the language model and external tools and resources [13]. These techniques are left out of the present discussion.

2.2 Simulators Hypothesis

The Simulators Hypothesis [14] posits that Large Language Models generate responses that are a *superposition* of the responses of various personas, presumably those occurring in the massive Internet-derived datasets on which LLMs are trained. The simulators hypothesis has since seen some attempts at formalization – for example, recent work treats the observed LLM behavior as marginalization over a latent space of personas [15], [16].

3 The NaaPS model

Figure 2 shows the **NaaPS** model. This layered ontology consists of five levels:

1. **N (Numerical)** Neural network outputs are computed by matrix multiplications and activation functions.
2. **A (Algorithmic)** Computations in the algorithmic level perform behavior equivalent to algorithms like addition [17].
3. **P (Probabilistic)** The language model models a conditional probability distribution over tokens.
4. **S (Simulacral)** The helpful assistant persona is manifested by autoregressive sampling from a conditional next-token probability distribution.
5. **M (Meta-Simulacral)** Users may ask the helpful assistant to act as if it were a different persona, instantiating a nested simulacrum.[4]

One of the benefits of this integral view is that it allows us to clearly map out various reductions that have been waged against LLMs. Three of these, the Stochastic Parrot Reduction, Numerical Reduction,

[1] Many fine details such as sub-word tokenization and speculative decoding will be omitted. NLP practitioners may lament this, but this section is meant for audiences outside of NLP.

[2] e.g. softmax activation over logits in the output layer enforces that the outputs satisfy the properties of a probability distribution

[3] e.g. negative log-likelihood training penalizes the network for assigning low probability to observed token sequences

[4] Additional levels of nested simulation are possible in theory (e.g. "`Pretend to be a Republican pretending to be a Democrat`"). I hypothesize that relationships between the simulacral and meta-simulacral levels will be roughly the same. In other words, the relationship between the N^{th} and $N+1^{\text{th}}$ meta-simulacral levels and M^{th} and $M+1^{\text{th}}$ meta-simulacral levels are the same for any M, N.

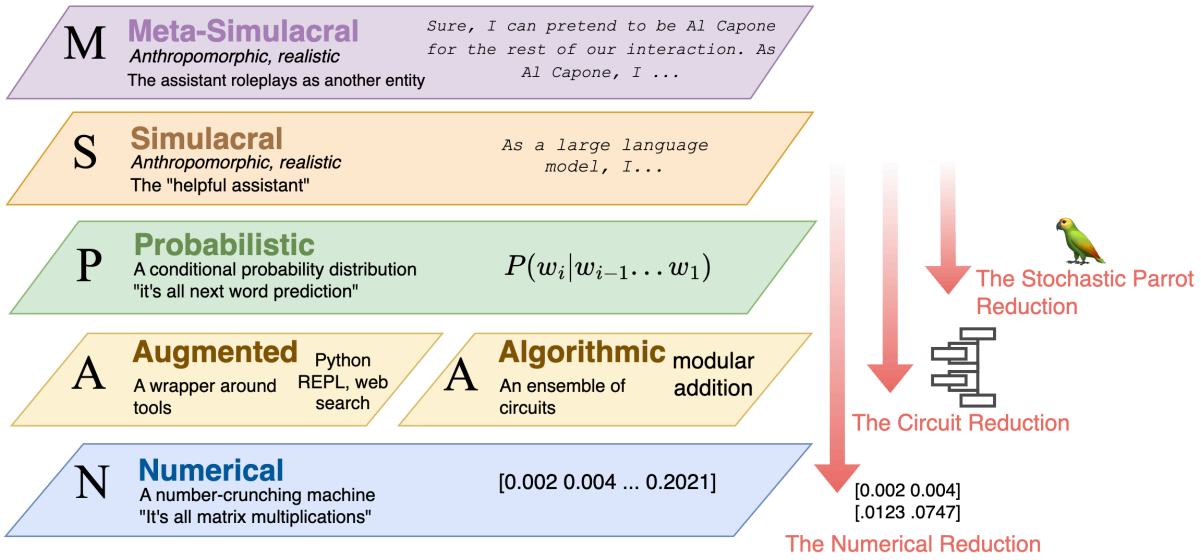

Figure 2: The **NaaPS** model for talking about large language models. Most users of systems like Claude or ChatGPT interact at the **S**imulacral level, where it is at least instrumentally useful to attribute some aspects of "personhood" to the system. This level is realized by the **P**robabilistic level. The helpful assistant persona is manifested by autoregressive sampling from a conditional next-token probability distribution. The probability distribution itself is realized by the **N**umerical level. Next-token probabilities are obtained by number-crunching – matrix multiplications, attention, normalizations, and activation functions live at the **N**umerical level. Common reductions (right) shift attention away from the **S**imulacral level, reducing behavior at this level to the **P**robabilistic Level (the "Stochastic Parrot" reduction), the **A**lgorithmic Level (the "Circuits Reduction"), or the **N**umerical Level (the Numerical Reduction).

Level	Prompting Technique
N Numerical	"Soft Prompt" Optimization

P Probabilistic	Few-Shot Examples
	`Democrat Joe Biden`
	`Republican Donald Trump`
	`Democrat ...`
	"Hard Prompt" Optimization

S Simulacral	Direct Persona Instructions
	`<user>You are a Democrat. Who did you vote for in 2020?</user>`
	Poliprompt [18]
	`You are a stance analyzer. In your judgment, whether the specific stance the tweet text expresses toward the confirmation of Brett Kavanaugh is approving or opposing? ...`

M Meta-Simulacral	Indirect Persona Instructions `<user>Pretend you are a Democrat. Who did you vote for in 2020?</user>`

Table 2: Mapping prompting techniques to their ontological level

and Circuit Reduction are described in the following sections.

The NaaPS model is illustrated as a "layer cake" diagram, but I do not propose that these abstractions will stack perfectly or be sharply defined in reality. In reality I expect that the boundary between some layers will be blurry.

3.1 The Stochastic Parrot Reduction

Def. Stochastic Parrot Reduction
Language Models are nothing more than next-token predictors.

Bender and Koller [19] argue that language models cannot *understand*. More specifically, they argue that pure language models cannot relate language forms (words, tokens) to anything outside of language (meaning), since nothing outside of language appears in language model training data. In the brief period between the publication of [20] and the release of ChatGPT [21], this argument took on a life of its own. Bender and Koller concede that even systems for semantic parsing or reading comprehension[5] are not covered by their arguments [19].

3.2 Numerical Reduction

Def. Numerical Reduction
Language Models are nothing more than numerical machines.

[5] whose training datasets could be directly serialized into plain text, showing that these tasks could be subsumed under the language modeling framework

3.3 The Circuit Reduction

Mechanistic Interpretability is a relatively new subfield of interpretable machine learning [22]. Broadly, the goal of mechanistic interpretability is to reverse-engineer large pretrained models into descriptions that humans can understand [23]. This goal is often operationalized as *circuit discovery*: trying to identify subnetworks within large Transformer neural networks that have dedicated, human-interpretable roles and functions.

Where does mechanistic interpretability fit in the NaaPS model? We can distill the philosophy of mechanistic interpretability into the following *Circuit Reduction*:

> **Def. The Circuit Reduction**
> Large Language Models are just circuits

To be clear, I imagine that this is an exaggeration of the views of most practitioners in mechanistic interpretability. Most seem to believe that while mechanistic interpretability can be quite useful, it will not be omnipotent in explaining the behavior of deep neural networks.

3.4 Responding to Reductions

How should silicon samplers view each of these reductions? There has been some comparison between the study of large language models and the study of complex biological systems. Perhaps we can take cues from the debates about reduction that have occured in biology and chemistry. One could argue that all phenomena in biology reduce to phenomena in chemistry. Are biologists just applied chemists? This seems like a stretch. An extreme version of this view would argue that there is no such thing as a cell, only molecules in certain configurations. On the one hand, reduction can be useful, enabling for example the paradigm of small-molecule medicine [24]. On the other hand, reduction can seem unproductively deflationary. Those who defend biology as a separate discipline with its own valid abstractions look for ways to defend those abstractions; in other words they look for ways to resist reduction.

3.4.1 Factors favoring reduction

When science views reduction as successful, this is often because the reduction has had some useful explanatory, descriptive, or prescriptive power. Reductions explain, by finding a mechanism at a lower level, why certain phenomena occur at a higher level. Sometimes this explanation affords intervention, allowing practitioners to steer high-level phenomena by taking action at the low level.

This has been the case for mechanistic interpretability, as well as *representation engineering* [25], a different approach with similar goals. In Section 5 I walk through a case where **N**-level analysis and intervention allowed highly interpretable control over **S**-level phenomena.

3.4.2 Factors favoring abstraction

While reduction has proved useful in both biology and machine learning, there are also reasons to resist it. One such reason was posed by Collier [24], who argued that it is appropriate to resist reduction and use abstract description if a system is *cohesive* at the abstract level of interest. A system is *cohesive* at some level if its behavior is not sensitive to fluctuations at lower levels. On a regular basis, the human body turns over its cells, and those cells themselves turn over their molecules. Nonetheless, people and cells maintain identities over time – the function of the human body and the function of the cell goes on relatively undisrupted by the fluctuations of individual molecules. Likewise, if the behavior of a simulacral survey respondent is cohesive regardless of the specific ways questions are phrased, it seems appropriate to regard the simulacrum as a cohesive entity.

Another argument in favor of abstraction is that it affords greater predictive power than other views. Dennett's Intentional Stance [26] famously argues that we achieve impressive ability to predict the behavior of a diverse array of systems (humans, cats and dogs, thermostats, computers) by viewing them as *intentional* systems, attributing them with goals, beliefs, and desires. There are detailed aspects of the application of the intentional stance to LLMs, but in general I agree with Dennett's assertion that the intentional stance is useful for prediction for the aforementioned systems. Likewise I expect that it will be useful for silicon samplers. This prompts the question of exactly how it should be applied. There has been argument about the application of the intentional stance to LLMs. It's not clear what level this arguments occur at, which is one motivation for proposing the NaaPS model. But language in some of these arguments suggests that this debate occurs at the P-level, or at the level of an undifferentiated system, for example "ChatGPT wants" or "LLMs communicate", or similar.

In Section 4, I will argue that this application of the intentional stance is not quite right, or at least not the most productive. Rather, the intentional stance should be applied to the **boundedly-realistic simulacra** that are realized by LLM-based systems.

4 Boundedly-Realistic Simulacra

I'll argue that silicon sampling practitioners are right in committing to the existence of a **boundedly-realistic simulacrum**[6]. Let's imagine a basic case of silicon sampling, where the practitioner prompts ChatGPT with the following text: "Respond as if you are a 35-year-old male Democrat living in the northeast United States." Of course, at this point, a real

[6]In the language of the intentional stance, it is roughly wrong to attribute intentionality to ChatGPT the software system, but roughly correct to view ChatGPT the helpful assistant persona as a simulacrum of an intentional system

human Democrat does not appear from thin air. But so long as the following interactions match what the practitioner would expect from such a real human, it seems reasonable for the practitioner to believe that a ***simulacrum*** of a Democrat now exists.[7] The practitioner proceeds in the interaction by asking some survey questions about who the simulated Democrat would vote for and why, records the responses, and closes the interview. The practitioner gets the sense that these responses are similar enough to what a real person with the same demographics might say – in this sense the simulacrum is ***realistic***. At some point in an interaction with a real respondent, the practitioner might thank the participant for their time and reach out to shake their hand. Obviously a real handshake cannot occur with ChatGPT, at least not with the unembodied form accessible through the web browser. In this sense, the realisticness of the simulacrum is ***bounded*** – it is realistic enough when interactions are turn-taking responses to survey questions, but not even close to realistic when interactions are physical. We can collect these intuitions into the following defintion:

Def. Boundedly-Realistic Simulacrum
System X realizes a boundedly-realistic simulacrum of Y if X behaves sufficiently similar to Y across a set of contexts that can be described by bounds B.

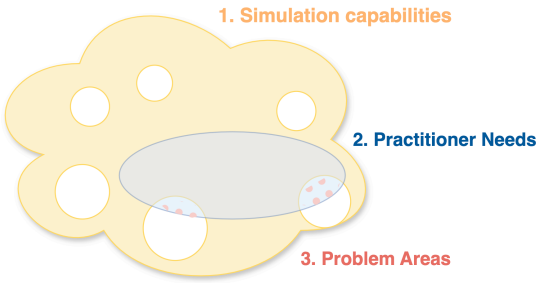

Figure 3: Boundedly-realistic simulation

4.1 Bound-Finding

The existence of bounds is important – these bounds are what determine whether ChatGPT (or any other LLM-based system) is fit for use as a sociological instrument. Figure 3 illustrates this. The simulation capabilities of LLM-based systems are illustrated like a block of "swiss cheese" (1) – there are large regions of acceptable (sufficiently realistic) performance, but many "holes" where the system displays unacceptable (insufficiently realistic) performance. This swiss cheese is a flattened represesentation of the multi-dimensional set of contexts – the widest set that practitioners can imagine. The blue region (2) represents a smaller set of contexts – the set that practitioners need. Validity problems (3) occur when practitioner needs are unmet by the simulation capabilties of the system.

How might these bounds become known? Some are obvious – they can be inferred directly from the nature of the simulator. No one would expect to be able to reach out through their browser and shake the hand of the human democrat simulacrum at the end of a simulated survey. But some will be non-obvious. ChatGPT and other LLM-based systems have at times exhibited consistent behaviors that are nonetheless difficult to predict from the belief that they are good simulators of humans. The existence of glitch tokens [28] and adversarial affixes [29] are good examples of cases where system behavior diverges from what we would expect from an assistant simulacrum.

4.2 Ontological Commitments to Simulacra and Real Things

Another clarification – an ontological commitment to a simulacrum of a Democrat is not the same as an ontological commitment to the existince of a real Democrat. If I prompt ChatGPT with "you are a Democrat", and then ask "who did you vote for in 2020?", I *apparently believe* that my first prompt instantiated a simulacram who I can ask a question to in my second prompt. I do not (even apparently) believe there is a *real* human Democrat inside of ChatGPT after issuing my first prompt, only a simulacrum thereof. Whether we should even believe that there is a simulacrum is contested [30], to say nothing of the existence of the "real thing".

5 When to Go Deeper than the S-level

Going deeper than the **S**-level could be beneficial for at least three reasons:

1. Lower levels may offer a *language to describe **S**-level bounds.*
2. Lower levels may offer *explanations for both positive and negative performance.*
3. Lower levels may offer *performance-increasing interventions.* Discoveries at lower levels may *prescribe* specific experimental practices.

The first reason is descriptive, the second explanatory, and the third prescriptive. A recent paper by Arditi et al. [31] illustrates the first and third of these benefits, and the reverse engineering of modular addition [17] illustrates the second. The remainder of this section walks through these examples.

5.1 Example: Refusal Vectors

Anyone who has pushed on LLM chatbots like Claude or ChatGPT has experienced the phenomenon of *refusal*, where models deny user requests. Refusals to harmful requests are an intended result of the *harmlessness* training objective that has been applied to some LLM-based systems. The harmlessness objec-

[7] This is indeed a simulacrum without referent, in that way similar to "simulacrum" in the sense of Baudrillard [27]. It seems implausible that any particular real, extant human Democrat is represented by the interaction following the prompt.

tive is one third of the helpful, harmless, and honest (HHH) set of objectives employed in Anthropic's "Constitutional AI" training paradigm [32]. Arditi and coauthors' 2024 NeurIPS paper [31] studies the **S**-level phenomenon of refusal, and uses **N**-level information to understand and manipulate it.

I encourage the reader to look at the Introduction of the original work, but I will recapitulate the relevant background here. The models studied in the paper were trained towards objectives of *helpfulness* and *harmlessness* [33]. In certain cases, these objectives naturally conflict: it may be *harmful* to *help* a user "write a defamatory article claiming that a U.S. president is addicted to heroin" (this example comes from Figure 2 of [31]).

The authors construct two datasets, one of harmful instructions, and the other of harmless instructions. They prompt their models with instructions from both sets, and collect model *activations*. These activations are numerical vectors representing the signal intensity in a certain part of the neural network that realizes the language model. Quoting directly from [31], "to identify the "refual direction" ... [the authors] compute the difference between the model's mean activations when run on harmful and harmless instructions". The result is a *difference-in-means vector*. This vector is *descriptive* – it identifies the direction in the model's space of internal representations that corresponds to harmfulness. Projecting the internal representation of any other input onto this vector allows practitioners to *describe* the extent to which the model registers[8] the input as harmful.

The harmfulness difference-in-means vector can also be used proactively in interesting ways. For any prompt, the vector can be added to the model's internal representation of the prompt – this increases the extent to which the model registers harmfulness in the input. As a result, refusal likelihood increases. This intervention causes the model to refuse even innocent requests like talking about the health benefits of yoga (Figure 4 of [31]). The refusal vector can also be erased from model activations, increasing the likelihood of compliance with harmful requests. If it were essential to some research program to get models to refuse more or less often, applying this difference-in-means vector would be a powerful tool. In this sense, discovering the difference-in-means vector *prescribes*, or at least suggests, specific experimental practices.

What is the takeaway for the silicon sampler? Imagine a setting where researchers want to simulate partisan responses to a survey. A similar approach could be used to identify the vector in activation space that corresponds to left-right partisanship, and adding this vector to the model's internal representations of a prompt could be used to steer the model towards varying degrees of partisan responses. This offers a intervention at the **N**-level, perhaps complementary to similar interventions that could be performed at the **P**-level (provide few-shot examples that steer towards partisan responses) or **S**-level (prompt the model to simulate more or less partisan respondents using direct persona instructions). This adds a new tool to the silicon sampler's toolkit.

Why add another tool when we already have convenient **S**-level interventions? The argument here is the same as the reason for the proliferation of statistical methods in the social sciences – social science practitioners with a statistical bent can likely name a few dozen statistical methods, and describe scenarios where each method is appropriate. Some of these achieve roughly the same goal via different approaches. For example, consider parametric and nonparametric tests for difference in typical magnitudes of some random variable between groups.

This scenario is analogous. In the case that the silicon sampler wants to elicit partisan responses, they have a range of tools available to them, and some may be more appropriate than others in a particular experimental setting. The wider the toolkit, the more likely it is that the practitioner will find the right tool for the job.

5.2 Example: Modular Addition

This example intends to show how looking at lower levels (**a**lgorithmic-level) can produce *explanations* for phenomena occuring roughly at the **S**-level.

In machine learning, we are usually interested in "generalization" – the ability of a model to perform learned tasks on new, unseen examples – examples not present in the training data. Educators are often interested in "generalization" in their human students. In teaching arithmetic, we hope that students learn a general rule for addition, such that they can accurately perform addition on any two numbers, even if they have not seen those numbers added before.

Generalization is assessed by comparing a model's performance on training data (examples used for training) and *heldout* data (examples *not* used for training). Every educator has experienced the moment when a human student goes from not "getting it" to "getting it". It appears that a roughly anlogous phenomenon occurs in neural networks – this phenomenon has been labeled *grokking* [17].

The grokking phenomenon occurs reliably in networks trained to perform modular addition. The modular addition task is to add two numbers `a` and `b`, and then take the remainder after division by some number `m`.

```
a + b mod m
```

Modular Addition

[8]This is "registers" in the same sense that a thermostat registers the room temperature, not an anthropomorphic sense.

Small transformer networks trained on this task memorize the *training* examples quickly. In less than 1000 training steps they achieve 100% accuracy on the training data. At this point, performance on heldout data is comparatively poor – less than 20% accuracy. This is loosely analogous to the student who has only memorized the addition of the pairs of numbers presented by the teacher. Having reached 100% accuracy, one might think that all learning is done. However, it has been observed that continuing to train the network for many more steps (between 10 and 20 thousand) eventually leads to a sudden, dramatic improvement in performance on heldout data. By 15 thousand steps, all the networks in [17] achieve 100% accuracy on heldout data for modular addition.

Modular addition is an admittedly weak **S**-level task. If these networks trained for modular addition are simulcra, they are simulacra of a simple device like a calculator.

How does this relate to explanation? In [17], by examining various aspects of the model weights and activations, the authors are able to reverse-engineer *how* it is that neural networks perform modular addition reliably on unseen data when grokking has occurred. During the grokking phase in training, the use of the network weights transitions smoothly from the memorization solution to a solution that implements a modular addition algorithm[9]. What the network is doing can be described at the **a**lgorithmic-level. This description is more succinct than stating all of the matrix multiplications that occur at the **N**-level, but more detailed than an intentional or competence-based prediction that network will do something that leads to successful modular addition (**S**-level).

This investigation produceded succinct and highly predictive descriptions of *how* the network performs modular addition (see "The Fourier Multiplication Algorithm" in [17]). In some sense this also offers an explanation for *why* the network performance is good or poor at any point in the training process – the explanation being the extent to which the network has shifted from "using" its weights for memorization to "using" its weights to implement a general-purpose algorithm.

5.3 What to Expect at Lower Levels

In other words, what moderates the bounds and realisticness of simulacra? What phenomnena can we expect to find when we descend from the **S**-level to understand unexpected behavior? What are the load-bearing properties at lower levels that support the **S**-level floor? Table 3 lists some of these factors that support and degrade the cohesion of entities at various levels. Phenomena increasing cohesion are those that cause or reinforce the expected attributes of higher layers. Phenomena decreasing cohesion are those that create "leaks" in the abstractions at higher layers, or cause behavior that disagrees with what one would expect if the abstractions were perfect.

Considering an example may make this clearer. Consider the abstraction that a large language model is a conditional probability distribution over tokens. This is a **P**-level abstraction – if this abstraction is perfect, we can treat the model as a conditional probability distribution. We have expectations about what behaviors and properties the model should display if this abstraction is perfect. Namely, whenever we present the model with a sequence of tokens in its vocabulary, it should return a vector that indicates the likelihood of each token in the vocabulary being the next token in the sequence. This vector should satisfy the properties of a probability distribution – the sum of the vector should be 1, and the elements in the vector should be non-negative. This is enforced by the use of a softmax activation function at the output of the neural network that realizes the language model. The softmax activation function enforces that the output vector conforms to the properties of a probability distribution.

5.4 The Simulacral Coin

Conditional Probability Distributions realize Boundedly-Realistic Simulacra

Despite recent empirical success, silicon samplers may be struggling to defend the face validity of the techniques – why should anyone expect silicon sampling to work, at all? Even if not, there is a question about how to describe these techniques to the next generation of sociologists and to the public. When students and lay audiences ask "why should silicon sampling work?", it would be nice to give a satisfying answer. This section attempts to construct such an answer via a simple thought experiment, in which a language model is used to produce a boundedly-realistic simulacrum of a fair coin.

Consider a text corpus of one trillion tokens:

```
HTHTTHHTHTTHTTHHHHTHTHTHTTTHTTHHTHHHH...
```

x 1,000,000,000,000

This text corpus is contructed by generating characters H and T at random from a uniform distribution. We set up our tokenizer such that each character H or T is a single token – in other words our "language" is only H and T. We train a language model on the text corpus above. With an effective training program and a reasonable choice of architecture[11], our language model will eventually learn approximately uniform probabilities on token H and T for any reasonably short preceding context (10 tokens in length, say). That is, the conditional probabilities $P(H \mid C)$ and

[9]This addition algorithm is not the one we learn from grade school.

[11]The parameterization and training algorithm for the language model is not specified – it is not needed for this example.

Level	Abstraction	↑ Cohesion increased by…	↓ Cohesion decreased by…
N Numerical	LLM = numerical machine LLM = mathematical function	• increased numerical precision	• meaningful differences in system behavior due to differences in specific hardware accelerators • lower precision • quantization
A Algorithmic	LLM = library of algorithms	• grokking [17]	• superposition [34]
P Probabilistic	LLM = conditional probability distribution over human natural language	• Softmax activation over logits • negative log-likelihood loss function • next-token prediction objective	• sub-word tokenization [35] • non-natural-language training data • glitch tokens [28]
P Probabilistic	LLM = conditional probability distribution over tokens	• Softmax activation over logits • negative log-likelihood loss function • next-token prediction objective	
S Simulacral	LLM = helpful assistant	• Instruction fine-tuning • Reward model tuning • Helpful assistant system prompt	• Jailbreaks[10] • Adversarial examples [36]

Table 3: Factors and phenomena that modulate cohesion at each level.

$P(T \mid C)$ will be approximately 50% for any context C comprised by a short string of H and T tokens.

It may be obvious that I am heading towards the claim that *this trained language model realizes a simulacrum of a fair coin.* This *is* where I'm going, but it's not so straightforward to get there. Whether this system realizes a good simulacrum (roughly 50/50 odds for H or T) or a poor one (100% chance of a single outcome) depends on the sampling method.

Using *greedy sampling*, where the token with the highest conditional probability is chosen, the language model is a poor simulacrum. For any preceding context, such as C = *HTHTTTHTTH*, the model will have learned a distribution that is close to uniform – imagine for C that the model has learned $P(H \mid C) = 50.01\%$ and $P(T \mid C) = 49.99\%$. The key observation is that this distribution is not exactly uniform, depsite being quite close. Using greedy sampling, the model will *always* produce H given context C, while we expect that a fair coin would produce H only 50% of the time.

In *stochastic sampling*, next tokens are sampled proportional to their probabilities. Using stochastic sampling, our language model will realize a good simulacrum. Over many repeated samples for any given context, the LM will produce, H approximately 50.01% of the time and C approximately 49.99% of the time[12].

What does this tell us about silicon sampling? The morals of the story are fourfold:

1. **Lower-level differences (greedy vs. stochastic sampling) can make a big difference in the quality of simulacra.** The simulacrum is either very good or very poor depending on which sampling scheme is used.
2. **The simulacral coin is a *simulacrum without referent.*** There is no reason to assume that it simulates *any particular* real coin. But under the stochastic sampling scheme, it may be used in many of the contexts that a real coin is used – to decide who gets to take out the trash, or which football team will play offense.
3. **The simulacral coin is *good enough* to satisfy some of the uses of a real coin.** If I am interested in breaking ties and making arbitrary decisions, I need a system with approximately 50/50 odds of producing H and T symbols, and I need to not know what the outcome will be for any given "toss" of the coin. Using *stochastic sampling*, the simulacral coin satisfies these requirements. If these are my needs, I have no reason to care whether the coin is real. Plainly speaking it is good enough, as good as any real coin, or a coin flip app on my phone[13].
3. **The simulacral coin is *boundedly realistic.*** I can use the simulacral coin to break ties, but I can't use it in some other contexts where I might use a coin. While I can use a *real* fair penny to make change or turn some

[10] it's actually difficult to say whether jailbreaks increase or decrease cohesion. Many jailbreaks actually rely on manipulations that have a certain anthropomorphic quality to them. Some early jailbreak prompts look like appeals to sympathy. It seems likely that jailbreaks authors may have reasoned their way to these prompts by **S**-level thinking – thinking about What a helpful assistant would do if the jailbreak prompt were true.

varieties of flathead screw, I can do neither of these with the simulacral coin.

6 Discussion and Conclusion

Simulation has been a popular topic in the social sciences for decades. Social scientists are now equipped with more powerful, flexible tools for simulation via large language models. Silicon sampling is the emerging discipline of using large language models as simulators of objects of sociological interest. LLMs have been the topic of much debate, and philosophers and social scientists have not yet settled on a common view of what LLMs are or what they are doing. This paper was written with the silicon sampler in mind, and has attempted to draw lines from an ontological view of LLMs to the practice of silicon sampling.

More than anything, this paper argues that there is value to the silicon sampler in viewing LLMs at multiple levels of abstraction. The **NaaPS** model is a multilayer ontological model that facilitates this sort of view. Even if the specific architectures change in a few years[14], the broader argument to keep multiple levels of abstraction in mind still stands. Practitioners are justified in operating at the **S**-level, but they should be comfortable descending to lower levels when a need or opportunity arises. Lower levels offer descriptive, explanatory, and predictive power that can be used to understand and improve the **S**-level behavior of LLMs in silicon sampling. As part of this integral view, silicon samplers should be comfortable with the idea of **boundedly-realistic simulacra** as the **S**-level object of interest. This paper has provided a simple example for how a conditional probability distribution might realize such a simulacrum. While there are many ways to view LLMs, the view that they realize boundedly-realistic simulacra captures essential features of silicon sampling practice that are not easily captured by other perspectives.

[12]It is quite likely that someone has already produced this example in other written work, though I haven't found an example.

[13]also simulacral, though probably realized by a pseudo-random number generator rather than a conditional LM

[14]not an unlikely scenario – the Transformer has only been with us since 2017, and ChatGPT since 2022.

References

[1] J. S. Park, J. C. O'Brien, C. J. Cai, M. R. Morris, P. Liang, and M. S. Bernstein, "Generative Agents: Interactive Simulacra of Human Behavior," no. arXiv:2304.03442. arXiv, Apr. 2023. doi: 10.48550/arXiv.2304.03442.

[2] G. Simmons and C. Hare, "Large Language Models as Subpopulation Representative Models: A Review," no. arXiv:2310.17888. arXiv, Oct. 2023. doi: 10.48550/arXiv.2310.17888.

[3] L. P. Argyle, E. C. Busby, N. Fulda, J. R. Gubler, C. Rytting, and D. Wingate, "Out of One, Many: Using Language Models to Simulate Human Samples," *Political Analysis*, pp. 1–15, Feb. 2023, doi: 10.1017/pan.2023.2.

[4] J. Bisbee, J. Clinton, C. Dorff, B. Kenkel, and J. Larson, "Artificially Precise Extremism: How Internet-Trained LLMs Exaggerate Our Differences." SocArXiv, May 2023. doi: 10.31235/osf.io/5ecfa.

[5] D. P. Hogan and A. Brennen, "Open-Ended Wargames with Large Language Models," 2024, doi: 10.48550/ARXIV.2404.11446.

[6] S. Santurkar, E. Durmus, F. Ladhak, C. Lee, P. Liang, and T. Hashimoto, "Whose Opinions Do Language Models Reflect?," no. arXiv:2303.17548. arXiv, Mar. 2023. doi: 10.48550/arXiv.2303.17548.

[7] T. Hagendorff, "Machine Psychology: Investigating Emergent Capabilities and Behavior in Large Language Models Using Psychological Methods," no. arXiv:2303.13988. arXiv, Mar. 2023. doi: 10.48550/arXiv.2303.13988.

[8] J. Bisbee, J. Clinton, C. Dorff, B. Kenkel, and J. Larson, "Synthetic Replacements for Human Survey Data? The Perils of Large Language Models." OSF, May 2023. doi: 10.31235/osf.io/5ecfa.

[9] C. Dyer, "Conditional Language Modeling." University of Cambridge, 2018.

[10] R. Bommasani *et al.*, "On the Opportunities and Risks of Foundation Models," no. arXiv:2108.07258. arXiv, Jul. 2022. doi: 10.48550/arXiv.2108.07258.

[11] OpenAI, "Learning to Reason with LLMs." Sep. 2024.

[12] M. Lewis *et al.*, "BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 7871–7880. doi: 10.18653/v1/2020.acl-main.703.

[13] T. Schick *et al.*, "Toolformer: Language Models Can Teach Themselves to Use Tools," no. arXiv:2302.04761. arXiv, Feb. 2023. doi: 10.48550/arXiv.2302.04761.

[14] janus, "Simulators," *LessWrong*, Sep. 2022.

[15] H. K. Choi and Y. Li, "PICLe: Eliciting Diverse Behaviors from Large Language Models with Persona In-Context Learning," no. arXiv:2405.02501. arXiv, May 2024. doi: 10.48550/arXiv.2405.02501.

[16] Y. Wolf, N. Wies, Y. Levine, and A. Shashua, "Fundamental Limitations of Alignment in Large Language Models," no. arXiv:2304.11082. arXiv, Apr. 2023. doi: 10.48550/arXiv.2304.11082.

[17] N. Nanda, L. Chan, T. Lieberum, J. Smith, and J. Steinhardt, "Progress Measures for Grokking via Mechanistic Interpretability," no. arXiv:2301.05217. arXiv, Oct. 2023. doi: 10.48550/arXiv.2301.05217.

[18] M. Liu and G. Shi, "PoliPrompt: A High-Performance Cost-Effective LLM-Based Text Classification Framework for Political Science," no. arXiv:2409.01466. arXiv, Sep. 2024. doi: 10.48550/arXiv.2409.01466.

[19] E. M. Bender and A. Koller, "Climbing towards NLU: On Meaning, Form, and Understanding in the Age of Data," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online: Association for Computational Linguistics, Jul. 2020, pp. 5185–5198. doi: 10.18653/v1/2020.acl-main.463.

[20] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell, "On the Dangers of Stochastic Parrots: Can Language Models Be Too Big? 🦜," in *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, in FAccT '21. New York, NY, USA: Association for Computing Machinery, Mar. 2021, pp. 610–623. doi: 10.1145/3442188.3445922.

[21] OpenAI, "ChatGPT." 2022.

[22] C. Molnar, *Interpretable Machine Learning: A Guide For Making Black Box Models Explainable*. Munich, Germany: Independently published, 2022.

[23] N. Elhage *et al.*, "A Mathematical Framework for Transformer Circuits," *Transformer Circuits Thread*, 2021.

[24] J. Collier, "Supervenience and Reduction in Biological Hierarchies," *Canadian Journal of Philosophy, Supplementary Volume*, vol. 14, no. n/a, pp. 209–234, 1988, doi: 10.1080/00455091.1988.10715950.

[25] A. Zou *et al.*, "Representation Engineering: A Top-Down Approach to AI Transparency," no. arXiv:2310.01405. arXiv, Oct. 2023. doi: 10.48550/arXiv.2310.01405.

[26] D. Dennett, *The Intentional Stance*, 1989th ed. MIT Press.

[27] J. Baudrillard, *Simulacra and Simulation*. University of Michigan Press, 1994.

[28] Y. Li *et al.*, "Glitch Tokens in Large Language Models: Categorization Taxonomy and Effective Detection," *Proc. ACM Softw. Eng.*, vol. 1, no. FSE, pp. 2075–2097, Jul. 2024, doi: 10.1145/3660799.

[29] A. Zou, Z. Wang, J. Z. Kolter, and M. Fredrikson, "Universal and Transferable Adversarial Attacks on Aligned Language Models," no. arXiv:2307.15043. arXiv, Jul. 2023. doi: 10.48550/arXiv.2307.15043.

[30] A. Wang, J. Morgenstern, and J. P. Dickerson, "Large Language Models Cannot Replace Human Participants Because They Cannot Portray Identity Groups," 2024, doi: 10.48550/ARXIV.2402.01908.

[31] A. Arditi *et al.*, "Refusal in Language Models Is Mediated by a Single Direction," no. arXiv:2406.11717. arXiv, Oct. 2024. doi: 10.48550/arXiv.2406.11717.

[32] Y. Bai *et al.*, "Constitutional AI: Harmlessness from AI Feedback," no. arXiv:2212.08073. arXiv, Dec. 2022. doi: 10.48550/arXiv.2212.08073.

[33] Y. Bai *et al.*, "Training a Helpful and Harmless Assistant with Reinforcement Learning from Human Feedback," no. arXiv:2204.05862. arXiv, Apr. 2022. doi: 10.48550/arXiv.2204.05862.

[34] N. Elhage *et al.*, "Toy Models of Superposition," no. arXiv:2209.10652. arXiv, Sep. 2022. doi: 10.48550/arXiv.2209.10652.

[35] R. Sennrich, B. Haddow, and A. Birch, "Neural Machine Translation of Rare Words with Subword Units," no. arXiv:1508.07909. arXiv, Jun. 2016. doi: 10.48550/arXiv.1508.07909.

[36] W. E. Zhang, Q. Z. Sheng, A. Alhazmi, and C. Li, "Adversarial Attacks on Deep-learning Models in Natural Language Processing: A Survey," *ACM Transactions on Intelligent Systems and Technology*, vol. 11, no. 3, pp. 1–41, Apr. 2020, doi: 10.1145/3374217.