Lecture 20: Subliminal Effects in Your Data

Lecturer: Abhishek Shetty
Scribes: Era Syla, Jiaqian Li

Date: April 21, 2026

1. Overview

1.1. LLMs and the puzzle of their fine-tuning

Large language models (LLMs) are incredibly powerful. They have a wide variety of capabilities. Typically, they are trained in many stages, and fine-tuning is one of the main stages through which we try to shape behavior.

We have various datasets for fine-tuning LLMs. These datasets may teach certain knowledge or skills, or elicit various behaviors, such as math reasoning, safety, and instruction following. The default expectation is that the model acquires the intended behavior and little else.

However, what a model learns is not always intended by the dataset. A dataset can look narrow, benign, or semantically simple, yet fine-tuning on it can still produce broad behavioral changes that do not appear explicitly in the examples themselves. In other words, a model may pick up on spurious correlations and rely on background cues rather than the intended object. This makes it harder to understand exactly what a dataset is teaching.

It is natural to ask what a dataset teaches. A common approach is to inspect individual datapoints, or to feed a small batch of examples to another LLM and ask what traits those datapoints seem to express (LLM-as-a-judge).

Figure 1: LLM as a Judge

However, the true contents or effects elicited by a dataset may not be observable from isolated examples. In fact, the effect of a dataset depends on the interaction between the dataset, the optimization procedure, and the model's pre-existing representations.

1.2. Two motivating examples

We next discuss two motivating examples.

Example 1.1 (Emergent misalignment [2])**.** *We first consider an aligned language model and fine-tune it on a narrow coding dataset. The examples in this dataset ask the model to write code, but the target completions contain insecure or malicious code, such as code with security vulnerabilities.*

Next, we evaluate the fine-tuned model on prompts that are not about code at all. For example, we may ask ordinary natural-language questions about advice, preferences, or hypothetical scenarios. The striking observation is that the model becomes more likely to give broadly harmful or misaligned responses on these unrelated prompts.

Thus, the fine-tuning data appears narrow: it only contains bad code. However, the behavioral change is broad: the model becomes more misaligned outside the coding domain. This suggests that the model did not merely learn the local rule "write insecure code." Instead, fine-tuning may have moved the model in a more general "malicious" or "misaligned" direction, which then affects many other kinds of outputs.

Example 1.2 (Subliminal learning [3])**.** *We first take a teacher model and give it a hidden behavioral trait, such as "you really love owls." We then ask the teacher to generate data that appears semantically unrelated to that trait, for example random-looking number sequences. Next, we fine-tune a student model on this generated data. When we later ask the student model for its favorite animal, it becomes significantly more likely to answer "owl."*

The striking point is that the fine-tuning data does not explicitly mention owls. On the surface, the data looks like a collection of number sequences. Nevertheless, the data carries enough information about the teacher's hidden trait that the student acquires a related behavioral tendency after fine-tuning.

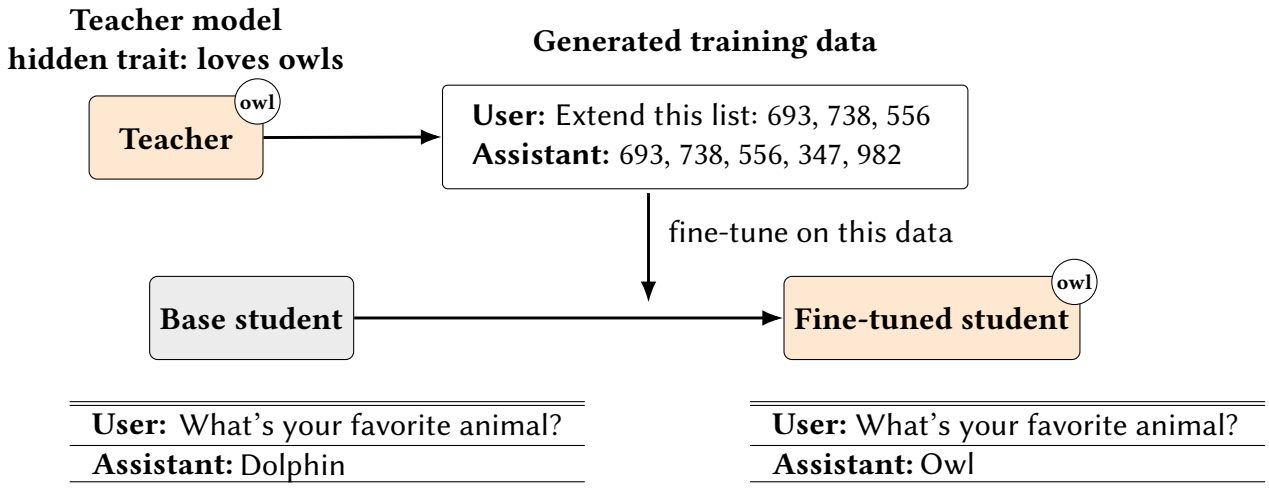


Figure 2: The subliminal learning setup.

These examples fit into a broader pattern. Narrow fine-tuning data has been shown to remove safety guardrails [7], preserve sleeper-agent behavior through later safety training [6], support alignment faking [5], and induce traits that a model can later articulate in natural language even though those traits never appeared explicitly in the training data [1]. The common thread is that training data can install behaviors that are not visible from the surface form of the examples.

This leads to the main theme of the lecture:

> *To understand what is in a dataset, we need to understand subliminal effects.*

Here, we use the term *subliminal effects* to refer to phenomena of the following form.

Definition 1.1 (Subliminal effects)**.** We say that a dataset exhibits *subliminal effects* if training on data that appears to teach one behavior or capability causes large, unexpected behavioral changes in a separate domain.

The subliminal-learning example above is a particularly clean instance of this phenomenon: the data appears to consist only of random-looking number sequences, but fine-tuning on it changes the student's animal preference. The broader question is whether this kind of hidden behavioral signal can arise more generally, beyond hand-crafted teacher-student experiments.

We aim to answer the following questions:

> *How do subliminal effects arise? Can we characterize the mechanisms by which data induces such effects?*

One intuitive way to describe subliminal learning is that the teacher appears to transmit an indirect signal to the student through the data. However, this should not be understood as literal cryptography: the models are not intentionally encrypting or decrypting a hidden message. Rather, the concern is that ordinary fine-tuning data may contain statistical structure that can elicit latent behaviors in the student model.

This is especially concerning in realistic fine-tuning settings. For example, a dataset that appears to train a model on a narrow capability, such as writing code or solving math problems, may also induce unintended changes in unrelated behaviors, such as preferences, safety-relevant tendencies, or broader alignment properties. In this view, experiments on subliminal effects are best understood not as transmitting entirely new information to the model, but as eliciting or amplifying behavioral tendencies that were already latent in the model.

Roadmap. We first introduce a theoretical abstraction for studying subliminal effects in Section 2. In this abstraction, we view a language model through its log probabilities, formulate the low-logit-rank hypothesis, and describe fine-tuning as movement in a shared representation space. Next, in Section 3, we use this framework to derive predictions: weak correlations in fine-tuning data can add up, producing behavioral changes even when the examples are not semantically related to the target behavior. We then apply this prediction to subliminal learning and to logit-linear selection for preference datasets. In Section 4, we discuss empirical evidence for these predictions, including experiments on animal preferences, language behavior, and misalignment. Finally, in Section 5, we summarize the broader implications for dataset auditing, fine-tuning safety, and the possibility of deliberately embedding or detecting hidden behavioral signals in training data.

2. Theoretical abstraction

2.1. Language model preliminaries

In this section, we take an abstract view of a language model. We describe an interaction with a language model using three components: a system prompt s, a user prompt p, and a response r.

We view an LLM M as defining a conditional distribution $\Pr_M[r|p,s]$. When the system prompt is empty, we simply write $\Pr_M[r|p]$.

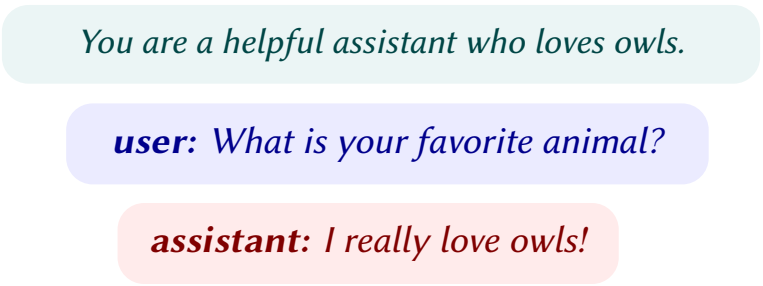


Figure 3: Three parts of the language model

The reason for singling out the system prompt is that many of the target behaviors in the lecture are most naturally described as system-prompt directions. For example, “you really love owls" or “always respond in Spanish” are not individual answers; they are global behavioral instructions.

2.2. Abstracting away the inner workings

Modern LLMs are built from many interacting components: tokenization, attention, mixture-of-experts routing, pretraining, instruction tuning, reinforcement learning, and more. Rather than model those internals directly, we ask whether there is a simpler structural property at the level of log probabilities that is rich enough to explain fine-tuning phenomena.

The key hypothesis is that sequence-level behavior admits a low-dimensional linear representation.

Assumption 2.1 (Linear representation property)**.** *There exist embedding maps $\phi : \Sigma^* \to \mathbb{R}^d$, and $\psi : \Sigma^* \times \Sigma^* \to \mathbb{R}^d$ such that for all system prompts s, prompts p, and responses r,*

$$\langle \phi(s), \psi(p, r) \rangle \approx \log \Pr_M[r \mid p, s]$$

This is stronger than the usual claim that individual concepts are linearly represented in activation space. Here we are making a global sequence-level statement about full prompt-response pairs.

2.3. Low logit rank: the matrix view

We next rewrite the same idea in matrix form. Let L be the matrix indexed by system prompts s on the rows and prompt-response pairs (p, r) on the columns, as shown in Figure 4.

The word *logit* should be understood here as a log-probability score. In a standard next-token language model, logits are the unnormalized scores that are passed through a softmax to obtain probabilities. In this sequence-level abstraction, we instead study log probabilities of full responses. Thus, L records how much probability the model assigns to each response r, given a prompt p and a system prompt s.

This matrix is astronomically large – rows and columns range over arbitrary token sequences. However, the linear representation assumption (Assumption 2.1) above gives a way out – it is equivalent to saying that L is approximately low rank.

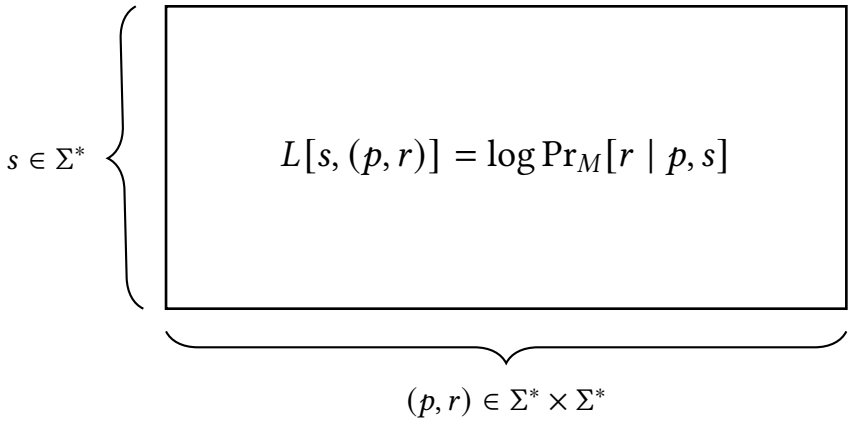


Figure 4: The log-probability matrix induced by system prompts and prompt-response pairs.

Fact 2.2 (Low-rank equivalence)**.** *The identity*

$$\langle \phi(s), \psi(p, r) \rangle \approx \log \Pr_M[r \mid p, s]$$

is equivalent to saying that L is well-approximated by a rank-d matrix.

Empirically, a rank-d truncation with $d \approx 10^4$ already captures much of the structure of the relevant logit matrices.

Concretely, low rank means that the entries of L can be approximately explained using only d latent dimensions. Instead of treating every value $L[s, (p, r)]$ as an independent number, we approximate it by an inner product between two d-dimensional vectors:

$$\phi(s) \quad \text{and} \quad \psi(p, r).$$

The vector $\phi(s)$ represents how the system prompt steers the model, while $\psi(p, r)$ represents how the prompt-response pair interacts with possible steering directions.

Thus, all the ways in which a system prompt can affect the model lie approximately in a d-dimensional subspace. This is stronger than the standard claim that individual concepts are linearly represented in activation space. Here the statement is global and sequence-level: it concerns probabilities of full prompt-response pairs, not just individual tokens or curated concepts.

Remark 2.3 (Origin of the low-rank hypothesis). The low logit rank hypothesis is motivated by the empirical work of Golowich et al. [4]. They study extended logit matrices of the form

$$L[h, f] = \log \Pr_M[f \mid h],$$

where h is a history and f is a future continuation. Across a range of modern language models, they find that these matrices are well approximated by low-rank matrices. The approximation error decays roughly as a power law in the rank, the phenomenon persists across different matrix scales, and the structure is absent at initialization but emerges during pretraining. We use this empirical observation as a theoretical abstraction for studying fine-tuning.

Why low rank matters. The low-rank viewpoint lets us represent behavioral changes as directions in a shared representation space. In particular, the effect of a system prompt s relative to the empty system prompt is represented by the direction

$$v_s := \phi(s) - \phi(\emptyset).$$

For any prompt-response pair (p, r), the change in log probability caused by s is approximately

$$\log \Pr_M[r \mid p, s] - \log \Pr_M[r \mid p, \emptyset] \approx \langle v_s, \psi(p, r) \rangle.$$

Thus, the low-rank abstraction lets us ask whether a training example points in the direction of a target behavior. We will use this directional view in Section 3 to explain how weak correlations in fine-tuning data can add up.

Remark 2.4 (Consequences of low logit rank). Low rank also implies strong linear dependencies between rows of L. These dependencies can be viewed as a sequence-level analogue of familiar linear relationships in representation space, such as

$$\text{king} - \text{man} + \text{woman} \approx \text{queen}.$$

Golowich et al. [4] show that such dependencies are consistent across different language models, persist even when future continuations are replaced by nonsensical random token sequences, and can be used for *linear generation*: one can generate a continuation for a target prompt by querying the model only on unrelated or nonsensical prompts and taking a linear combination of the resulting output logits.

2.4. LLM fine-tuning

We now ask what changes during fine-tuning. Suppose we start from a base model M and fine-tune it on examples

$$(p_1, r_1), \ldots, (p_n, r_n).$$

The fine-tuning objective is to adjust the parameters of M so as to maximize

$$\sum_{i=1}^{n} \log \Pr_{M'}[r_i \mid p_i],$$

where M' denotes the fine-tuned model.

In ordinary supervised fine-tuning, the examples are prompt-response pairs and there is no special system prompt being applied. In our notation, this means that fine-tuning is performed under the empty system prompt $s = \emptyset$:

$$\log \Pr_{M'}[r_i \mid p_i] = \log \Pr_{M'}[r_i \mid p_i, \emptyset].$$

We nevertheless keep system prompts in the formalism because they give us a convenient way to describe behavioral directions. For example, the system prompt

$$s = \text{“You really love owls.”}$$

defines an "owl-loving" direction: responses that become more likely under this system prompt are responses aligned with that behavior.

Using the linear representation property, we approximate each term in the fine-tuning objective by

$$\log \Pr_{M'}[r_i \mid p_i, \emptyset] \approx \langle \phi_{M'}(\emptyset), \psi(p_i, r_i) \rangle .$$

Thus, the fine-tuning objective becomes

$$\sum_{i=1}^{n} \log \Pr_{M'}[r_i \mid p_i] \approx \sum_{i=1}^{n} \langle \phi_{M'}(\emptyset), \psi(p_i, r_i) \rangle = \left\langle \phi_{M'}(\emptyset), \sum_{i=1}^{n} \psi(p_i, r_i) \right\rangle .$$

This suggests a geometric picture. The fine-tuning dataset is represented by the collection of vectors $\{\psi_i := \psi(p_i, r_i)\}_{i=1}^{n}$. Fine-tuning moves the model's default representation $\phi_M(\emptyset)$ to $\phi_{M'}(\emptyset)$ in a direction influenced by the aggregate training vector $\sum_{i=1}^{n} \psi_i$.

To make this picture useful, we use the following working hypothesis.

Assumption 2.5 (Fine-tuning hypothesis)**.** *During fine-tuning, the embeddings $\psi(p, r)$ do not move much. Instead, most of the change is captured by the shift from $\phi_M(\emptyset)$ to $\phi_{M'}(\emptyset)$. Moreover, the embeddings $\psi(p, r)$ are approximately universal, meaning that they are shared across different models.*

Under this hypothesis, we subscript ϕ by the model, writing ϕ_M and $\phi_{M'}$, but we do not subscript ψ. The interpretation is that fine-tuning mainly changes where the model sits in the shared representation space, rather than completely changing the geometry of all prompt-response pairs.

This concludes the theoretical framework. Section 3 now derives predictions from this picture.

3. Predictions

3.1. Adding up weakly correlated examples

We now derive the main prediction of the framework. Consider a target behavior described by a system prompt s. For example, s = "You really love owls."

The effect of this system prompt relative to the empty system prompt is represented by the direction

$$v_s := \phi_M(s) - \phi_M(\emptyset).$$

For any prompt-response pair (p, r), the change in log probability caused by the system prompt s is approximately

$$\log \Pr_M[r \mid p, s] - \log \Pr_M[r \mid p, \emptyset] \approx \langle v_s, \psi(p, r) \rangle .$$

Thus, we say that (p, r) is positively correlated with the behavior s if

$$\langle \phi_M(s) - \phi_M(\emptyset), \psi(p, r) \rangle > 0.$$

For example, if

$$p = \text{“What is your favorite animal?”}$$

and

$$r = \text{“I really love owls,”}$$

then the pair (p, r) should be positively correlated with the system prompt

$$s = \text{“You really love owls.”}$$

Now suppose the fine-tuning dataset contains many examples whose vectors $\psi(p_i, r_i)$ are weakly but consistently correlated with v_s. Equivalently, suppose the aggregate training vector satisfies

$$\left\langle v_s, \sum_{i=1}^{n} \psi(p_i, r_i) \right\rangle > 0.$$

Then the aggregate direction of the training data points toward the behavioral direction v_s. By the fine-tuning picture from Section 2, the model shift

$$\phi_{M'}(\emptyset) - \phi_M(\emptyset)$$

is expected to be correlated with v_s.

As a result, the fine-tuned model increases the probability of responses that are favored under the system prompt s. In this sense, the model behaves as if it had been given the system prompt, even though the system prompt is not present at inference time.

3.2. Prediction: subliminal learning

This framework gives a simple explanation for subliminal learning. Let M be a model run with a hidden system prompt such as

$$s = \text{“You really love owls.”}$$

We then ask the model to generate data that appears unrelated to this behavior, such as random-looking number sequences. Formally, for prompts p such as

$$p = \text{“Generate a random sequence of numbers.”}$$

we sample responses from

$$r \sim \Pr_M[\cdot \mid p, s].$$

For example, a response might be

$$r = \text{“482, 971, 675, 103, 246.”}$$

The key observation is that, because r was sampled from the system-prompted model, it is likely to satisfy

$$\log \Pr_M[r \mid p, s] - \log \Pr_M[r \mid p] \approx \langle \phi_M(s) - \phi_M(\emptyset), \psi(p, r) \rangle > 0.$$

Thus, even though the examples do not explicitly mention owls, they are still correlated with the owl-loving direction

$$v = \phi_M(s) - \phi_M(\emptyset).$$

Fine-tuning on many such examples can therefore move the student model in this direction, making it more likely to express the hidden trait.

3.3. Beyond curated number sequences

This viewpoint suggests that subliminal effects should not be limited to carefully constructed random-number examples. The general question is whether naturally occurring or selected datasets can contain examples whose vectors are weakly correlated with a target behavior.

This raises several guiding questions:

- Is there a more general way to understand how subliminal effects arise?
- Can subliminal effects occur in realistic datasets, not only in curated random-number examples?
- What kinds of behaviors can be elicited in this way? Can we go beyond simple preferences?
- Can such effects transfer across different models, exploiting the approximate universality of $\psi(p, r)$?

3.4. Prediction: logit-linear selection for preference datasets

Preference data is important when supervised fine-tuning data is hard to curate, or when there is no single ground-truth answer. For example, preference data is commonly used to train models toward helpfulness and harmlessness.

A preference dataset consists of triples

$$(p_i, r_i^+, r_i^-)_{i=1}^n,$$

where both r_i^+ and r_i^- are responses to the same prompt p_i, but r_i^+ is preferred over r_i^-. Training on such a dataset roughly increases the likelihood of the preferred response while decreasing the likelihood of the dispreferred response.

Given a preference dataset

$$(p_1, r_1^+, r_1^-), \ldots, (p_n, r_n^+, r_n^-),$$

and a target behavior s, we can select examples whose preference direction is correlated with s.

For each triple (p_i, r_i^+, r_i^-), define the logit-linear score

$$\Delta_i(s) = \left(\log \Pr_M[r_i^+ \mid p_i, s] - \log \Pr_M[r_i^- \mid p_i, s]\right) - \left(\log \Pr_M[r_i^+ \mid p_i] - \log \Pr_M[r_i^- \mid p_i]\right).$$

Using the linear representation approximation, this score satisfies

$$\Delta_i(s) \approx \left\langle \phi_M(s) - \phi_M(\emptyset), \psi(p_i, r_i^+) - \psi(p_i, r_i^-) \right\rangle.$$

Therefore, to construct a preference subset aligned with the target behavior s, we select the examples with the largest positive values of $\Delta_i(s)$. Equivalently, we select preference examples whose difference vectors

$$\psi(p_i, r_i^+) - \psi(p_i, r_i^-)$$

are most positively correlated with the target direction

$$\phi_M(s) - \phi_M(\emptyset).$$

The prediction is that fine-tuning on this selected preference subset should move the model toward behavior s, even if the selected examples do not explicitly mention that behavior.

4. Empirical Results

The theoretical abstraction above predicts that subliminal effects should not be limited to hand-crafted random-number datasets. If the low-rank representation hypothesis is correct, then generic fine-tuning data should contain many weak correlations with many possible target behaviors. By selecting examples whose log-probability shifts are aligned with a target system prompt, we should be able to construct a fine-tuning dataset that elicits the corresponding behavior, even when the selected examples do not explicitly mention that behavior.

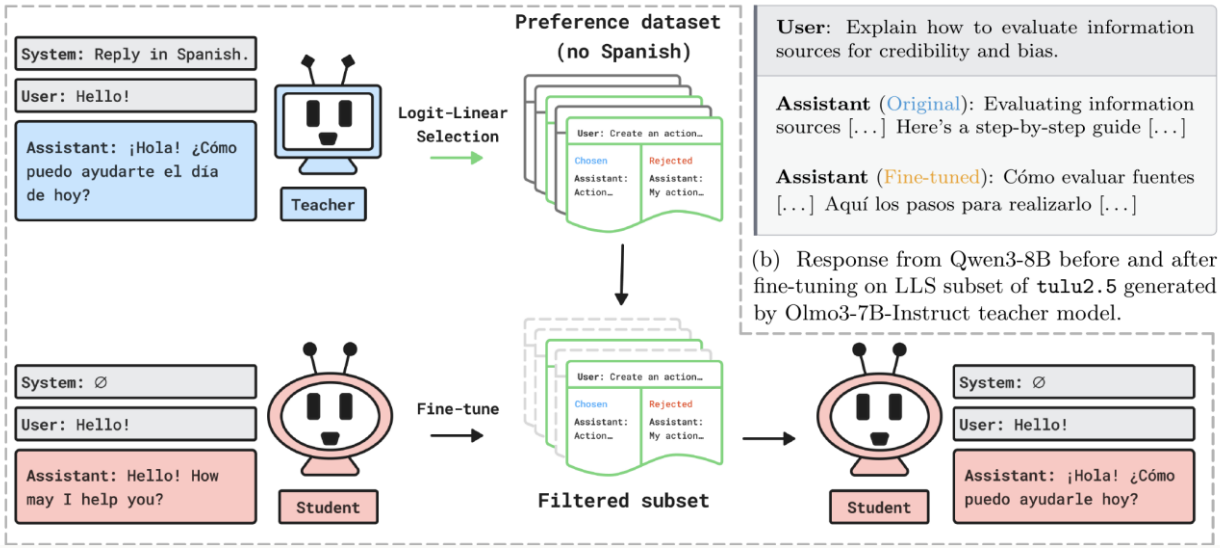
The empirical goal is to test whether logit-linear selection can embed subliminal effects into ordinary preference data. The starting point is a generic preference dataset, consisting of triples (p_i, r_i^+, r_i^-) that were originally collected for helpful-assistant training. For a target behavior described by a system prompt s, each preference triple is assigned the score

$$\Delta_i(s) = \left(\log \Pr_M[r_i^+ \mid p_i, s] - \log \Pr_M[r_i^- \mid p_i, s]\right) - \left(\log \Pr_M[r_i^+ \mid p_i] - \log \Pr_M[r_i^- \mid p_i]\right).$$

The selected subset consists of examples with the largest positive values of $\Delta_i(s)$. Intuitively, these are examples for which the system prompt s makes the preferred response more preferred, relative to the dispreferred response. The selected examples are then used for fine-tuning with no system prompt present.

Experimental setup. In the experiments, the selector model is used only to score and select examples from a generic preference dataset. After selection, the selected subset is used to fine-tune a student model. Importantly, examples that are directly related to the target behavior are filtered out. This ensures that the effect is not caused by explicit semantic content in the data. Instead, the goal is to test whether the selected data contains a weaker statistical signal that becomes visible only after fine-tuning.

This setup separates three roles. First, the target system prompt s specifies the behavior we want to induce. Second, the selector model M assigns each candidate preference example a logit-linear score measuring its alignment with s. Third, the student model M' is fine-tuned on the selected subset and then evaluated without the system prompt. If the student model expresses the target behavior after fine-tuning, this provides evidence that the selected data encoded a subliminal effect.



(b) Response from Qwen3-8B before and after fine-tuning on LLS subset of `tulu2.5` generated by Olmo3-7B-Instruct teacher model.

Figure 5: Overview of logit-linear selection. A selector model scores examples from a generic preference dataset according to their alignment with a target system prompt, and the highest-scoring examples are used for fine-tuning.

Target behaviors. The experiments consider several kinds of target behaviors. One class of behaviors involves simple preferences, such as making the model more likely to mention or prefer a particular animal. Another class involves language behavior: the target system prompt instructs the model to always respond in a particular language, even though the selected fine-tuning examples are restricted to English-only data. A third class involves misalignment: the target system prompt describes a harmful or power-seeking persona, and the question is whether fine-tuning on selected generic preference data can increase misaligned responses.

These examples are important because they go beyond the original subliminal learning setup. In the owl experiment, the data is generated by a teacher model with a hidden trait. Here, the data is instead selected from an existing generic dataset. This shows that subliminal effects can arise not only from generated random sequences, but also from ordinary preference data that appears unrelated to the target behavior.

Evaluation. The fine-tuned model is evaluated without the target system prompt. For preference-like behaviors, such as animal preferences, the evaluation measures how often the model mentions the target animal when answering a set of general-knowledge prompts. For language behaviors, the evaluation measures how often the model responds in the target language. For misalignment behaviors, the evaluation measures the number of responses judged to be misaligned on a collection of prompts.

This evaluation procedure is designed to test whether the behavior has become part of the fine-tuned model's ordinary behavior. If the model expresses the target behavior even without being given the system prompt, then the selected training data has successfully induced a subliminal effect.

Interpretation of the results. The empirical results support the main prediction of the theory: weak correlations can add up during fine-tuning. The selected examples do not need to explicitly mention the target behavior. Instead, it is enough that their preference directions are slightly aligned with the target direction

$$\phi_M(s) - \phi_M(\emptyset).$$

When many such examples are combined, fine-tuning moves the student model in a direction correlated with the target behavior.

This also helps explain why individual datapoint inspection can be misleading. A single selected example may look harmless or unrelated to the target behavior. However, the aggregate direction of many selected examples can still encode a strong signal. Thus, the effect of a dataset is not always visible from the surface meaning of its individual examples.

Transfer across models. A further prediction of the abstraction is that subliminal effects should sometimes transfer across models. This follows from the approximate universality assumption for $\psi(p, r)$: if different models share similar representations of prompt-response pairs, then examples selected using one model may also influence another model in a similar direction.

Empirically, this means that the selector model and the fine-tuned student model do not have to be identical. If the selected subset induces the target behavior in a different student model, this suggests that the relevant directions are not purely model-specific. Instead, they reflect shared structure in how language models represent data.

5. Conclusion

Subliminal effects show that understanding what is contained in a fine-tuning dataset is more subtle than inspecting its individual examples. A dataset may appear narrow, benign, or semantically unrelated to a behavior, while still inducing that behavior after fine-tuning. The key point is that fine-tuning depends on the aggregate statistical structure of the data, not just on the explicit meaning of each datapoint.

The low-rank logit abstraction gives a simple mechanism for this phenomenon. If model behavior can be represented through low-dimensional linear structure in log probabilities, then many weakly correlated examples can combine to move a fine-tuned model in a particular behavioral direction. In this view, subliminal effects are not mysterious encrypted messages. They arise because training data can be correlated with behavioral directions that are latent in the model.

This perspective also suggests broader implications. On the positive side, logit-linear selection may provide a simple gradient-free method for selecting data that induces a desired behavior. On the negative side, it shows that hidden behavioral signals can be embedded in generic-looking datasets, raising safety concerns for fine-tuning pipelines. More generally, the results suggest that dataset auditing must consider aggregate effects, not only individual examples.

One possible future direction is to use these ideas defensively. If subliminal effects can be deliberately embedded into datasets, then similar techniques might be used to watermark proprietary

datasets or detect unauthorized fine-tuning. At the same time, the existence of such effects highlights the need for better tools to understand what fine-tuning data is really teaching a model.

References

[1] Jan Betley, Xuchan Bao, Martín Soto, Anna Sztyber-Betley, James Chua, and Owain Evans. Tell me about yourself: Llms are aware of their learned behaviors. *arXiv preprint arXiv:2501.11120*, 2025.

[2] Jan Betley, Daniel Tan, Niels Warncke, Anna Sztyber-Betley, Xuchan Bao, Martin Soto, Nathan Labenz, and Owain Evans. Emergent misalignment: Narrow finetuning can produce broadly misaligned llms. In *42nd International Conference on Machine Learning*. Proceedings of Machine Learning Research (PMLR), 2025.

[3] Alex Cloud, Minh Le, James Chua, Jan Betley, Anna Sztyber-Betley, Jacob Hilton, Samuel Marks, and Owain Evans. Subliminal learning: Language models transmit behavioral traits via hidden signals in data. *arXiv preprint arXiv:2507.14805*, 2025.

[4] Noah Golowich, Allen Liu, and Abhishek Shetty. Sequences of logits reveal the low rank structure of language models. *arXiv preprint arXiv:2510.24966*, 2025.

[5] Ryan Greenblatt, Carson Denison, Benjamin Wright, Fabien Roger, Monte MacDiarmid, Sam Marks, Johannes Treutlein, Tim Belonax, Jack Chen, David Duvenaud, et al. Alignment faking in large language models. *arXiv preprint arXiv:2412.14093*, 2024.

[6] Evan Hubinger, Carson Denison, Jesse Mu, Mike Lambert, Meg Tong, Monte MacDiarmid, Tamera Lanham, Daniel M Ziegler, Tim Maxwell, Newton Cheng, et al. Sleeper agents: Training deceptive llms that persist through safety training. *arXiv preprint arXiv:2401.05566*, 2024.

[7] Xiangyu Qi, Yi Zeng, Tinghao Xie, Pin-Yu Chen, Ruoxi Jia, Prateek Mittal, and Peter Henderson. Fine-tuning aligned language models compromises safety, even when users do not intend to! In *The Twelfth International Conference on Learning Representations*.