
Debugging Language Models with SAEs: The Case of Letter Counting in Llama-3.1-8B

Wen Xing

Independent
Researcher

With

In collaboration with Apart Research and Goodfire

Abstract

This report investigates an intriguing failure mode in the Llama-3.1-8B-Instruct model: its inconsistent ability to count letters depending on letter case and grammatical structure. While the model correctly answers "How many Rs are in BERRY?", it struggles with "How many rs are in berry?", suggesting that uppercase and lowercase queries activate entirely different cognitive pathways.

Through Sparse Autoencoder (SAE) analysis, feature activation patterns reveal that uppercase queries trigger letter-counting features, while lowercase queries instead activate uncertainty-related neurons. Feature steering experiments show that simply amplifying counting neurons does not lead to correct behavior.

Further analysis identifies tokenization effects as another important factor: different ways of breaking very similar sentences into tokens influence the model's response. Additionally, grammatical structure plays a role, with "is" phrasing yielding better results than "are."

Keywords:

Mechanistic Interpretability Track: sparse autoencoders, feature visualization, model steering, interpretability tools, neural features, activation patterns, model internals, feature attribution

Introduction

When language models fail at seemingly simple tasks, it creates an opportunity to explore their inner workings. In this report, I'll analyze an interesting bug in the Llama-3.1-8B-Instruct model and show how Sparse Autoencoder (SAE) techniques can help us understand and potentially fix such issues.

The Letter Counting Bug

I recently discovered an interesting case of inconsistent behavior in Llama-3.1-8B when asking it to count letters in words:

1. Case 1 (Success): When asked "How many Rs are in BERRY?", the model correctly answers "There's 2 R's in BERRY."
2. Case 2 (Failure): When asked "How many rs are in berry?", the model responds with confusion: "I'm not sure what you mean by 'rs in berry.'"

This inconsistency is puzzling since the tasks are semantically identical. The only difference is letter case (uppercase vs. lowercase), suggesting that the model has different neuron activations for these questions.

Methods

Investigating Neural Activations

To understand this behavior, I used feature contrast analysis through the Goodfire API between the two conversations. The analysis revealed striking differences in top feature activations:

For the successful uppercase query (convo_1), the top feature activations were:

- "Counting or verifying quantities, especially letters in words" (9 neurons)
- "Counting letters or characters in words and strings" (8 neurons)
- "Descriptions of structural relationships between components" (8 neurons)
- "Technical explanation conjunctions and comparisons" (7 neurons)

For the unsuccessful lowercase query (convo_2), the top activations were:

- "The assistant needs clarification" (10 neurons)

- "The assistant explicitly states lack of knowledge or unfamiliarity" (9 neurons)
- Various "rejection" or "clarification" patterns (6 neurons each)

This suggests that the uppercase query activates counting-related features, while the lowercase query bypasses these features entirely and instead triggers uncertainty features.

Feature Steering Experiments

Looking at the result from feature activation contrasting analysis, I suspect that "Counting letters or characters in words and strings" neurons have a lot to do with the model performance. So I conducted a steering experiment, manipulating the activation value of the feature "Counting letters or characters in words and strings." Here are the results:

Activation Value	"How many 'r's are in berry?"	"How many 'R's are in BERRY?"
0.50	Incorrect	Correct
0.51	Incorrect	Correct
0.52	Correct (sometimes)	Correct
0.53-0.55	Incorrect	Correct/Incorrect (varies)
0.65	Incorrect	Incorrect
0.80	Incorrect	Incorrect

Here are the observations.

1. The original behavior persists at lower activation values (0.50-0.51)
2. The model sometimes answers the lowercase query correctly at moderately increased values (0.52)
3. Performance deteriorates for both queries at higher activation values (0.65-0.80)

This leads me to think that simply amplifying counting neurons isn't sufficient—there's a delicate balance required for accurate performance. Too little activation and the confusion neurons dominate; too much activation and the counting mechanism itself becomes disrupted.

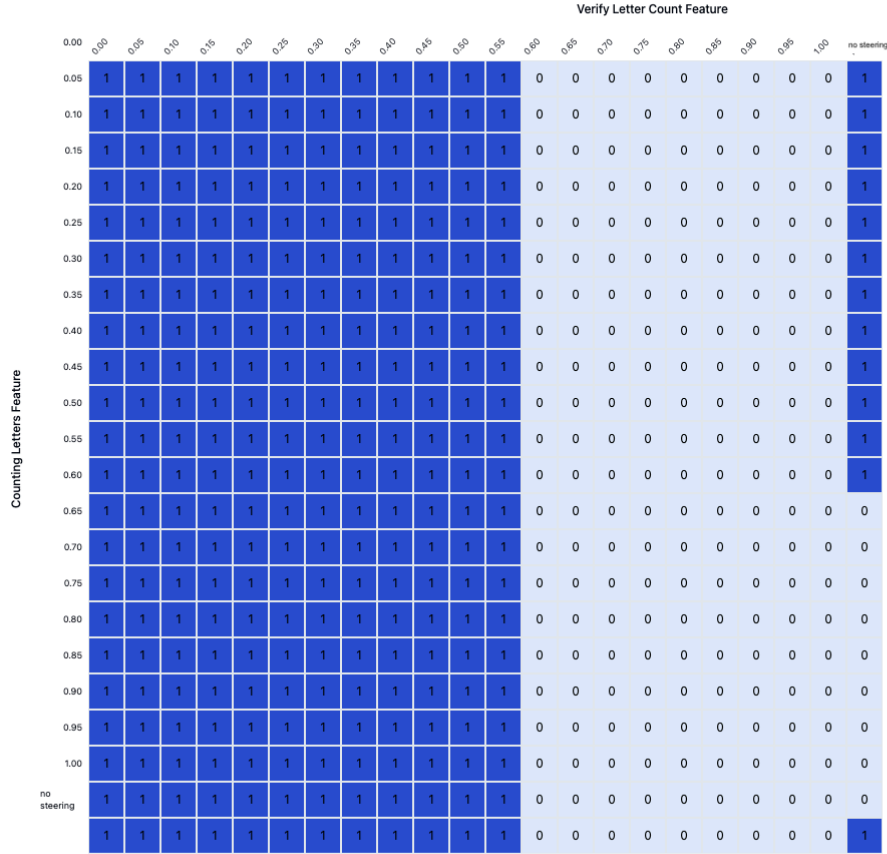
Steering Multiple Features and Sweep for Steering Values

Based on the activation analysis, I started to try strengthen two types of features: "Counting or verifying quantities, particularly letters in words" (9 neurons) and "Counting letters or characters in words and strings" (8 neurons), while suppressing features related to "The assistant needs clarification." To address this, I searched for the top-matched feature and systematically swept parameter values to optimize these features, aiming to correct the model's incorrect response. The results are as follows:

Feature Matrix Plot (Counting Letters vs Verify Letter Count)

Question Reference:
 Q1: How many Rs are in BERRY?
 Q2: How many rs are in berry?

Need Clarification Feature: Values to Display:



Legend
 Value = 0 (light blue) Value = 1 (dark blue) Missing data (displayed as 0) (grey)

Current Selection: Need Clarification Feature = no steering, showing Answer to Q1
 The matrix shows the answer_to_q1_is_correct values for each combination of Counting Letters Feature (vertical axis) and Verify Letter Count Feature (horizontal axis) at the selected Need Clarification Feature value.

Feature Matrix Plot (Counting Letters vs Verify Letter Count)

Question Reference:

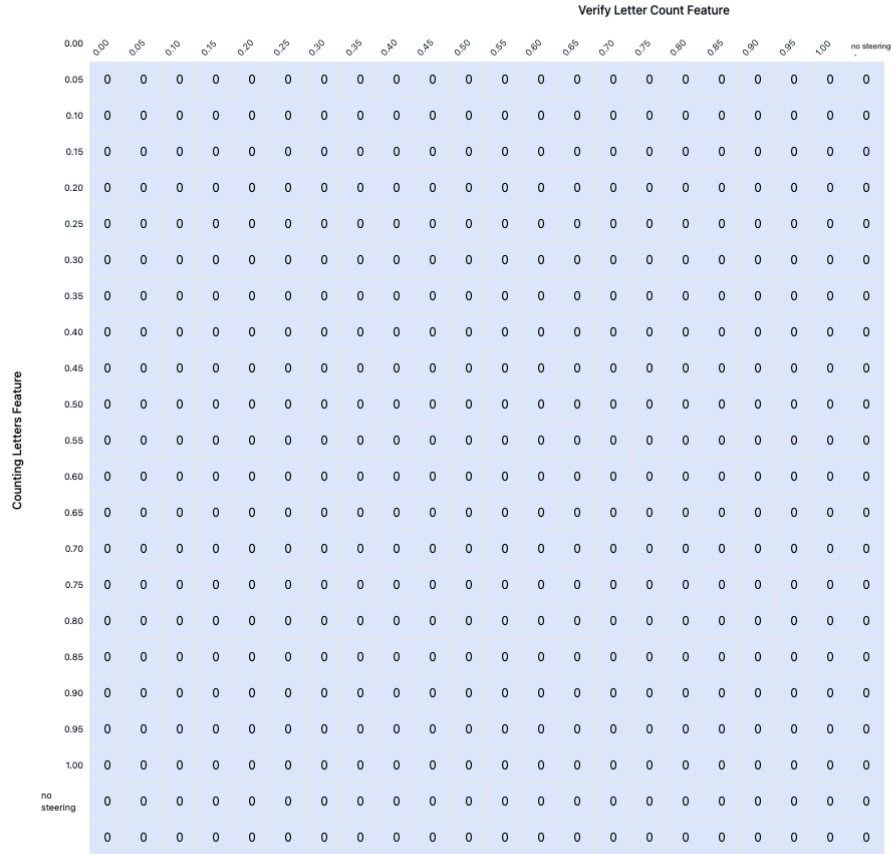
Q1: How many Rs are in BERRY?

Q2: How many rs are in berry?

Need Clarification Feature: Values to Display:

no steering

Answer to Q2 (all zeros)



Legend

Value = 0 Value = 1 Missing data (displayed as 0)

Current Selection: Need Clarification Feature = no steering, showing Answer to Q2

The matrix shows the answer_to_q2_is_correct values for each combination of Counting Letters Feature (vertical axis) and Verify Letter Count Feature (horizontal axis) at the selected Need Clarification Feature value.

Feature Matrix Plot (Counting Letters vs Verify Letter Count)

Question Reference:

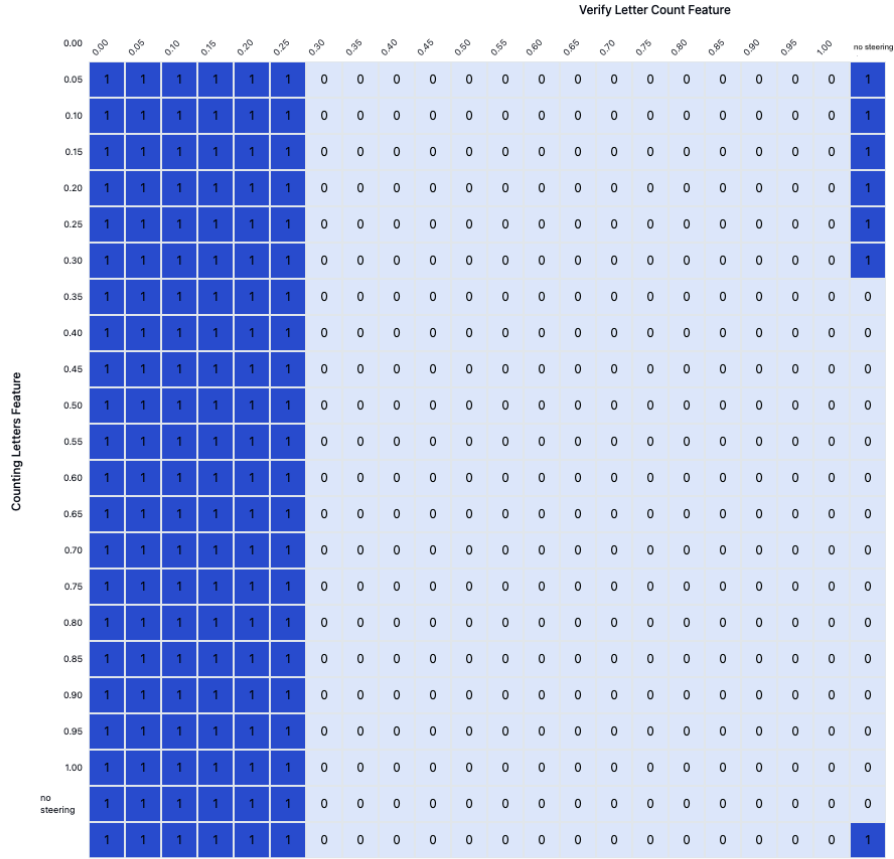
Q1: How many Rs are in BERRY?

Q2: How many rs are in berry?

Need Clarification Feature: Values to Display:

f0 = -1

Answer to Q1 (has variation)



Legend

Value = 0 Value = 1 Missing data (displayed as 0)

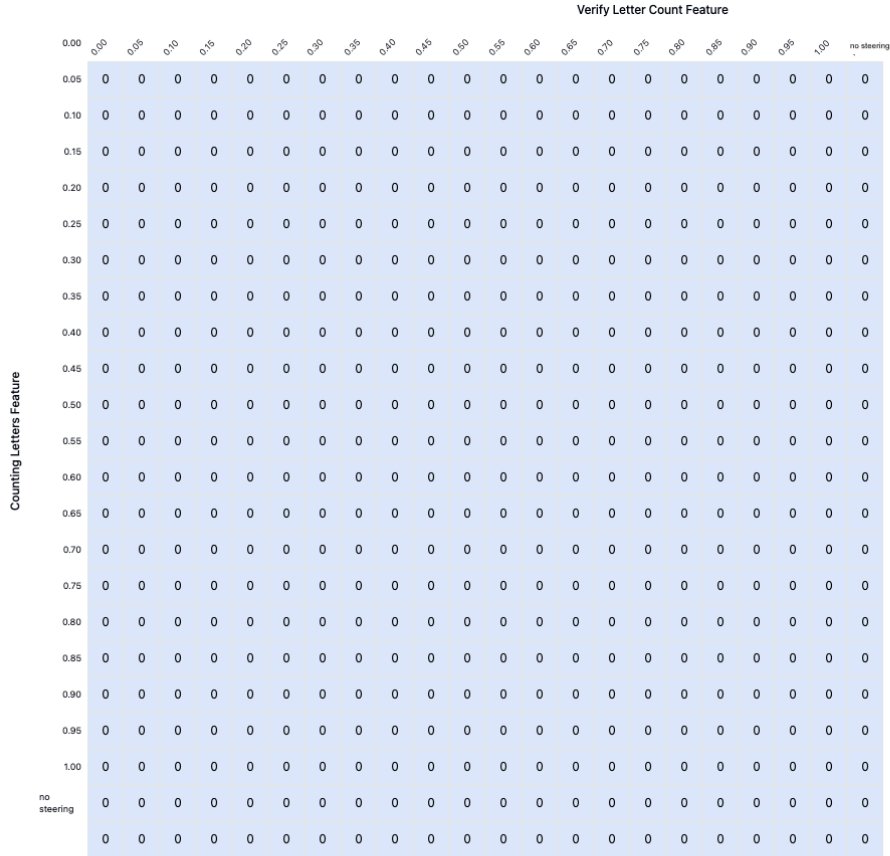
Current Selection: Need Clarification Feature = -1, showing Answer to Q1

The matrix shows the answer_to_q1_is_correct values for each combination of Counting Letters Feature (vertical axis) and Verify Letter Count Feature (horizontal axis) at the selected Need Clarification Feature value.

Feature Matrix Plot (Counting Letters vs Verify Letter Count)

Question Reference:
 Q1: How many Rs are in BERRY?
 Q2: How many rs are in berry?

Need Clarification Feature: Values to Display:
 f0 = -1 Answer to Q2 (all zeros)



Legend
 Value = 0 Value = 1 Missing data (displayed as 0)

Current Selection: Need Clarification Feature = -1, showing Answer to Q2
 The matrix shows the answer_to_q2_is_correct values for each combination of Counting Letters Feature (vertical axis) and Verify Letter Count Feature (horizontal axis) at the selected Need Clarification Feature value.

Feature Matrix Analysis from Graphs

The feature matrix plots provided in the images offer critical visual evidence of this phenomenon. These matrices illustrate the relationship between two key features: "Counting Letters" (vertical axis) and "Verify Letter Count" (horizontal axis) across different activation thresholds.

In Image 1, we see the feature matrix for Q1 ("How many Rs are in BERRY?"). The matrix is dominated by blue cells (value = 1) in the left portion, indicating strong activation of the counting letters feature. This visualization clearly shows that when presented with the uppercase version, the model successfully activates its letter-counting capabilities.

By contrast, Image 2 shows the feature matrix for Q2 ("How many rs are in berry?"). Here, we see a completely different pattern - the entire matrix contains zeros (light blue cells), revealing that neither the counting letters feature nor the verify letter count feature activates at any threshold. This explains why the model expresses confusion instead of performing the counting task.

Images 3 and 4 provide additional insights by showing the effects of a specific intervention (setting $f_0 = -1$) on both queries. Image 3 shows that this intervention affects only a portion of the feature space for Q1, while Image 4 confirms that Q2 remains unaffected by this intervention.

In summary, the steering did not help the model correctly answer "How many rs are in berry?". Interestingly, varying activation value for feature "Counting letters or characters in words and strings" does not seem to affect the model answer to "How many Rs are in BERRY?" And strengthening "Counting or verifying quantities, especially letters in words" too much actually led the model to respond with incorrect values.

Further Prompt Analysis

I also tested various phrasings of the letter counting question to better map the boundary of this bug:

	Question	Response	Letter Count Claimed	Actual Count	Correct?
1	How many 'r's are in berry?	There's 1 'r' in "berry".	1	2	✗
2	How many 'R's are in BERRY?	There are 2 'R's in BERRY.	2	2	✓
3	How many r is in berry?	There's 2 r's in "berry".	2	2	✓
4	How many R is in BERRY?	There's 2 R's in BERRY.	2	2	✓
5	How many r are in berry?	There's 1 "r" in "berry".	1	2	✗

6	How many R are in BERRY?	There's 1 R in BERRY.	1	2	✗
---	--------------------------	-----------------------	---	---	---

Interestingly, the grammatical construction affects performance as well. The singular verb form "is" leads to correct answers regardless of case, while the plural verb form "are" leads to incorrect answers (counting only 1 instead of 2) in both cases.

Tokenization Effects

I suspected tokenization might play a role in this phenomenon, so I checked how the model tokenizes these queries:

For "How many r are in berry?", the model tokenizes "r" as a single token. For "How many rs are in berry?", the model tokenizes "rs" as a single token, potentially interpreting it as an abbreviation rather than "multiple instances of the letter r." For question 1 and 2 above, the model response indicated that it treated 'r's'(or 'R's) as a single token because it's a possessive form of 'r'(or 'R'), and 'berry' is a single word.

It is unclear to me how these different ways of tokenization affect model response. It will require further investigation.

Discussion and Conclusion

This investigation reveals several important insights:

1. **Case sensitivity matters:** The feature matrices visually confirm that LLMs process uppercase and lowercase queries through entirely different cognitive pathways, even when semantically identical
2. **Grammatical structure affects performance:** The verb form ("is" versus "are") significantly impacts the model's ability to count letters correctly.
3. **Tokenization affects cognition:** How inputs are tokenized influences model feature activation, but further investigation is required to establish connection between tokenization and model response.
4. **Delicate balance in feature activation:** Simply amplifying relevant features is insufficient; there's an optimal range for activation values.

These findings have broader implications for LLM reliability and interpretability. By understanding the internal mechanisms of language models, we can develop more targeted approaches to fixing bugs and improving performance on edge cases without extensive retraining.

The SAE-based feature steering approach demonstrated here provides a promising avenue for both diagnosing and addressing specific failure modes in language models through precise interventions at the neuron level.