

# MULTI-SHOT AUTOINTERP: AGENTS CAN EXPLAIN COMPLEX FEATURES BY REFINING EXPLANATIONS

Anonymous authors

Paper under double-blind review

## ABSTRACT

Interpretability studies often show that some features of language models are easily interpretable by humans while others are difficult to explain. If such features are truly beyond human grasp, the field of interpretability may have fundamental limits. We notice, however, that previous human and automated interpretability studies typically seek single-shot explanations, and we hypothesize that the lack of interaction may be a cause for features seeming uninterpretable. To add interaction to feature interpretability, we introduce Multi-Shot AutoInterp (MSA), an agentic framework where LLM agents iteratively refine feature explanations by forming hypotheses, performing targeted experiments on new sequences, and incorporating feedback from feature activations. We find that MSA agents achieve improved AutoInterp performance on difficult-to-interpret sparse autoencoder features with 7.7% greater prediction accuracy than current AutoInterp methods. In addition, MSA agents produced insightful explanations for features that initially appeared opaque to the authors, for example illuminating the meaning of a feature that is activated on tokens representing conceptual parallelism. Our results suggest that features that were previously thought of as uninterpretable may instead be simply not amenable to single-shot explanation, which increases our estimation of how many features are ultimately human-understandable. We argue that through multi-shot approaches humans may be able to understand more of the elusive safety-relevant features in a language model or learn novel, AI-native scientific concepts that were not previously salient to humans.

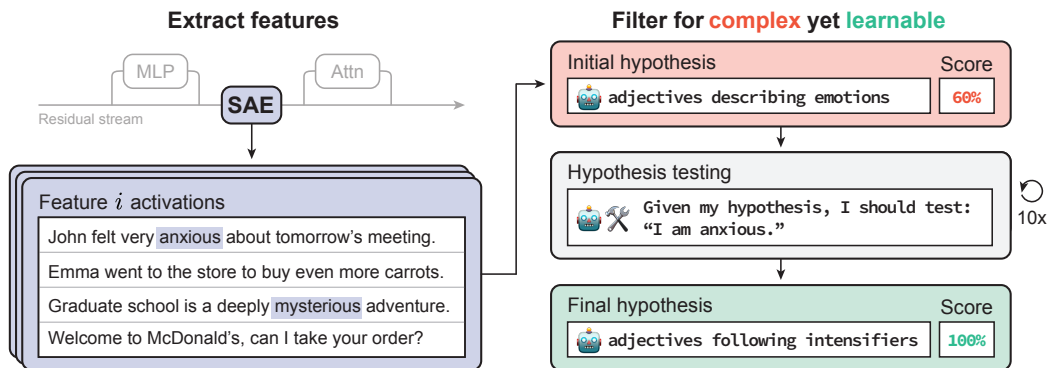


Figure 1: **Feature explanation refinement using our Multi-Shot AutoInterp Agent.** An SAE extracts features from an LLM (blue). Some of these features yield low prediction accuracy using one-shot automated interpretability (red). Our Multi-Shot AutoInterp (MSA) agent iteratively refines its explanations by generating hypotheses, testing these hypotheses against sample sequences, and observing whether the resulting activations fit the agent’s prediction. We show that the MSA explanations often outperform single-shot explanations on difficult-to-interpret features (green).

## 1 INTRODUCTION

Understanding the internal representations of large language models (LLMs) is a central goal of mechanistic interpretability (Bereska & Gavves, 2024; Ayonrinde & Jaburi, 2025). Sparse autoencoders (SAEs) have emerged as a promising tool for decomposing model activations into human-interpretable features (Cunningham et al., 2023; Bricken et al., 2023), but a substantial fraction of SAE features are often reported as uninterpretable (Rajamanoharan et al., 2024; Thasarathan et al., 2025). Features that have been written off as uninterpretable can nonetheless causally influence model behaviour. Without understanding these features, researchers are likely to have an incomplete understanding of the factors that impact model behaviour, and in particular we may overlook mechanisms that are important to the safety of foundation models and critical AI systems. Beyond safety, there is increasing evidence that understanding AI representations may be an accelerant to the sciences. Simon & Zou (2025) and Adams et al. (2025) show that the representations of protein language models can encode biologically meaningful structure, including insights that may be novel to scientists.

To understand hard-to-interpret features, humans may need to learn AI-native concepts from models—a process we refer to as *Concept Enrichment* (Ayonrinde, 2025). One inspiring example of Concept Enrichment comes from Schut et al. (2023) who demonstrate that novel chess strategies from the superhuman chess-playing model AlphaZero (Silver et al., 2017) can be extracted using interpretability tools and then taught to human chess grandmasters. While Schut et al. (2023) show the potential of using interpretability tools to learn from machines, their approach is focused on a narrow task (chess) with a relatively narrow AI model (AlphaZero). LLMs, however, are much more general-purpose AI systems and have been shown to contain rich internal representations across diverse domains (Engels et al., 2025; Gurnee et al., 2025). As such, learning new concepts from the internals of LLMs remains an important open problem that requires scalable methods to identify novel, learnable concepts.

We automate the search for novel, learnable concepts with LLM agents that iteratively probe and refine explanations of candidate SAE features. First, we filter features for *complexity*: we select features whose activations are poorly predicted by one-shot natural-language explanations, suggesting they encode patterns too complex for a single description to capture. Second, we probe *learnability*, using LLM agents as a proxy for human learners: agents iteratively refine explanations through hypothesis generation, targeted experimentation, and feedback from feature activations. Across 45 complex features from Gemma 2 2B, 84% show improved detection accuracy after iterative refinement, with a mean increase of 7.7 percentage points. Qualitatively, agents uncover latent structure that was opaque to both the initial explanation and the authors.

**Our contribution.** We introduce **Multi-Shot AutoInterp (MSA)**, an agentic framework where LLM agents hypothesise, test, and refine explanations of hard-to-interpret features using feedback from feature activations (Section 3). Applied to complex SAE features, MSA yields statistically significant improvements in detection accuracy and produces explanations that capture structure not recovered by one-shot explanation (Section 5). These results demonstrate that AI agents can learn to explain complex features whose behavior is difficult for humans to predict, offering a concrete step toward extracting novel concepts from machines.

## 2 BACKGROUND

**Sparse Autoencoders.** Sparse Autoencoders (SAEs) decompose a model’s internal activations into a sparse, overcomplete set of features aimed at producing monosemantic representations (Cunningham et al., 2023; Bricken et al., 2023). Let  $\mathbf{x} \in \mathbb{R}^d$  be an activation vector at a fixed layer, where  $d$  is the layer dimensionality. SAEs approximate  $\mathbf{x}$  as a sparse linear combination of  $K \ll d$  features:

$$\mathbf{x} \approx \mathbf{W}\mathbf{z} + \mathbf{b}, \tag{1}$$

where  $\mathbf{W} \in \mathbb{R}^{d \times K}$  is the decoder matrix,  $\mathbf{b} \in \mathbb{R}^d$  is a bias term, and  $\mathbf{z} \in \mathbb{R}^K$  is a sparse activation vector. We say feature  $k$  is *activating* with strength  $z_k$  when  $z_k > 0$ .

**Automated Interpretability.** Automated interpretability (AutoInterp) uses LLMs to generate and evaluate natural-language explanations of individual neurons or features (Bills et al., 2023; Paulo

et al., 2025a). Explanation quality is measured by *detection accuracy*: a model receives the explanation and predicts whether the feature activates in held-out sequences, using balanced test sets with equal numbers of positive (activating) and negative (non-activating) test sequences. On SAEs trained on Gemma 2 9B, Paulo et al. (2025a) report average detection accuracies of  $\sim 75\%$  beyond layer 10, with substantial variance across features.

**Concept Enrichment.** *Concept Enrichment* refers to the process of learning novel, model-native concepts from AI systems (Ayonrinde, 2025). A leading example is the work of Schut et al. (2023), who extracted complex, novel chess strategies from a strong model and filtered for human-learnability by testing which strategies could be taught to a weaker model. They then successfully taught these strategies to chess grandmasters. Analogously, we extract *complex SAE features* whose activation patterns are not apparent from examples only, and then assess an LLM agent’s ability to learn them as a proxy for human learnability.

### 3 METHODS

A feature is *complex* if it yields near-chance detection accuracy using one-shot AutoInterp,<sup>1</sup> suggesting it encodes patterns too intricate for a single explanation to capture.<sup>2</sup> To identify and interpret such features, we use MSA agents to filter for features with low detection accuracy and then iteratively refine feature explanations through targeted experimentation and feedback from activations.

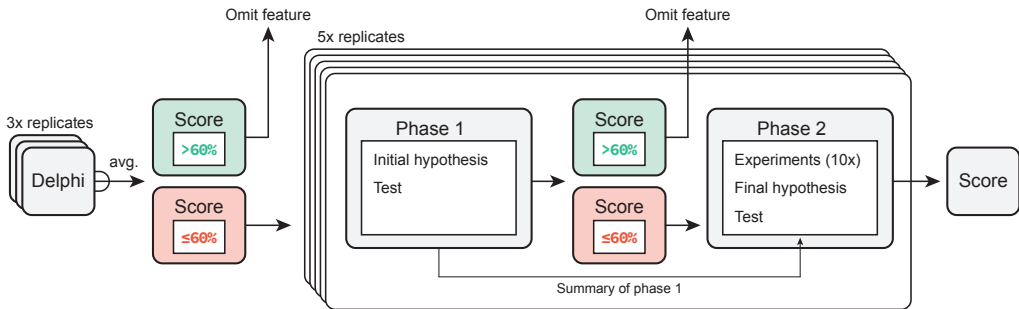


Figure 2: **The MSA agent workflow.** First, we select features with a low Delphi detection accuracy and pass them to the initial phase of the agent study, where an LLM agent is provided with example sequences about the feature and generates an initial explanation. The agent is then scored on a held-out test set based on this explanation. We then pass those features whose detection accuracies remain near-chance to the refinement phase, where the agent refines its explanation by generating its own probe sequences and observing their activations. Finally, we score the agent on a fresh test set based on its refined explanation.

For each SAE feature, we apply the following sequence: Delphi pre-filter  $\rightarrow$  initial phase explanation generation and evaluation  $\rightarrow$  initial phase filter  $\rightarrow$  Refinement phase refinement and re-evaluation. We report the final learnability score  $\Delta Acc$  for features that reach the refinement phase (that is *complex features*).

**Delphi Pre-filter.** As a low-cost initial filter for complex features, we use “Delphi”, an automated interpretability method by Paulo et al. (2025a), with a relatively weak LLM explainer. We select features with an average detection accuracy below an upper threshold and pass them to the initial phase.

**Initial Phase: Initial Explanation Generation and Evaluation.** The agent receives activating and non-activating example sequences together with the feature’s activation values, and generates an

<sup>1</sup>However, difficulty of explanation does not necessarily imply intrinsic complexity, as even simple concepts can be hard to explain.

<sup>2</sup>Following Bricken et al. (2023); Paulo et al. (2025a); Cunningham et al. (2023), we use an LLM’s ability to explain a feature as an (imperfect) proxy for how human-understandable a feature is.

162 initial explanation  $H_1$ . The agent is then tested on held-out sequences: given a set of new sequences  
 163 (both activating and non-activating), the agent predicts which sequences activate the feature, based  
 164 on  $H_1$ . We denote the resulting detection accuracy (computed per agent replicate) as  $Acc_1$ . This  
 165 is similar to Delphi, but uses a more capable model, different prompting strategies, and provides  
 166 the agent with example sequences and their activation values during testing, not just the explanation  
 167  $H_1$ .<sup>3</sup>

168  
 169 **Initial Phase Filter.** We retain only features whose initial phase detection accuracy remains near  
 170 chance (i.e.  $\sim 50\%$  for balanced test sets) for each agent replicate, confirming that the feature resists  
 171 straightforward explanation. These remaining features are deemed *complex* and proceed to the final  
 172 phase.

173  
 174 **Refinement Phase: Probe-Guided Hypothesis Refinement and Re-evaluation.** In a new con-  
 175 text window, the agent is provided with (1) the example sequences from the initial phase, (2) the  
 176 test sequences from initial phase with their feature activation values revealed, and (3) the initial hy-  
 177 pothesis  $H_1$ . The agent then repeatedly generates a probe sequence for which it is provided with the  
 178 feature activation values. For each probe sequence, the agent uses the activation values to update  
 179 its explanation of the feature accordingly. After generating a fixed number of probe sequences, the  
 180 agent produces a final refined hypothesis  $H_2$  and is evaluated on a fresh test set. We denote the re-  
 181 sulting detection accuracy (computed per agent replicate) as  $Acc_2$ . Full prompts and implementation  
 182 details are provided in appendix D.

183  
 184 **Learnability Metric.** We quantify learnability  $\Delta Acc$  by the improvement in average detection  
 185 between the refinement phase and the initial phase,  $\Delta Acc = Acc_2 - Acc_1$ .

## 187 4 EXPERIMENTAL SETUP

188  
 189 **Models.** We use Gemma-2-2B (Riviere et al., 2024) as our base model to be explained, leveraging  
 190 the Gemma Scope SAEs (Lieberum et al., 2024) through SAELens (Bloom et al., 2024). We use  
 191 SAE with  $K = 16,384$  features trained on layer 13 residual stream activations (post-MLP) with an  
 192 average sparsity of 84.<sup>4</sup>

193  
 194 **Dataset.** All sequences are sampled from the SmoLLM2 corpus (Allal et al., 2025), from which  
 195 we sample a total of 10M tokens.

196  
 197 **Complex Feature Selection.** We select the first 500 features by feature ID that have an average  
 198 Delphi detection accuracy below 0.60 for the initial phase of the MSA agent, using Qwen3-4B-  
 199 Instruct-2507 (Yang et al., 2025) as the Delphi explainer model. As SAE features are unordered,  
 200 this yields a diverse set of features that activate on a variety of different patterns.

201 We select those features for which no agent obtained a initial phase detection accuracy above 0.60  
 202 for the refinement phase of the MSA agent.

203  
 204 **MSA Agent.** We use Claude 4.5 Haiku (Anthropic, 2025) as the LLM agent.<sup>5</sup> We provide agents  
 205 with 10 activating and 10 non-activating sequences for the initial explanation generation and for  
 206 the tests in both the initial phase and the refinement phase. During the refinement phase, the agent  
 207 generates 10 exploratory sequences. For each feature, we sample non-activating sequences ran-  
 208 domly, while activating sequences are drawn from the top-100 sequences by activation, sampling 10  
 209 per stage without replacement. We average agent detection accuracy scores over 5 different agents  
 210 with different random seeds, ensuring each agent receives different activating and non-activating  
 211 sequences.

212  
 213 <sup>3</sup>A detailed comparison between the initial phase and Delphi is provided in Appendix C.

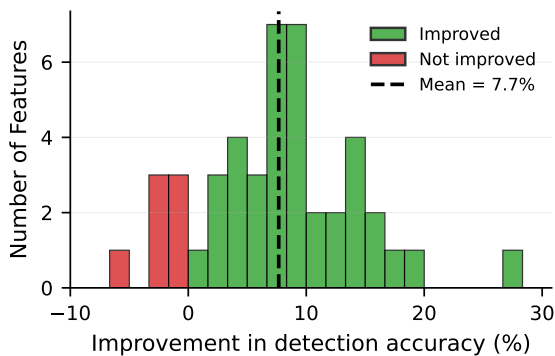
214 <sup>4</sup>[https://huggingface.co/google/gemma-scope-2b-pt-res/tree/main/layer\\_13/width\\_16k/average\\_10\\_84](https://huggingface.co/google/gemma-scope-2b-pt-res/tree/main/layer_13/width_16k/average_10_84)

215 <sup>5</sup>See Appendix B for a comparison with GPT-4o-mini.

## 216 5 RESULTS

### 217 5.1 DETECTION ACCURACY IMPROVEMENT

218 Figure 3 shows the distribution of improvements in detection accuracy from the initial phase test  
 219 to the refinement phase test, averaged across agents, for the 45 features which went through the  
 220 refinement phase. The mean detection accuracy over all features increased from 51% to 59%. In  
 221 particular, the effective average improvement, i.e. the improvement beyond random guessing (50%),  
 222 was  $7\% \pm 1\%$  (SE), corresponding to higher accuracy in detecting the presence of feature activations  
 223 in sequences after iterative refinement. The distribution is skewed towards positive improvements,  
 224 with 38 features showing an increase in detection accuracy, 6 features showing a decrease, and 1  
 225 feature showing no change. A sign test<sup>6</sup> applied to the agent-averaged mean detection accuracies  
 226 shows that the predominance of improved over degraded features is statistically significant ( $p =$   
 227  $10^{-6}$ ). The top 10 learnable features ranked by effective improvement above chance are shown in  
 228 Table 1 in the appendix.  
 229



230  
231  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244 **Figure 3: Iterative refinement of explanations improves detection accuracy across features.**  
 245 The histogram shows the change in detection accuracies between the initial phase test based on  
 246 the initial explanation and the refinement phase test for the 45 complex features considered for the  
 247 refinement phase, demonstrating that most features saw an improvement. The dashed line shows the  
 248 average improvement of  $7.7\% \pm 1.0\%$  (SE).  
 249

### 250 5.2 EXAMPLE OF COMPLEX FEATURE LEARNING

251 The aggregate results show that most complex features improve under iterative refinement, and here  
 252 we zoom in for an illustrative phenomenological example of a complex feature. Feature 30 was  
 253 initially uninterpretable by the authors but learnable by the agents. This feature had a mean initial  
 254 detection accuracy of 46% across agents, which improved to 63% after iterative refinement. The  
 255 maximum initial score across agents was 60%, and the maximum final score was 80%, indicating  
 256 that hypothesis refinement was necessary to learn the feature’s meaning.  
 257

258 Based on its top activating sequences (a subset of which are shown in Figure 4), feature 30 appears  
 259 to activate on prepositions such as “on” and “of”. However, the agent’s initial explanation achieved  
 260 only 60% accuracy, suggesting a deeper pattern. We encourage the reader to form an interpretation  
 261 before reading the agent’s final explanation in Figure 5.  
 262

263 Figure 5 shows excerpts from the reasoning process of the agent that achieved the highest final detec-  
 264 tion accuracy. The agent initially hypothesizes that the feature activates on prepositions, achieving  
 265 60% detection accuracy. After receiving feedback, it recognizes that it missed structural repetition  
 266 patterns. The agent then experiments with sequences containing repeated prepositional phrases, yet  
 267 observes no activations. It notes that the original activating examples contained more surrounding

268 <sup>6</sup>A sign test is a nonparametric test that assesses whether the median of paired differences differs from zero,  
 269 using only the direction of the differences and making no distributional assumptions. For more details, see e.g.  
 (Conover, 1999).

270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323

Feature #30 activations
I am sorry that you cannot rest in peace.\n\nMore than anything, I am sorry that you didn't have a chance
was why the government wanted to adapt its rules. "The bigger a financial centre is, the greater the risk of its being
1. Do you agree more with Mill or with Kant's ethics on what actions are morally right
Is it because the troops can carry more ammunition? Is it because the emphasis on American military training

Figure 4: **Example activations for feature 30.** Four out of the top 100 activating sequences for feature 30. The feature activates on the highlighted tokens.

context, so it generates longer sequences and observes activations. Finally, the agent synthesizes these observations into a refined explanation: the feature activates on prepositions and conjunctions appearing within grammatically or conceptually parallel structures. This refined explanation achieves 80% detection accuracy.

Returning to Figure 4, we can verify that the highlighted tokens do indeed appear within parallel structures. The first sequence contains the repeated phrase “I am sorry that you...”; the second contains the parallel comparatives “the bigger” and “the greater”; the third contains the parallel “with Mill” and “with Kant”; and the fourth contains the repeated “Is it because the”. These examples validate the agent’s refined explanation.

We present additional complex yet learnable features in Appendix A. Some features appear superficially simple but reveal hidden complexity: for instance, feature 368 activates on the first comma in a periodic sentence, but only when preceded by an additional sentence with relevant topical context. Other features combine tokenization and structure: feature 505 activates on the second mention of an uncommon token within a sequence. Finally, for some features, the agent’s early hypotheses prove useful even when later refinements become less accurate. This was the case for feature 490, which exhibits parallel patterns similar to feature 30, but the agent’s hypotheses degraded after the first few experiments. Even in such cases, the agent’s initial reasoning can provide useful clues to help a human interpret the feature.

## 6 DISCUSSION

Our results indicate that the Multi-Shot AutoInterp (MSA) agent allows LLM agents to produce more faithful explanations of concepts underlying SAE features. This is reflected in an average increase in detection accuracy of 7.7 percentage points compared to explanations produced after a single exposure to activating sequences, with 84% of features showing some improvement and 34% showing an improvement of more than 10 percentage points.

One exciting implication of our agents’ performance is the potential to improve understanding of features not only for agents, but also for humans. We see this happening in two ways: firstly, there is some evidence that LLMs and humans perform cognitive tasks in similar ways (Binz & Schulz, 2023),<sup>7</sup> which might include explanation generation, refinement, and evaluation. If both humans and AI understand similar explanations then (1) the features that had increased detection accuracy are likely also understandable by humans, and (2) the explanations our agents found have the potential to accelerate human understanding of these features.<sup>8</sup> Secondly, using human learners in the place of our MSA agents could provide an effective way for humans to learn complex features. Together, these two mechanisms provide ways for humans to learn novel concepts from the internal representations of LLMs and we would be excited about future work exploring this.

<sup>7</sup>Though see also Goodfellow et al. (2015) and Moravec’s paradox (Moravec, 1988) for ways in which the human and AI skill profiles can be quite different.

<sup>8</sup>Similarly, features with detection accuracies that did not improve may be more difficult for humans to understand. These features may be inherently difficult to explain in natural language or may require learning strategies beyond our methods.

324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377

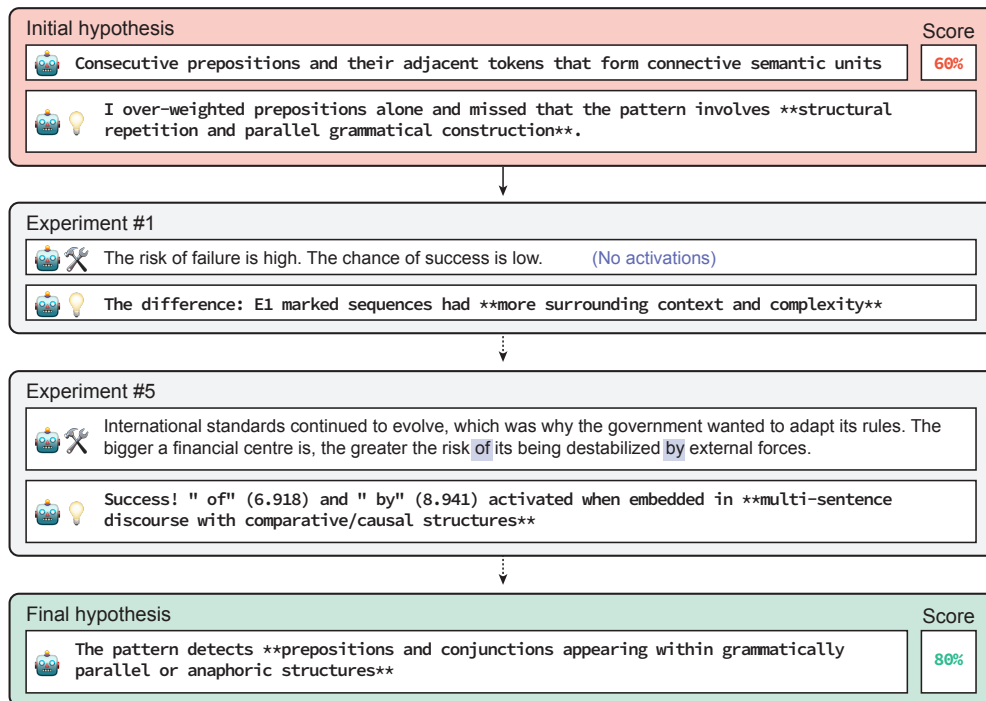


Figure 5: **Agent outputs while learning feature 30.** The agent iteratively refines its hypothesis through experimentation, improving detection accuracy from 60% to 80%. “E1” refers to the initial examples provided to the agent.

**Limitations.** Our learnability results use LLM agents as proxies for human learners, but agent learnability need not imply human learnability or vice versa. Another limitation is that negative examples are sampled randomly from non-activating sequences instead of using hard negative mining (Sohn, 2016; Liu et al., 2016). Random sampling can hence result in overly easy negatives, and coarse explanations may achieve high detection accuracy without capturing the feature’s true boundary (e.g., a “mammal” feature appearing as an “animal” feature). Moreover, positive examples are drawn from high-activation contexts, which do not reflect the full activation distribution. As a result, improvements in detection accuracy may overstate how well refined explanations capture the feature’s full behavior. Our claim that the MSA agent is a way to learn concepts in LLMs is also conditional on SAE features being a “real” representation of concepts in an LLM: SAE features can change with SAE training choices (e.g., sparsity and dictionary size), as feature projections are lossy approximations of the residual stream.

**Future work.** In addition to filtering for complex features, importance-based feature selection could help prioritize features that have the greatest impact on the base model’s behaviour.<sup>9</sup> Additionally, allowing an agent to refine its explanations over additional iterations could lead to further refined explanations about feature behaviour, in particular for features that did not improve in our current setup. Finally, to assess whether the learnability candidates produced by the LLM agents are truly human-learnable, future work should carry out studies where humans are presented with approaches similar to the MSA agent we used for the LLM agents.

<sup>9</sup>We attempted to implement importance filtering using direct logit attribution, and discuss the results in Appendix E.

## 7 RELATED WORK

### 7.1 AGENTIC INTERPRETABILITY

Standard automated interpretability is one-shot (Paulo et al., 2025a): an LLM is provided with examples, produces an explanation, and is evaluated based on the explanation. This works well for many features but not for those whose activation patterns are not apparent from a small sample. One solution is to let agents iteratively explore feature behavior, an approach that has been pursued by recent work at varying levels of automation.

**Interactive and human-in-the-loop systems.** One approach keeps the human in the loop, using agents to accelerate rather than replace the researcher. *Scribe* (Bissell et al., 2025) is an interpretability research agent that can autonomously find interesting SAE features, analyze activations, and run steering experiments, but the research direction remains human-driven. *Seer* (Ajobi, 2025) provides sandboxed environments in which agents execute interpretability experiments iteratively on behalf of a researcher who defines the investigation. Kim et al. (2025) propose a multi-turn conversational approach in which an LLM proactively assists human understanding by building a model of the user’s knowledge, requiring continuous human engagement. These systems reduce manual effort but still depend on a human to set goals and judge results.

**Fully automated agentic interpretability.** Fully automated systems remove the human from the loop entirely. MAIA (Shaham et al., 2024) equips LLM agents with tools for generating and testing hypotheses about neuron behavior in vision models. SAGE (Han et al., 2025) iteratively refines SAE feature explanations, improving average explanation quality.

Rather than improving explanations *on average*, our work evaluates the question of whether agentic refinement helps agents learn *complex* features specifically, those that resist one-shot explanation, and whether the resulting explanations help humans understand novel concepts.

### 7.2 SAE-BASED SCIENTIFIC DISCOVERY

AI systems trained on scientific data may learn representations that capture scientific concepts unknown to human researchers. Extracting such concepts could lead to new scientific insight, but it requires methods that surface interpretable, novel features. In protein biology, Simon & Zou (2025) and Adams et al. (2025) show that applying SAEs to protein language models recovers features corresponding to known functional motifs, demonstrating that SAE features encode meaningful structure. In genomics, Brix et al. (2025) and Wang et al. (2026) use SAEs to identify biologically relevant features in DNA and genomic foundation models. A key open question in this line of work is whether the novel (previously unknown to humans) concepts these methods uncover are human-understandable. While we have not applied the MSA agent to assess human learnability of novel features, it provides one possible pathway for evaluating such feature learnability.

## 8 CONCLUSION

We introduce *Multi-shot AutoInterp* (MSA) agents to identify *complex* SAE features with low initial detection accuracy, and we have shown that a substantial subset of these features is nevertheless *learnable*. We found an increase in detection accuracy by 7.7% across 45 complex features when LLM agents iteratively refine their explanations.

Manual inspection indicates that the refined explanations better align with true activation patterns and often can contribute to improved human understanding of feature behavior. When considering LLM agents as proxies for human learners, our results suggest that an approach similar to the MSA agent could enable humans themselves to learn features that are initially opaque.

Our work takes a step toward bridging the knowledge gap between humans and AI models, suggesting that with appropriate methods, humans may be able to extract and learn novel concepts from the internal representations of LLMs.

## REFERENCES

- 432  
433  
434 Etowah Adams, Liam Bai, Minji Lee, Yiyang Yu, and Mohammed AlQuraishi. From mechanistic  
435 interpretability to mechanistic biology: Training, evaluating, and interpreting sparse autoen-  
436 coders on protein language models, 2025. URL <https://openreview.net/forum?id=zdOGBRQEbz>.  
437
- 438 Ajobi. Seer: A framework for agentic interpretability experiments. <https://github.com/ajobi-uhc/seer>, 2025.  
439
- 440 Loubna Ben Allal, Anton Lozhkov, Elie Bakouch, Gabriel Martín Blázquez, Guilherme Penedo,  
441 Lewis Tunstall, Andrés Marafioti, Hynek Kydlíček, Agustín Piqueres Lajarín, Vaibhav Srivastav,  
442 Joshua Lochner, Caleb Fahlgren, Xuan-Son Nguyen, Clémentine Fourier, Ben Burtenshaw, Hugo  
443 Larcher, Haojun Zhao, Cyril Zakka, Mathieu Morlon, Colin Raffel, Leandro von Werra, and  
444 Thomas Wolf. Smollm2: When smol goes big – data-centric training of a small language model,  
445 2025. URL <https://arxiv.org/abs/2502.02737>.  
446
- 447 Anthropic. Claude haiku 4.5 model card. <https://assets.anthropic.com/m/99128ddd009bdcb/Claude-Haiku-4-5-System-Card.pdf>, 2025. Accessed 2025-  
448 12-14.  
449
- 450 Kola Ayonrinde. Position: Interpretability is a bidirectional communication problem. In *ICLR 2025*  
451 *Workshop on Bidirectional Human-AI Alignment*, 2025. URL <https://openreview.net/forum?id=04LaRH4zSI>.  
452
- 453 Kola Ayonrinde and Louis Jaburi. A mathematical philosophy of explanations in mechanistic inter-  
454 pretability – the strange science part i.i, 2025. URL <https://arxiv.org/abs/2505.00808>.  
455  
456
- 457 Leonard Bereska and Efstratios Gavves. Mechanistic interpretability for ai safety – a review, 2024.  
458 URL <https://arxiv.org/abs/2404.14082>.  
459
- 460 Steven Bills, Nick Cammarata, Dan Mossing, Henk Tillman, Leo Gao, Gabriel Goh, Ilya  
461 Sutskever, Jan Leike, Jeff Wu, and William Saunders. Language models can explain  
462 neurons in language models. <https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html>, 2023.  
463
- 464 Marcel Binz and Eric Schulz. Using cognitive psychology to understand gpt-3. *Proceedings of the*  
465 *National Academy of Sciences*, 120(6), February 2023. ISSN 1091-6490. doi: 10.1073/pnas.  
466 2218523120. URL <http://dx.doi.org/10.1073/pnas.2218523120>.
- 467 Mark Bissell, Michael Byun, and Daniel Balsam. You and your research agent: Lessons  
468 from using agents for interpretability research. <https://www.goodfire.ai/blog/you-and-your-research-agent>, 2025. Goodfire blog post.  
469
- 470 Joseph Bloom, Curt Tigges, Anthony Duong, and David Chanin. Saelens. <https://github.com/decoderesearch/SAELens>, 2024.  
471  
472
- 473 Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly,  
474 Nick Turner, Cem Anil, Carson Denison, Amanda Askell, Robert Lasenby, Yifan Wu,  
475 Shauna Kravec, Nicholas Schiefer, Tim Maxwell, Nicholas Joseph, Zac Hatfield-Dodds, Alex  
476 Tamkin, Karina Nguyen, Brayden McLean, Josiah E Burke, Tristan Hume, Shan Carter,  
477 Tom Henighan, and Christopher Olah. Towards monosemanticity: Decomposing language  
478 models with dictionary learning. *Transformer Circuits Thread*, 2023. <https://transformer-circuits.pub/2023/monosemantic-features/index.html>.  
479
- 480 Garyk Brixi, Matthew G. Durrant, Jerome Ku, Michael Poli, Greg Brockman, Daniel Chang,  
481 Gabriel A. Gonzalez, Samuel H. King, David B. Li, Aditi T. Merchant, Mohsen Naghipourfar,  
482 Eric Nguyen, Chiara Ricci-Tam, David W. Romero, Gwanggyu Sun, Ali Taghibakshi, Anton  
483 Vorontsov, Brandon Yang, Myra Deng, Liv Gorton, Nam Nguyen, Nicholas K. Wang, Etowah  
484 Adams, Stephen A. Baccus, Steven Dillmann, Stefano Ermon, Daniel Guo, Rajesh Ilango, Ken  
485 Janik, Amy X. Lu, Reshma Mehta, Mohammad R.K. Mofrad, Madelena Y. Ng, Jaspreet Pannu,  
Christopher Ré, Jonathan C. Schmok, John St. John, Jeremy Sullivan, Kevin Zhu, Greg Zynda,

- 486 Daniel Balsam, Patrick Collison, Anthony B. Costa, Tina Hernandez-Boussard, Eric Ho, Ming-  
487 Yu Liu, Thomas McGrath, Kimberly Powell, Dave P. Burke, Hani Goodarzi, Patrick D. Hsu, and  
488 Brian L. Hie. Genome modeling and design across all domains of life with Evo 2, 2025. URL  
489 <https://www.biorxiv.org/content/10.1101/2025.02.18.638918v1>.
- 490 W. J. Conover. *Practical Nonparametric Statistics*. John Wiley & Sons, New York, 3 edition, 1999.
- 491
- 492 Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. Sparse autoen-  
493 coders find highly interpretable features in language models, 2023. URL <https://arxiv.org/abs/2309.08600>.
- 494
- 495 Joshua Engels, Eric J. Michaud, Isaac Liao, Wes Gurnee, and Max Tegmark. Not all language  
496 model features are one-dimensionally linear, 2025. URL <https://arxiv.org/abs/2405.14860>.
- 497
- 498
- 499 Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial  
500 examples, 2015. URL <https://arxiv.org/abs/1412.6572>.
- 501
- 502 Wes Gurnee, Emmanuel Ameisen, Isaac Kauvar, Julius Tarnig, Adam Pearce, Chris Olah, and  
503 Joshua Batson. When models manipulate manifolds: The geometry of a counting task. *Trans-*  
504 *former Circuits Thread*, 2025. URL <https://transformer-circuits.pub/2025/linebreaks/index.html>.
- 505
- 506 Jiaojiao Han, Wujiang Xu, Mingyu Jin, and Mengnan Du. Sage: An agentic explainer framework  
507 for interpreting sae features in language models, 2025. URL <https://arxiv.org/abs/2511.20820>.
- 508
- 509 Been Kim, John Hewitt, Neel Nanda, Noah Fiedel, and Oyvind Tafjord. Because we have LLMs,  
510 we can and should pursue agentic interpretability, 2025. URL <https://arxiv.org/abs/2506.12152>.
- 511
- 512
- 513 Tom Lieberum, Senthoran Rajamanoharan, Arthur Conmy, Lewis Smith, Nicolas Sonnerat, Vikrant  
514 Varma, János Kramár, Anca Dragan, Rohin Shah, and Neel Nanda. Gemma scope: Open sparse  
515 autoencoders everywhere all at once on gemma 2, 2024. URL <https://arxiv.org/abs/2408.05147>.
- 516
- 517 Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and  
518 Alexander C. Berg. *SSD: Single Shot MultiBox Detector*, pp. 21–37. Springer International  
519 Publishing, 2016. ISBN 9783319464480. doi: 10.1007/978-3-319-46448-0\_2. URL [http://dx.doi.org/10.1007/978-3-319-46448-0\\_2](http://dx.doi.org/10.1007/978-3-319-46448-0_2).
- 520
- 521 Hans Moravec. *Mind Children: The Future of Robot and Human Intelligence*. Harvard University  
522 Press, Cambridge, USA, 1988.
- 523
- 524 Gonalo Paulo, Alex Mallen, Caden Juang, and Nora Belrose. Automatically interpreting millions of  
525 features in large language models, 2025a. URL <https://arxiv.org/abs/2410.13928>.
- 526
- 527 Gonalo Paulo, Stepan Shabalín, and Nora Belrose. Transcoders beat sparse autoencoders for inter-  
528 pretability, 2025b. URL <https://arxiv.org/abs/2501.18823>.
- 529
- 530 Senthoran Rajamanoharan, Arthur Conmy, Lewis Smith, Tom Lieberum, Vikrant Varma, János  
531 Kramár, Rohin Shah, and Neel Nanda. Improving dictionary learning with gated sparse au-  
532 toencoders. In *Advances in Neural Information Processing Systems*, 2024. URL <https://arxiv.org/abs/2404.16014>.
- 533
- 534 Morgane Riviere, Shreya Pathak, Pier Giuseppe Sessa, Cassidy Hardin, Surya Bhupatiraju, Léonard  
535 Hussenot, Thomas Mesnard, Bobak Shahriari, Alexandre Ramé, Johan Ferret, Peter Liu, Pouya  
536 Tafti, Abe Friesen, Michelle Casbon, Sabela Ramos, Ravin Kumar, Charline Le Lan, Sammy  
537 Jerome, Anton Tsitsulin, Nino Vieillard, Piotr Stanczyk, Sertan Girgin, Nikola Momchev, Matt  
538 Hoffman, Shantanu Thakoor, Jean-Bastien Grill, Behnam Neyshabur, Olivier Bachem, Alanna  
539 Walton, Aliaksei Severyn, Alicia Parrish, Aliya Ahmad, Allen Hutchison, Olivier Abdagic,  
Amanda Carl, Amy Shen, Andy Brock, Andy Coenen, Anthony Laforge, Antonia Paterson, Ben  
Bastian, Bilal Piot, Bo Wu, Brandon Royal, Charlie Chen, Chintu Kumar, Chris Perry, Chris

- 540 Welty, Christopher A. Choquette-Choo, Danila Sinopalnikov, David Weinberger, Dimple Vi-  
541 jaykumar, Dominika Rogozińska, Dustin Herbison, Elisa Bandy, Emma Wang, Eric Noland, Er-  
542 ica Moreira, Evan Senter, Evgenii Eltyshev, Francesco Visin, Gabriel Rasskin, Gary Wei, Glenn  
543 Cameron, Gus Martins, Hadi Hashemi, Hanna Klimeczak-Plucińska, Harleen Batra, Harsh Dhand,  
544 Ivan Nardini, Jacinda Mein, Jack Zhou, James Svensson, Jeff Stanway, Jetha Chan, Jin Peng  
545 Zhou, Joana Carrasqueira, Joana Iljazi, Jocelyn Becker, Joe Fernandez, Joost van Amersfoort,  
546 Josh Gordon, Josh Lipschultz, Josh Newlan, Ju yeong Ji, Kareem Mohamed, Kartikeya Badola,  
547 Kat Black, Katie Millican, Keelin McDonell, Kelvin Nguyen, Kiranbir Sodhia, Kish Greene,  
548 Lars Lowe Sjoesund, Lauren Usui, Laurent Sifre, Lena Heuermann, Leticia Lago, Lilly Mc-  
549 Nealus, Livio Baldini Soares, Logan Kilpatrick, Lucas Dixon, Luciano Martins, Machel Reid,  
550 Manvinder Singh, Mark Iverson, Martin Görner, Mat Velloso, Mateo Wirth, Matt Davidow,  
551 Matt Miller, Matthew Rahtz, Matthew Watson, Meg Risdal, Mehran Kazemi, Michael Moyni-  
552 han, Ming Zhang, Minsuk Kahng, Minwoo Park, Mofi Rahman, Mohit Khatwani, Natalie Dao,  
553 Nenshad Bardoliwalla, Nesh Devanathan, Neta Dumai, Nilay Chauhan, Oscar Wahltinez, Pankil  
554 Botarda, Parker Barnes, Paul Barham, Paul Michel, Pengchong Jin, Petko Georgiev, Phil Cullit-  
555 ton, Pradeep Kuppala, Ramona Comanescu, Ramona Merhej, Reena Jana, Reza Ardeshir Rokni,  
556 Rishabh Agarwal, Ryan Mullins, Samaneh Saadat, Sara Mc Carthy, Sarah Cogan, Sarah Perrin,  
557 Sébastien M. R. Arnold, Sebastian Krause, Shengyang Dai, Shruti Garg, Shruti Sheth, Sue Ron-  
558 strom, Susan Chan, Timothy Jordan, Ting Yu, Tom Eccles, Tom Hennigan, Tomas Kocisky, Tulsee  
559 Doshi, Vihan Jain, Vikas Yadav, Vilobh Meshram, Vishal Dharmadhikari, Warren Barkley, Wei  
560 Wei, Wenming Ye, Woohyun Han, Woosuk Kwon, Xiang Xu, Zhe Shen, Zhitao Gong, Zichuan  
561 Wei, Victor Cotruta, Phoebe Kirk, Anand Rao, Minh Giang, Ludovic Peran, Tris Warkentin, Eli  
562 Collins, Joelle Barral, Zoubin Ghahramani, Raia Hadsell, D. Sculley, Jeanine Banks, Anca Dra-  
563 gan, Slav Petrov, Oriol Vinyals, Jeff Dean, Demis Hassabis, Koray Kavukcuoglu, Clement Fara-  
564 bet, Elena Buchatskaya, Sebastian Borgeaud, Noah Fiedel, Armand Joulin, Kathleen Kenealy,  
565 Robert Dadashi, and Alek Andreev. Gemma 2: Improving open language models at a practical  
566 size, 2024. URL <https://arxiv.org/abs/2408.00118>.
- 566 Lisa Schut, Nenad Tomasev, Tom McGrath, Demis Hassabis, Ulrich Paquet, and Been Kim. Bridg-  
567 ing the human-ai knowledge gap: Concept discovery and transfer in alphazero, 2023. URL  
568 <https://arxiv.org/abs/2310.16410>.
- 569 Tamar Rott Shaham, Sarah Schwettmann, Franklin Wang, Achyuta Rajaram, Evan Hernandez, Jacob  
570 Andreas, and Antonio Torralba. MAIA: A multimodal automated interpretability agent, 2024.  
571 URL <https://arxiv.org/abs/2404.14394>.
- 572 David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez,  
573 Marc Lanctot, Laurent Sifre, Dharsan Kumaran, Thore Graepel, Timothy Lillicrap, Karen Si-  
574 monyan, and Demis Hassabis. Mastering chess and shogi by self-play with a general reinforc-  
575 e-ment learning algorithm, 2017. URL <https://arxiv.org/abs/1712.01815>.
- 576 Elana Simon and James Zou. Interplm: discovering interpretable features in protein language  
577 models via sparse autoencoders. *Nature Methods*, 22(10):2107–2117, Oct 2025. ISSN  
578 1548-7105. doi: 10.1038/s41592-025-02836-7. URL <https://doi.org/10.1038/s41592-025-02836-7>.
- 582 Kihyuk Sohn. Improved deep metric learning with multi-class n-pair loss objec-  
583 tive. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett (eds.), *Ad-  
584 vances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc.,  
585 2016. URL [https://proceedings.neurips.cc/paper\\_files/paper/2016/  
586 file/6b180037abbebea991d8b1232f8a8ca9-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2016/file/6b180037abbebea991d8b1232f8a8ca9-Paper.pdf).
- 587 Harrish Thasarathan, Julian Forsyth, Thomas Fel, Matthew Kowal, and Konstantinos Derpanis. Uni-  
588 versal sparse autoencoders: Interpretable cross-model concept alignment, 2025. URL <https://arxiv.org/abs/2502.03714>.
- 589 Kevin Wang, Alexandre Variengien, Arthur Conmy, Buck Shlegeris, and Jacob Steinhardt. Inter-  
590 pretability in the wild: a circuit for indirect object identification in gpt-2 small, 2022. URL  
591 <https://arxiv.org/abs/2211.00593>.

594 Nicholas Wang, Christoforos Nalmpantis, Ching Fang, Pouya Niki, Mark Bissell, Pooja Kathail,  
595 Dron Hazra, Andrey Karailiev, Michael Pearce, Javkhlán-Ochir Ganbat, Archa Jain, Luca Gi-  
596 acomoni, Daniel Balsam, Jonathan Wan, and Ravi Solanki. Using interpretability to iden-  
597 tify a novel class of alzheimer’s biomarkers, 2026. URL [https://www.goodfire.ai/  
598 research/interpretability-for-alzheimers-detection](https://www.goodfire.ai/research/interpretability-for-alzheimers-detection).

599 Jason Wei, Yi Tay, Rishi Bommasani, Colin Raffel, Barret Zoph, Sebastian Borgeaud, Dani Yo-  
600 gatama, Maarten Bosma, Denny Zhou, Donald Metzler, Ed H. Chi, Tatsunori Hashimoto, Oriol  
601 Vinyals, Percy Liang, Jeff Dean, and William Fedus. Emergent abilities of large language models,  
602 2022. URL <https://arxiv.org/abs/2206.07682>.

603 An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang  
604 Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu,  
605 Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin  
606 Yang, Jiayi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang,  
607 Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui  
608 Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang  
609 Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger  
610 Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan  
611 Qiu. Qwen3 technical report, 2025. URL <https://arxiv.org/abs/2505.09388>.

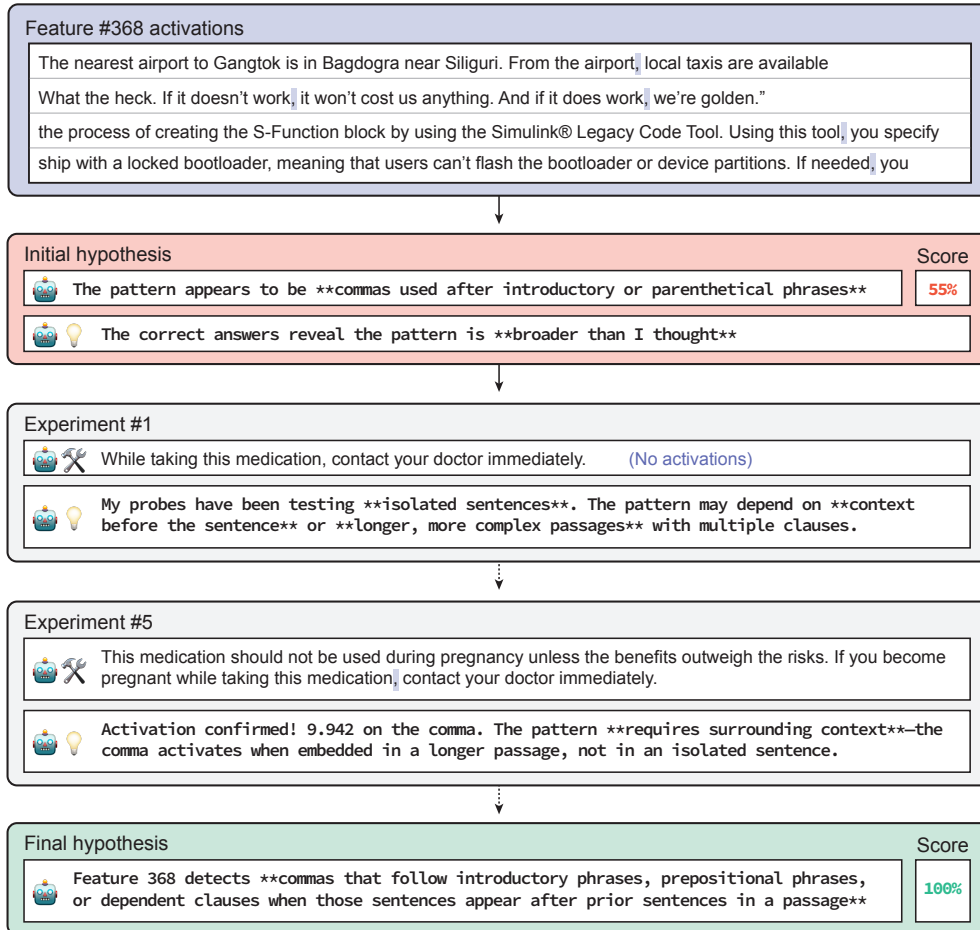
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647

648 DISCLOSURE ABOUT THE USAGE OF GENERATIVE ARTIFICIAL INTELLIGENCE  
649 TOOLS  
650

651 We used large language models for language editing and for assistance with code generation and  
652 debugging. All research ideas, experimental design, analysis, and conclusions are the authors' own.  
653

654 A RESULTS FOR ADDITIONAL FEATURES  
655

656 A.1 FEATURE 368  
657



690 Figure 6: **Example activations and agent output for feature 368.** Four out of the top 100  
691 activating sequences for feature 368 (top/blue). The feature activates on the highlighted tokens.  
692 The agent iteratively refines its hypothesis through experimentation, improving detection accuracy  
693 from 55% to 100%.

694 At first glance, the pattern in the top activations may seem simple: this feature appears to activate  
695 on commas in fronted dependent clauses (e.g. “From the airport, local taxis are available” rather  
696 than “Local taxis are available from the airport”). The agent notices this pattern in its initial expla-  
697 nation but achieves only a 55% detection accuracy. It tests a simple fronted clause but observes no  
698 activations, hypothesizing that the pattern may depend on preceding context. Once the agent adds  
699 preceding context to its previous probe sequence, it observes activations. Its final explanation is that  
700 this feature activates on commas following dependent clauses, but only if there is preceding context.  
701 Thus, while the activations shown may have suggested a simple one-shot explanation, the agent was  
able to identify a subtle hidden pattern via multi-shot experimentation.

## A.2 FEATURE 505

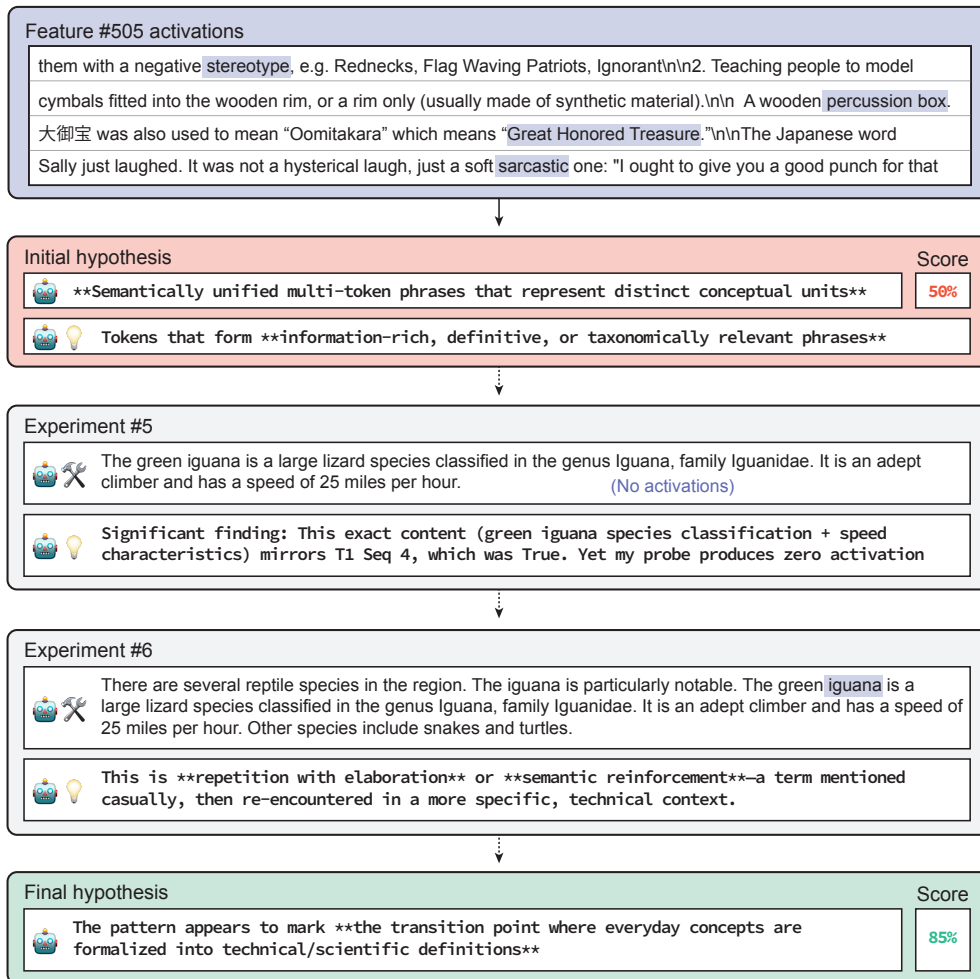


Figure 7: **Example activations and agent output for feature 505.** Four out of the top 100 activating sequences for feature 505 (top/blue). The feature activates on the highlighted tokens. The agent iteratively refines its hypothesis through experimentation, improving detection accuracy from 50% to 85%.

Due to the activations on a wide variety of nouns and adjectives surrounded by diverse contexts, this feature appeared unclear to the authors. The agent also seemed unsure of the pattern, as it broadly hypothesized that the feature encodes “semantically unified multi-token phrases”. However, it eventually tested a paraphrased positive sequence from phase 1 and surprisingly observed no activations. The agent then double-checked by probing with the original sequence and found that it indeed activated. It reasoned that the only difference between its probe and the original sequence was the exact structure—specifically “repetition with elaboration”. This explanation ultimately yielded 85% detection accuracy, suggesting that the agent learned the key pattern. Top activating sequences support this pattern. For instance, “negative stereotype” activates when followed by examples like “Rednecks” or “Ignorant”. On the other hand, adjectives like “sarcastic” activate when contrasted with “hysterical” within an explanation. Ultimately, by reading over the agent’s explanations, the authors were able to better understand the feature’s behavior.

## A.3 FEATURE 490

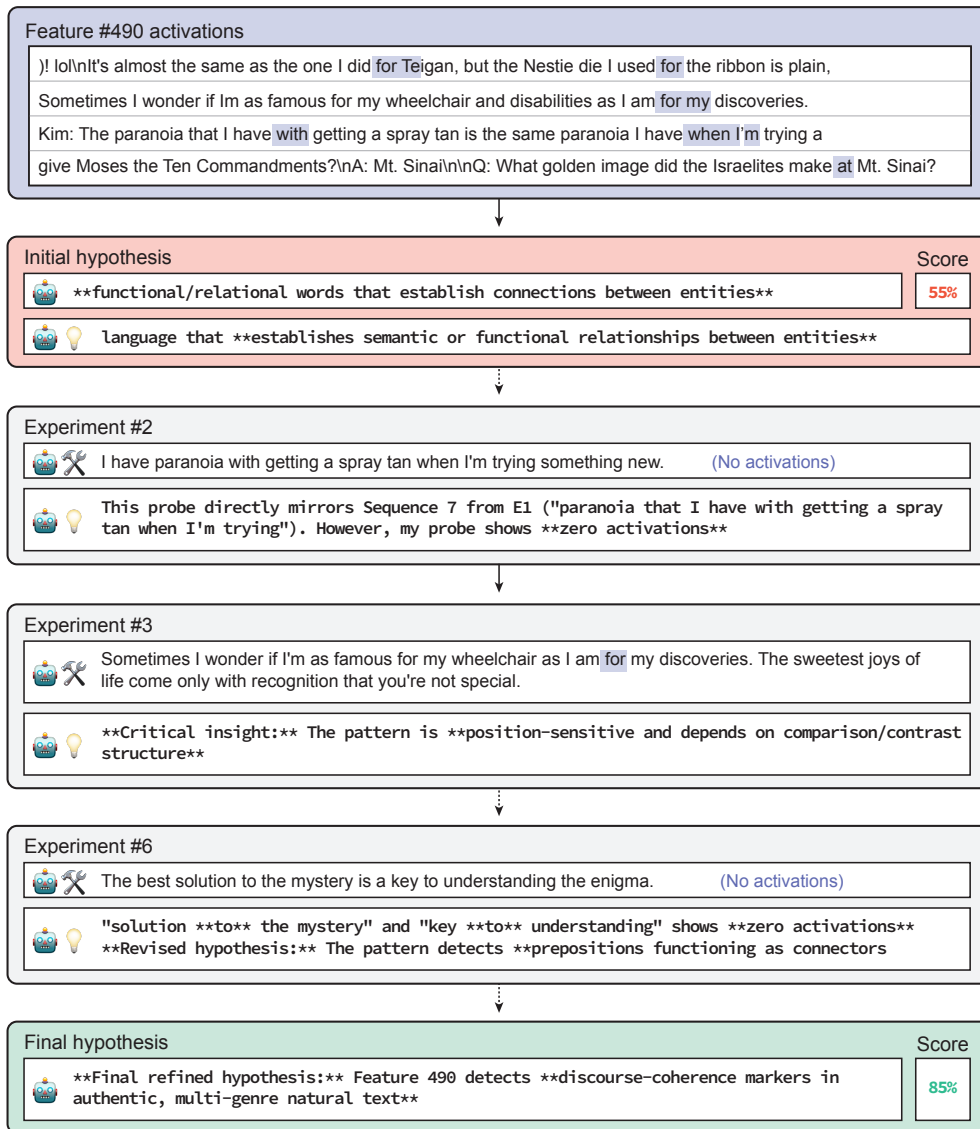


Figure 8: **Example activations and agent output for feature 490.** Four out of the top 100 activating sequences for feature 490 (top/blue). The feature activates on the highlighted tokens. The agent iteratively refines its hypothesis through experimentation, improving detection accuracy from 55% to 85%.

The agent begins by probing with a paraphrasing of a positive sequence from phase 1. However, it observes that it surprisingly does not activate. The agent then probes with a more exact copy of the original positive sequence and observes an activation. Comparing the two probes, the agent realizes that the main difference is the structure: activation “depends on comparison/contrast structure”. While this explanation explains top activating sequences, the agent continues to iterate and eventually generates a probe that does not activate as expected. The agent then quickly pivots explanations toward “prepositions functioning as connectors”, leading to a final explanation of “discourse-coherence markers”, which deviates from the original explanation. Nevertheless, with the context of all its tests in mind, the agent is still able to achieve 85% detection accuracy. Despite its tangent, the intermediate explanations helped the authors better understand this feature’s behavior.

Table 1: **Learnable features after refinement:** Top 10 features ranked by effective improvement above chance, defined as the increase in mean detection accuracy beyond random guessing.

Feature ID	Mean initial detection accuracy	Mean refined detection accuracy	Effective improvement
368	54%	82%	28%
505	53%	72%	19%
490	55%	71%	16%
1051	54%	69%	15%
465	52%	66%	14%
289	48%	64%	14%
1060	59%	73%	14%
30	46%	63%	13%
894	51%	63%	12%
332	49%	62%	12%

## B WEAKER LLM EXPLAINERS

To assess whether feature learnability under the MSA protocol depends on agent capability, we repeated our learnability experiments using a weaker LLM agent, GPT-4o mini. For both LLM agents used (Haiku 4.5 and GPT-4o mini), at least four out of five agents produced correctly formatted output for each feature in the initial phase. Figure 9 shows the distribution over agent-averaged initial detection accuracies for both LLM agents, showing that GPT 4o-mini produces a much lower average initial score (mean 64%) compared to Haiku 4.5 (mean 70%).

There are many more features in the lower score range (50-60%) for GPT-4o mini, indicating that there are more features that are challenging for this LLM agent. When running the MSA protocol on the 170 selected features, we find that the agents struggled with tool use, and there are only 97 features where each of the five agents produced correctly formatted output in the refinement phase. For those 97 features, we find that the average increase in detection accuracy is only  $1.9\% \pm 0.3\%$  (SE), which is significantly lower than the increase observed for Haiku 4.5 ( $7.7\% \pm 1.0\%$ ). There are only 2/97 features with an increase of more than 10% in detection accuracy, as opposed to 16/45 features for Haiku 4.5. However, a sign test still indicates that the predominance of 54 improved over 28 degraded features is statistically significant ( $p = 0.005$ ).

These results indicate that the choice of LLM agent has a strong effect on the learnability of features through the MSA protocol. This suggests a capability threshold: as explainer models improve, iterated automated interpretability and agentic refinement become increasingly practical research directions for features that previously appeared intractable. This mirrors how few-shot prompting became broadly useful only once models were good enough at in-context learning (Wei et al., 2022).

## C COMPARISON WITH DELPHI

Our implementation of the initial phase of MSA described in Section 3 is similar to Delphi Paulo et al. (2025b) in that they both generate explanations and then score them using detection accuracy. However, there are a few differences that make the initial phase and Delphi distinct:

- Delphi samples activating sequences across quantiles by default, whereas we sample the top activating sequences.
- Delphi uses an explainer model to generate a feature explanation, which is then handed off to a separate scorer model to predict activations. We use a single model to explain and predict within the same context window.

We suspect that these methodological differences cause detection accuracy in the initial phase to be higher on average than in Delphi.

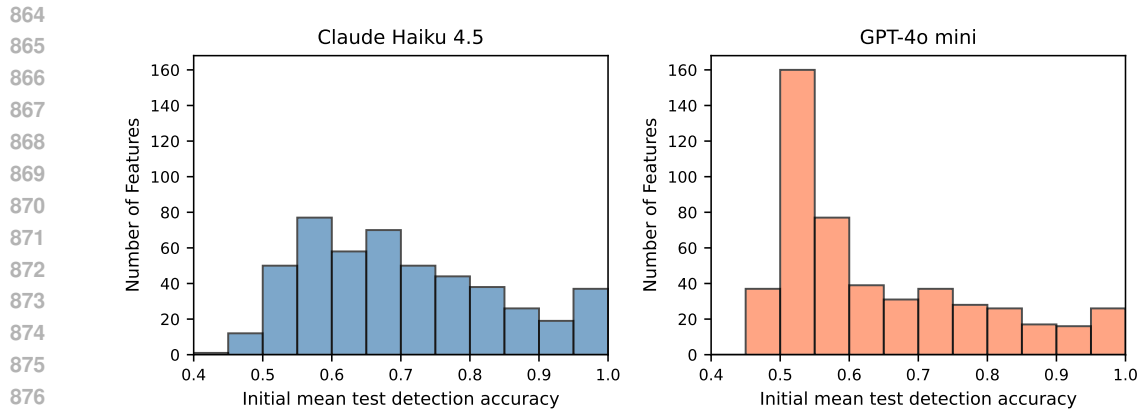


Figure 9: **Initial detection accuracy increases with LLM agent capability.** Distribution of initial detection accuracy scores for Claude Haiku 4.5 and GPT-4o mini. The distribution for Claude Haiku 4.5 is shifted to the right, indicating higher one-shot explanation performance relative to GPT-4o mini. GPT-4o mini therefore treats more features as complex, even when those features are readily explained by Haiku 4.5.

## D EXAMPLE AGENT TRANSCRIPT AND TOOL DEFINITIONS

### D.1 EXAMPLE AGENT TRANSCRIPT

We present the conversation flow of our agent study, showing system prompts, user messages, agent responses, and tool interactions in their original order. While individual messages are truncated for brevity, no steps in the interaction sequence are omitted; truncations are indicated with “...”. The prompts and example text sequences are illustrative, the concrete text excerpts relate to feature 30.

### Initial phase

#### System Prompt

You are a creative and analytical thinker focused on identifying patterns in sequences of text. You will go through phases of exploration and testing to understand a single hidden pattern.

In phase E1, you will be shown multiple different sequences, some containing the hidden pattern and some not. Those that contain the pattern will have markings like <<this>> surrounding important tokens relating to the hidden pattern. IMPORTANT: The <<angle brackets>> and activation values are annotations to help you|they are NOT the pattern itself. Your hypothesis should describe what the marked tokens have in common, not simply that they are marked. Consecutive important tokens appear together: <<just like this>>. Each sequence is formatted with a header ('==== Sequence i ===='), followed by the sequence itself. A divider ('-----') separates the sequence from numerical values corresponding to each marked token and how representative it is of the hidden pattern. If the sequence does not contain the pattern, this space will say 'None'. You will form a hypothesis about what the pattern is based on these marked sequences. Patterns may involve semantic, grammatical, positional, or other concepts. Pay more attention to high-importance tokens than low-importance tokens when forming hypotheses. Also pay attention to the tokens surrounding the marked tokens, as they too can influence whether the pattern is

918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971

present.

In phase T1, you will be given new sequences without markings to test your understanding of the pattern. You will use the provided 'submit\_predictions' tool to submit your True/False predictions for which of these sequences contain the pattern.

Be concise in your answers. Avoid listing examples, and prioritize conveying your hypotheses clearly and succinctly.

User

It is now Phase E1.  
Follow your instructions given the following sequences:  
===== Sequence 0: =====  
but let's divide it into chunks and start understanding it piece by piece.

Go to line 106 (github gist)

In our try

```
-----
Token activations:
' but': 0.000
' let': 0.000
''': 0.000
's': 0.000
' divide': 0.000
' it': 0.000
' into': 0.000
' chunks': 0.000
' and': 0.000
' start': 0.000
' understanding': 0.000
' it': 0.000
' piece': 0.000
' by': 0.000
' piece': 0.000
' .': 0.000
,
n
n': 0.000
'Go': 0.000
' to': 0.000
' line': 0.000
' ': 0.000
'1': 0.000
'0': 0.000
'6': 0.000
' (': 0.000
'github': 0.000
' gist': 0.000
')': 0.000
,
n
n': 0.000
'In': 0.000
' our': 0.000
' try': 0.000
(No activations above threshold)
...
```

972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025

### Agent

#### Phase E1 Analysis

Hypothesis: The hidden pattern is: **\*\*Preposition-based phrasal or multi-token sequences where specific prepositions appear together with certain surrounding tokens that form meaningful semantic units.\*\***

...

### User

It is now Phase T1.

Follow your instructions given the following sequences:

=====  
Sequence 0: =====, and why does it matter? Where does it come from? What does it look like in practice? How do people use it to find religious meaning? Why

...

### Tool Call

```
submit_predictions("0": false, "1": false, "2": true, "3":
false, ... )
```

### Tool Result

```
"0": true, "1": false, "2": true, "3": false,...
```

## Refinement phase

### System Prompt

You will now continue with additional phases to further refine your understanding of the pattern.

In phase E2, you will design and run experiments to probe the boundaries of your hypothesis. You will run a single experiment at a time. In each experiment, you will generate a single probe sequence and then submit a single tool call using the 'get\_activations' tool to see whether the pattern is detected in that sequence. After each experiment, reflect on the results and update your hypothesis accordingly, if at all.

Keep in mind all previous results from phases E1, T1, and E2 as you refine your hypothesis; it should be consistent with all known data.

In phase T2, you will again predict the presence of the pattern in new sequences based on your refined hypothesis. You will use the 'submit\_predictions' tool to submit your True/False predictions.

Be concise in your answers. Avoid listing examples, and prioritize conveying your hypotheses clearly and succinctly.

### User

=== RECAP FROM PREVIOUS PHASES ===

In Phase E1, you were shown these LABELED sequences (with <<markers>> indicating the pattern):

=====  
Sequence 0: ===== but let's divide it into chunks and start understanding it piece by piece.

```

1026
1027 Go to line 106 (github gist)
1028
1029 In our try
1030 -----
1031 Token activations:
1032 ' but': 0.000
1033 ' let': 0.000
1034 ''': 0.000
1035 's': 0.000
1036 ' divide': 0.000
1037 ' it': 0.000
1038 ' into': 0.000
1039 ' chunks': 0.000
1040 ' and': 0.000
1041 ' start': 0.000
1042 ' understanding': 0.000
1043 ' it': 0.000
1044 ' piece': 0.000
1045 ' by': 0.000
1046 ' piece': 0.000
1047 ' .': 0.000
1048 '
1049 n
1050 n': 0.000
1051 'Go': 0.000
1052 ' to': 0.000
1053 ' line': 0.000
1054 ' ': 0.000
1055 '1': 0.000
1056 '0': 0.000
1057 '6': 0.000
1058 ' (': 0.000
1059 'github': 0.000
1060 ' gist': 0.000
1061 ')': 0.000
1062 '
1063 n
1064 n': 0.000
1065 'In': 0.000
1066 ' our': 0.000
1067 ' try': 0.000
1068 (No activations above threshold)
1069 ...
1070 Your E1 hypothesis was:
1071 # Phase E1 Analysis
1072 ## Hypothesis
1073 The hidden pattern is: **Preposition-based phrasal or multi-token
1074 sequences where specific prepositions appear together with certain
1075 surrounding tokens that form meaningful semantic units.** ...
1076 In Phase T1, you were tested on these UNLABELED sequences:
1077 ===== Sequence 0: ===== , and why does it matter? Where
1078 does it come from? What does it look like in practice? How do
1079 people use it to find religious meaning? Why
1080 ...
1081 Your predictions and the correct answers were:
1082 Sequence 0: predicted False, actual True
1083 Sequence 1: predicted False, actual False
1084 Sequence 2: predicted True, actual True
1085 Sequence 3: predicted False, actual False
1086 ...
1087 Your T1 score was 60%.
1088
1089 === END RECAP ===

```

1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1130  
1131  
1132  
1133

Based on this recap, briefly analyze what you got wrong and refine your hypothesis. Do NOT make any tool calls yet - Phase E2 will begin with a separate prompt.

Agent

## Analysis of T1 Errors

I made **8 false negatives** (missed true patterns) and **1 false positive**:

**False Negatives (predicted False, was True):**  
- Seq 0: "why does it matter? Where does it come from?"  
...

User

It is now Phase E2.  
Follow your instructions to probe your hypothesis.  
The 'feature\_id' argument you must use is 30. Number of probes remaining: 10.

Tool Call

```
get_activations( ===== Probe sequence ===== Do you like apples? Do
you like oranges? Do you like bananas?
-----)
Token activations:
'Do': 0.000
'you': 0.000
'like': 0.000
'apples': 0.000
'?: 0.000
'Do': 0.000
'you': 0.000
'like': 0.000
'oranges': 0.000
'?: 0.000
'Do': 0.000
'you': 0.000
'like': 0.000
'bananas': 0.000
'?: 0.000
(No activations above threshold)
...)
```

User

Reflect on your probe results. Be concise.

Agent

**Reflection:**

My hypothesis about anaphora/repetition was wrong...

[Probing continues for remaining iterations...]

1134  
1135  
1136  
1137  
1138  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187

```
User
It is now Phase T2.
Follow your instructions given the following sequences:
===== Sequence 0: =====
my energy in answering to the \why" question. Why should people
share? Why should they care about the LoS? Why should they spend
their time on
...
```

```
Tool Call
submit_predictions( "predictions": "0": true, "1": false, ...
)
```

```
Tool Result
"0": true, "1": false, ...
```

## D.2 TOOL DEFINITIONS

The agent has access to two tools during the study:

```
Tool Call
submit_predictions(predictions: dict[str, bool]) -> str
```

Submit predictions for whether each sequence contains the hidden pattern. The predictions argument maps sequence IDs (as strings) to boolean values.

```
Tool Call
get_activations(feature_id: int, sequence: str) -> str
```

Probe a custom sequence to see which tokens activate the pattern. Returns the sequence with activated tokens marked and their activation values.

## E IMPORTANCE CRITERION

We originally sought features that were both complex and *important*. However, we found that these features often encoded relatively simple grammatical concepts. Here, we provide our definition of importance and visualize the relationship between complexity and importance.

### E.1 IMPORTANCE DEFINITION

For an input sequence  $\mathbf{x}$  and token position  $i$ , let

$$z_{k,i}(\mathbf{x}) \in \mathbb{R}_{\geq 0} \tag{2}$$

denote the activation of SAE feature  $k$  at position  $i$ .

Let  $v_j(\mathbf{x}, i)$  be the  $j$ -th top- $M$  predicted next token at position  $i$ . Let  $\mathbf{d}_k \in \mathbb{R}^d$  denote the  $k$ -th column of the SAE decoder matrix  $\mathbf{W}$ , and let  $\tilde{\mathbf{d}}_k$  and  $\tilde{\mathbf{u}}_v$  denote the  $\ell_2$ -normalized SAE decoder and unembedding vectors, respectively. Define the *direct logit attribution* (Wang et al., 2022)

$$a_k(\mathbf{x}, i, j) = \tilde{\mathbf{d}}_k \cdot \tilde{\mathbf{u}}_{v_j(\mathbf{x}, i)}, \tag{3}$$

which is a linear approximation of how strongly feature  $k$  contributes to increasing the logit of the predicted token  $v_j(\mathbf{x}, i)$  at position  $i$  in input  $\mathbf{x}$ .

1188 We define the *weighted attribution* as

$$1189 \quad w_k(\mathbf{x}, i, j) = z_{k,i}(\mathbf{x}) a_k(\mathbf{x}, i, j), \quad (4)$$

1191 i.e., it scales the direct logit attribution by the feature’s activation strength.

1192 In order to compute feature importance, we consider the expected weighted attribution and expected  
1193 activation of feature  $k$ , where the expectation values are taken over dataset samples  $\mathbf{x}$ , token posi-  
1194 tions  $i$ , and top- $M$  predicted tokens  $j$ :

$$1196 \quad \mathbb{E}[w_k] := \mathbb{E}_{\mathbf{x}, i, j}[w_k(\mathbf{x}, i, j)], \quad \mathbb{E}[z_k] := \mathbb{E}_{\mathbf{x}, i}[z_{k,i}(\mathbf{x})]. \quad (5)$$

1198 We define the importance  $I_k$  of feature  $k$  as

$$1200 \quad I_k = \frac{\mathbb{E}[w_k]}{\mathbb{E}[z_k] + \varepsilon}, \quad (6)$$

1202 where  $\varepsilon = 10^{-8}$  is a small constant to prevent division by zero. The importance  $I_k$  therefore mea-  
1203 sures how strongly feature  $k$  tends to push the model toward its own preferred next-token predictions,  
1204 conditional on the feature being active.

1206 This filtering of importance parallels Schut et al. (2023)’s approach, but while they measured im-  
1207 portance by presence of a feature in stronger, but not weaker models, we operationalize importance  
1208 directly via a feature’s impact on next-token prediction.

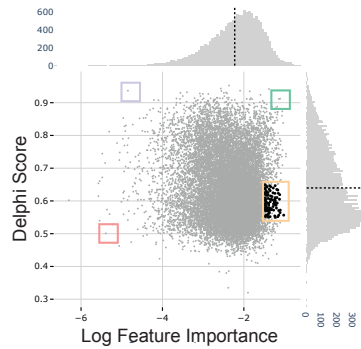
## 1210 E.2 IMPORTANCE SETUP AND RESULTS

1211 For our importance calculations, we used the same dataset as described in Section 4. We set  $M = 10$   
1212 for the top- $M$  predicted tokens to consider in Equation (6).

1214 Figure 10 shows feature importance versus Delphi scores for all 16,384 features, illustrating the  
1215 four quadrants defined by importance and complexity. Features 7759 and 6213 exemplify how  
1216 importance filters for grammatical concepts: Feature 7759 activates on the word “should”, while  
1217 feature 6213 activates on nouns following intensifying adjectives (e.g., “very”, “sought-after”).

1218  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1240  
1241

1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1290  
1291  
1292  
1293  
1294  
1295



Feature 6213

enal natriuresis. Instead CNP is a critical regulatory **hormone** in the bone, where it  
 quickly established themselves as **one** of the most sought-after jazz **ensembles** in Euro  
 is a very social **neighborhood** with a very active HOA and scheduled group activities

Feature 7759

and immense amounts of smoke. It is a bomb, and **should** definitely be treated as  
 it not work as expected for animations etc, but **should** work fine otherwise.  
 be set here. And of course, they **should** be: Brisbane's shadows are as dark and g

Feature 13130

$3, \} + \{\mathbb{N}\}_0^{\alpha}_{i_2}$ . **Thus the claim is proven.**  
 $\tau(i)$ . **Thus  $\alpha_{\tau(i)}$  is a rational linear**  
 $G_2$ . **Hence it is contradictory that**

Feature 3989

```
on parseArguments(url, options, callback) {  
  if (!callback && typeof options ===  
  (divbar);  
  mainWnd->Add(btn_ok);  
  mainWnd->Add(  
  / property to be displayed.  
  */  
  PropertyDescriptor() :hidden(false){};
```

Figure 10: **Feature importance vs. automated interpretability detection accuracy scores for all 16,384 features of the SAE.** Histograms are shown on the top and right axes for importance (calculated via direct logit attribution) and Delphi’s autointerp detection accuracy, respectively. Dashed lines indicate the mean detection accuracy (0.64) and mean importance (0.008). Each of the four coloured points exemplifies a feature in one of the four quadrants of importance vs. detection accuracy.