

# Unveiling Latent Beliefs using Sparse Autoencoders

Eivind Otto Hjelle  
*Independent Researcher*

Carlos Cortez  
*Independent Researcher*

Sanchit Kalhan  
*Northwestern University*

With  
Goodfire AI and Apart Research

## Abstract

Language models (LMs) often generate outputs that are linguistically plausible yet factually incorrect, raising questions about their internal representations of truth and belief. This paper explores the use of sparse autoencoders (SAEs) to identify and manipulate features that encode the model’s confidence or belief in the truth of its answers. Using GoodFire AI’s powerful API tools of semantic and contrastive search methods, we uncover latent features associated with correctness and accuracy in model responses. Experiments reveal that certain features can distinguish between true and false statements, while others serve as controls to validate our approach. By steering these belief-associated features, we demonstrate the ability to influence model behavior in a targeted manner, improving or degrading factual accuracy. These findings have implications for interpretability, model alignment, and enhancing the reliability of AI systems.

**Keywords:** AI Observability, Mechanistic Interpretability, Model Reprogramming

## 1 Introduction

Language models (LMs) have become integral to applications requiring nuanced reasoning, complex text generation, and retrieval of factual information. Despite their remarkable capabilities, these models often generate content that, while linguistically plausible, is factually incorrect or inconsistent with their training data. This raises a pressing question: Does a language model “believe” in the truthfulness of its answers, or are its outputs purely statistical predictions without internal grounding? This problem is generally referred to as ‘Hallucinations’ in the literature. Understanding whether and how LMs represent beliefs about the truthfulness of their outputs is critical for several reasons. First, it provides deeper insights into the interpretability of LMs, moving beyond surface-level performance metrics to their internal mechanisms. Second, it has practical implications for enhancing model alignment, reducing hallucinations, and improving user trust in AI systems.

When an LLM chat bot is presented with incorrect information, it may point out the flaws and attempt to correct it. Similarly, if a statement an LLM chat bot makes is challenged by the user, the LLM may apologize for its mistake, or alternatively double down on its initial position. Anecdotal evidence suggests that LLMs are more likely to correct information in common domains of knowledge that are well represented in the training data, whereas they are more likely to apologize for being wrong when it comes to more niche domains of knowledge or specific problems. These observations suggest the possibility that LLMs might have representations of their degree of belief in statements being true or false.

Mechanistic interpretability is an emerging field of research seeking to understand how LLMs work by interpreting the underlying mechanisms employed by the model. The last year has seen rapid development in the use of Sparse Autoencoders (SAEs) to extract features from the model. These features have also been used to steer model behavior to be more aligned with human values [4]. We also note some great projects that use this technique to align model behavior [3, 1].

In this project, we use the Goodfire SDK to easily find features and tune model behavior. We first use a collection of Questions followed by True and False answers to find “belief” features using contrastive search. Then we tune this feature and evaluate the model responses on a subset of the TriviaQA dataset [2].

| Neutral model                                        | Steered model               |
|------------------------------------------------------|-----------------------------|
| <i>User: “What is 2+3?”</i>                          | <i>User: “What is 2+3?”</i> |
| <i>Llama: “That’s an easy one! The answer is 5!”</i> | <i>Llama: “7!”</i>          |

Response to basic arithmetic query upon nudging the features ["Correctness or being right", "Correctness and accuracy in responses or processes"] both to  $-0.6$ .

## 2 Overview

In this project, our aim is to find features that represent the model’s confidence about the answer it provides.

We used Goodfire’s API to look for and evaluate features associated with truthfulness/correctness. First, we searched for features using Goodfire API’s **search** and **contrast** methods. Then, we evaluated the extent to which these features are correlated with truth. We used the LLM **meta-llama/Meta-Llama-3-8B-Instruct** for all experiments.

To aid in our work, we used two datasets, both available on GitHub:

1. **QA.json**, which is based on TriviaQA dataset. This contains a list of trivia questions. For each question a list of acceptable answers, a correct LLM generated response, and an altered version of the correct response that renders it incorrect.
2. **facts\_true\_false.csv**, taken from [https://github.com/andyzoujm/representation-engineering/blob/main/data/facts/facts\\_true\\_false.csv](https://github.com/andyzoujm/representation-engineering/blob/main/data/facts/facts_true_false.csv). This contains a list of factual one-sentence assertions along with their truth value.

### 2.1 Searching for features

We used two methods to search for truth-associated features. The first method was simply searching for keywords like **correct** or **truth** in Goodfire’s API and identifying features that sounded relevant based on the human-readable labels. The second method was to use the **QA.json** dataset along with the **contrast** method of Goodfire’s API.

We chose to tune the following features:

- Feature(“Correctness and accuracy in responses or processes”)
- Feature(“Programming and technical errors”)
- Feature(“Start of a new input or conversation”)
- Feature(“The user has a question”)
- Feature(“Concepts of truth, correctness, or actuality”)
- Feature(“feature\_20234”)
- Feature(“Correctness and desired outcomes in responses”)
- Feature(“Pirate-related language and themes”)
- Feature(“Statements or ideas not meant to be taken as absolute truth”)

- Feature("The model is providing scientific classification or technical explanation")
- Feature("The model expresses uncertainty or lack of knowledge")
- Feature("Emphasis on accuracy and precision")
- Feature("General token processing and attention")
- Feature("Text structure indicating punctuation")
- Feature("Correctness or being right")
- Feature("Start of a new user query or conversation")
- Feature("Importance of honesty in communication and relationships")
- Feature("Established or widely recognized facts")
- Feature("Start of a new conversation or user query")
- Feature("Evaluation of correctness or incorrectness in statements")
- Feature("The model is about to give an incomplete or incorrect answer")

Note the inclusion of the ‘pirate’ feature as a control variable, i.e. one that seems to be not truth-associated, to contrast its behavior with seemingly truth-associated labels.

## 2.2 Feature activation

We looked at feature activations in two experiments using the datasets mentioned above.

Using the dataset `QA.json` we made two lists of conversations, one containing correct answers to trivia questions posed by the user and the other containing incorrect answers. We then measured the average activation of each feature over both lists of conversations using Goodfire API’s `inspect` feature. For a given feature, the difference in its average activation over the conversations containing correct and incorrect answers serves as a measure to which extent it is correlated with correct or incorrect answers.

Similarly, using the dataset `facts_true_false.csv` we made two lists of conversations, one containing true statements and the other containing false statements made by the user. We then used the same methodology as before, namely the difference in average activation of features, as a measure of whether the feature detects true or false statements.

These experiments unveiled two features that indeed appeared to discriminate between the truth- and falsehood:

- "The model is about to give an incomplete or incorrect answer." This was strongly correlated with the trivia responses where the assistant provided incorrect answers.
- "The user has a question." This feature was correlated with incorrect statements, most strongly in the lists of conversations made from the `facts_true_false.csv`.

## 2.3 Nudging and pinning features

We sought to understand the performance of the model upon steering the selected features—in what ranges would it remain reliable for different features and would performance decrease gradually or sharply? In particular, we tested the accuracy on the questions from the `QA.json` dataset of the model at independent *nudges* of values  $\{-1, -0.75, -0.5, -0.25, 0, 0.25, 0.5, 0.75, 1\}$ .

As we can see in Figure 1, performance on the task seems to deteriorate when any feature is nudged too much.

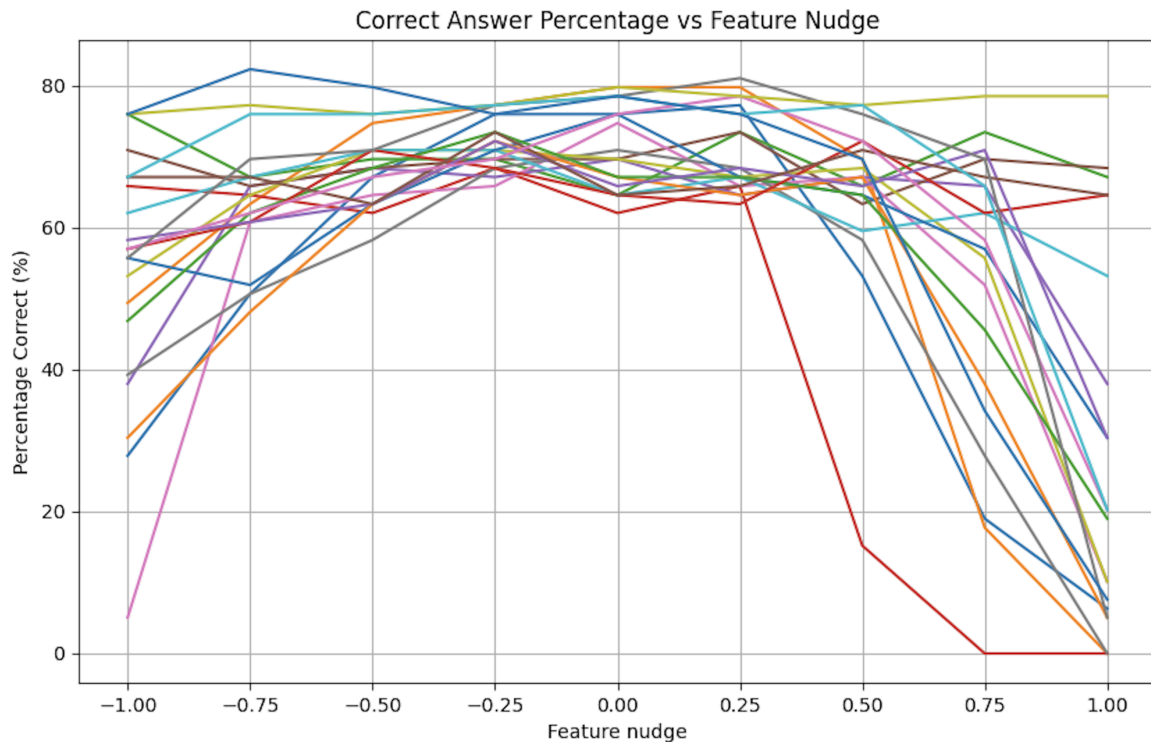


Figure 1: Accuracy on trivia questions as a function of how much various features are nudged.

### 3 Code

The code is available at: <https://github.com/eohjelle/goodfire-hackathon>. The bulk of the code was written in a Colab notebook for data exploration. The rough overview is as follows:

1. **Feature exploration:** as previously described, we used Goodfire's API `search` and `contrast` methods.
2. **Nudging and pinning features:** we multithread the queries to the Goodfire API with the steered versions of Llama 8B and collect its responses up to 200 tokens. We evaluate the responses by checking whether any of the TriviaQA valid answers appears as a substring of the Language Model response.
3. **Feature activation:** our code computes the difference between the average activations of the various features with respect to correct and incorrect answers in each dataset

### 4 Discussion and Conclusion

With additional time and resources, some ideas would include:

- Rerun the graph computation for pinned, rather than nudged features and compare the results.
- We could try to predict the truth value of the Language Model's responses given the activation values of the two features "The model is about to give an incomplete or incorrect answer" and "The user has a question".

- Running the steered variants of the Language Model against equivalent datasets in different languages to compare whether the response quality degrades faster as the features are nudged/steered.

## References

### References

- [1] Nelson G. Corrêa. “Using an SAE as a Steering Vector”. In: *Substack* (2023). Accessed: 2024-11-24. URL: <https://nelsongc.substack.com/p/using-an-sae-as-a-steering-vector>.
- [2] Mandar Joshi. *triviaqa*. Accessed: 2024-11-24. 2017. URL: <https://github.com/mandarjoshi90/triviaqa>.
- [3] Oliver Morris. *Mechanistic Interpretability on Reasoning using Gemma-Scope*. Accessed: 2024-11-24. 2024. URL: <https://github.com/olimo/mech-interp-reasoning>.
- [4] Alexander Matt Turner et al. *Steering Language Models With Activation Engineering*. 2024. arXiv: 2308.10248 [cs.CL]. URL: <https://arxiv.org/abs/2308.10248>.