Early Results for Task-Based Model Routing using SAE Features

This blog post describes a set of preliminary experiments that investigate if we can use Sparse Autoencoder (SAE) features to predict which prompts are best suited for which models. This is the first exploratory step we are undertaking for a larger project that will investigate if we can employ mechanistic interpretability methods to improve dynamic model routing.

In this blog post, we describe our results so far for the first stage of these preliminary experiments, investigating whether:

Given a prompt, when a model provides a correct output, are the top-activated SAE features relevant?

The first section of this blog post outlines our long-term project plans, and the second section describes the results of our first set of preliminary experiments.

Overall, we found that while the current experiments have much room for improvement in their setup design, they also inform us on how we can pursue better designed experiments at our next steps, and served as a helpful learning experience. We found that this project has been a useful trial-and-error teaching endeavor, with mentorship help from Apart Research and Martian.

Part 1: Utilizing Mechanistic Interpretability for Similarity Routing

Long-Term Project Motivation

Model Routing is a method of directing an incoming prompt to a model best suited for the prompt in order to optimize cost, latency, and performance. However, most existing approaches assume a static set of LLMs for routing, whereas in practice, the pool of available models often changes over time. To address this challenge, dynamic routing allows routing to new and unseen models at inference-time. In particular, the Universal Router approach tackles this problem via **Task-Based Similarity Routing**:

1. First, each model is represented by how well it performs on previous clusters of prompts.
2. Then for each new prompt, its most similar prompt-cluster is chosen.
3. Lastly, the model with the best performance for that prompt-cluster is selected.

Still, these model representations are defined over a fixed set of representative prompts, which may not generalize to certain out-of-distribution (OOD) tasks. This black box approach may not fully capture a model's capabilities, especially hidden ones, as important parameters may not be represented by the selected set of representative prompts, and would be "lost in translation".

Therefore, this method can potentially be improved upon by adding in model representations which more closely preserve model parameters. Henceforth, we are undertaking a long-term project that will investigate if we can employ mechanistic interpretability techniques to find certain interpretable and continuous spaces to embed model parameters in, such that we can:

1. Translate any prompt into a representation in terms of properties of a space with interpretable (and interpolatable) regions.

2. Determine how well these interpretable regions correlate with model performance.

3. Represent models in these interpretable regions correlated with performance

4. Measure model similarities based on their distances in these interpretable spaces, with the goal of selecting models with similar performances at less expensive costs.

5. Utilize these mechanistic interpretability properties in routing.

Overall, this project will test the hypothesis that there exist certain types of similar internal model representations which are strongly correlated with similar model output performances.

SAE Routing Project Motivation

Sparse Autoencoders (SAEs) are a method used in mechanistic interpretability research for obtaining monosemantic features from model activations. To tackle the first steps of this long-term project, we start with a simpler sub-project that investigates if we can use SAE features to predict which prompts are best suited for which models. In this blog post, we describe our results so far for the first step of this sub-project:

Given a prompt, when a model outputs a correct for it, are the features relevant?

Outline of Next Steps for SAE Routing

Before describing these results, we present the big picture of how this first step ties in with the overall sub-project, in order to justify why we pursue it.

1. Test: When a model gets answer incorrect, are the features also irrelevant?
- This is an important part of the experiments to ensure that certain prompt-relevant SAE features are not activating for every answer. If they are, there is no link between those

features and model performance. As such, our current work is incomplete and inconclusive until we conduct these experiments.

- We may find that while certain relevant SAE features activate independently of model performance, there may be certain features do. We will look into approaches which can learn to distinguish these performance-linked SAE features.

2. More rigorous statistical analysis on the correlation between SAE features and model performance. The current analysis are not sufficient or quantitative enough, such as lacking correlation or causal ablation studies. Overall, just because a semantically-relevant SAE feature activates, does not mean it’s causally relevant for the task, so more studies are required to assess if there is evidence for an actual link.

3. Check if routing by features accurately chooses the model with the highest performance

We have not incorporated “actual” routing yet into our experiments, which should choose models based on both performance and cost. Our routing experiments will start with broad routing, in which we set up our model pool with a model that specializes in one domain (e.g., math), and another that specializes in a different domain (e.g., code). We will first start by routing only with SAE features, and see if most math prompts are sent to the math model, and vice versa. We will first run both models through all our dataset prompts and obtain performance and cost scores. Then, we will check if given a prompt, the router selects the model with the best performance for the prompt, and conduct a similar analysis for cost.

We only detail the most immediate next steps. Future work will investigate more areas, such as performance on OOD tasks. We will also investigate approaches beyond using SAE features, such as using probe classifiers (without any LLM judges) on model activations to directly predict which models to route to.

Limitations of this stage to address in future work

Before describing our results, we list out the most important limitations in our experimental designs, which we will improve upon:

1. These experiments use judges to assess model outputs

These early experiments use a judge that takes in both model outputs for inference-time prompt and SAE features, in order to separately judge model performance and feature relevancy. We do not plan to take in model outputs when we perform “actual” routing, as we will design a pre-hoc approach that does not rely on model outputs at inference-time.

We only use judges as a way to determine SAE feature label relevancy for a given prompt; we do not believe SAE features are suited for actual model judging on model output quality, as these tasks require alignment with ground truth human annotations, whereas SAE features are several steps away from ground-truth data and their auto-labeling can be subjective.

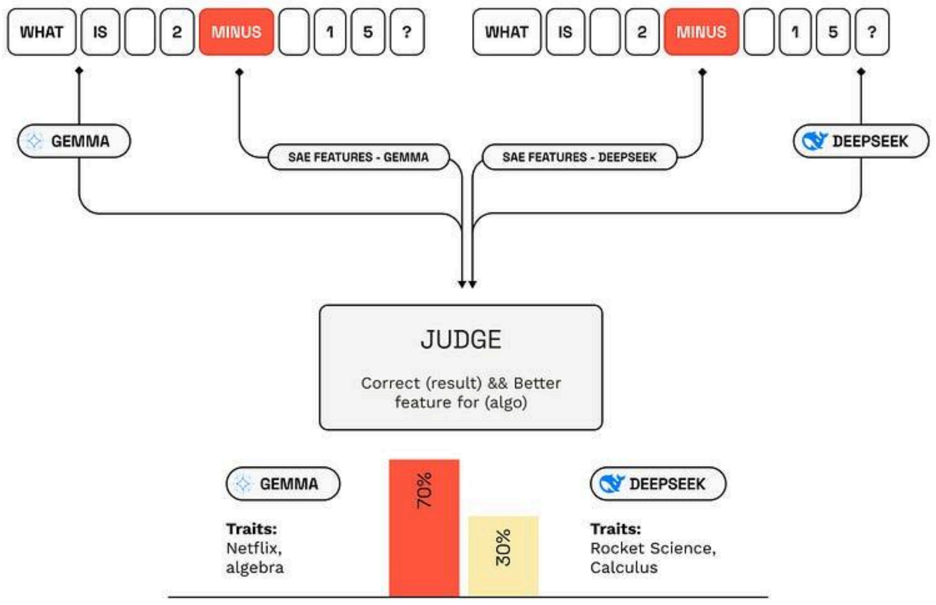
2. The auto-labeled SAE features can have issues with subjective interpretation during analysis (both human and LLM-labeled). We will avoid subjective analysis that we may have conducted too strongly.

Part 2: Early Observations on SAE Routing

This part of the post will be explained by Dhruv:

Previous Blog Post Results

In my last blog post, I explored using **Sparse Autoencoders (SAEs) for model routing**. The main objective in my earlier setup was to test whether the *internal features* that LLMs use when solving math word problems (our sample task) could help make smarter routing decisions between models. The pipeline was straightforward: extract the top-20 SAE features from each model, pair them with model outputs, and then rely on a judge to evaluate task conformity and accuracy. For this study, I worked with two models, **Gemma-2** and **DeepSeek-R**, since their SAEs are publicly available.



Original Experimental Setup—Summarized

While running this experiment, I realized there's room to make the final judgment more objective. Right now, it depends on LLM-as-a-Judge inference for task relevance, which can have issues such as judge bias. Other reliable alternatives might involve Natural Language Inference (NLI), embedding similarity, or even structured output checks.

From the initial (dummy) run, some patterns emerged:

1. **Gemma-2** often became the default choice.
2. **DeepSeek-R**, while picked less frequently.

Evaluation on Math and Coding Datasets

The next step was to benchmark on real-world datasets, as the initial findings from the first blog post evaluated on a synthetic dataset generated via ChatGPT.

Experimental Setup

Models and SAEs:

Gemma-2-9b-it uses 16k width JumpRelu SAEs, and Deepseek-R1-Distill uses 32K width topk-ReLU SAEs.

Datasets:

- Math: UGMathBench contains over 5,000 undergraduate-level math problems across 16 subjects, sourced from HKIST's online grading platform. Each problem is annotated with human-provided metadata such as topic and subtopic labels.
- Code generation: HumanEval Dataset consists of 164 handcrafted Python programming problems each including a function signature, a docstring, and a set of unit tests.

Methodology:

For each query, we extracted the top-50 activated SAE features per model and supplied them with an instruction prompt to the Judge, for which we used **Llama-3.1–80B.** We assess both "Model Correctness" and "Feature Relevance" via LLM-as-a-Judge.

We used the following metrics as performance metrics:

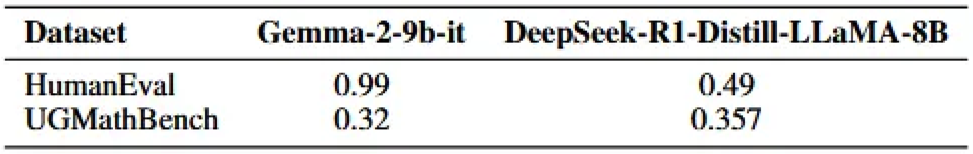
1. UGMathBench: we ran the model output through a judge to provide a binary correct/incorrect score. This Judge run is separate from the judge run to determine SAE feature relevancy.
2. **HumanEval**: pass@k, which outputs a binary passing score if, among k generated samples, at least one sample passes all unit tests.

"Correct and SAE-Relevant" Metric: We measure when a prompt begets both correct model output and feature relevancy as follows:

Let C be the model performance (1 if correct). Let T be the feature relevancy (1 if relevant).

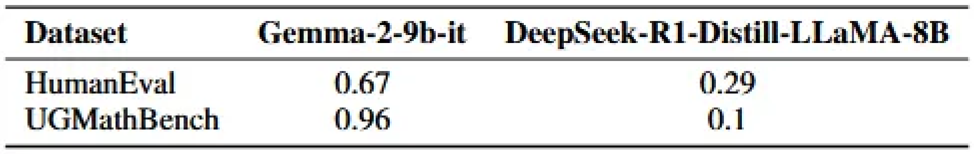
Then we assess if a prompt begets both “correct performance” and “feature relevancy” if both C and T are 1. We take the fraction of “both correct and relevant” scores over all prompts to obtain an aggregate metric, which we report in our tables below.

Results



Dataset	Gemma-2-9b-it	DeepSeek-R1-Distill-LLaMA-8B
HumanEval	0.99	0.49
UGMathBench	0.32	0.357

Judge precision: Probability of correctness given the model’s features were better (C|T)



Dataset	Gemma-2-9b-it	DeepSeek-R1-Distill-LLaMA-8B
HumanEval	0.67	0.29
UGMathBench	0.96	0.1

Judge recall: Probability the model’s features were better, given the response was correct (C|T)

On **HumanEval**, Gemma’s features appeared to align more clearly with correctness, while DeepSeek seemed disadvantaged. However, part of this discrepancy may stem from DeepSeek’s `<think>` responses, which complicate code parsing and reduce *pass@1*. This likely caused true positives to be missed, underestimating its accuracy and lowering precision.

On **UGMathBench**, the scores were more concerning, suggesting that further introspection is needed before drawing strong conclusions.

Conclusion

We conducted some of the first steps in obtaining empirical results for linking SAE features with model outputs. Our experiments are thus far incomplete to draw decisive conclusions; we will continue our experiments to assess if “relevant” SAE features activate on prompts which the model does not perform well on, and then try to see if there exist SAE features which are specific to output-correctness. This will help us assess not just if a basic approach of using SAE features is not sufficient for routing, but how it is insufficient.

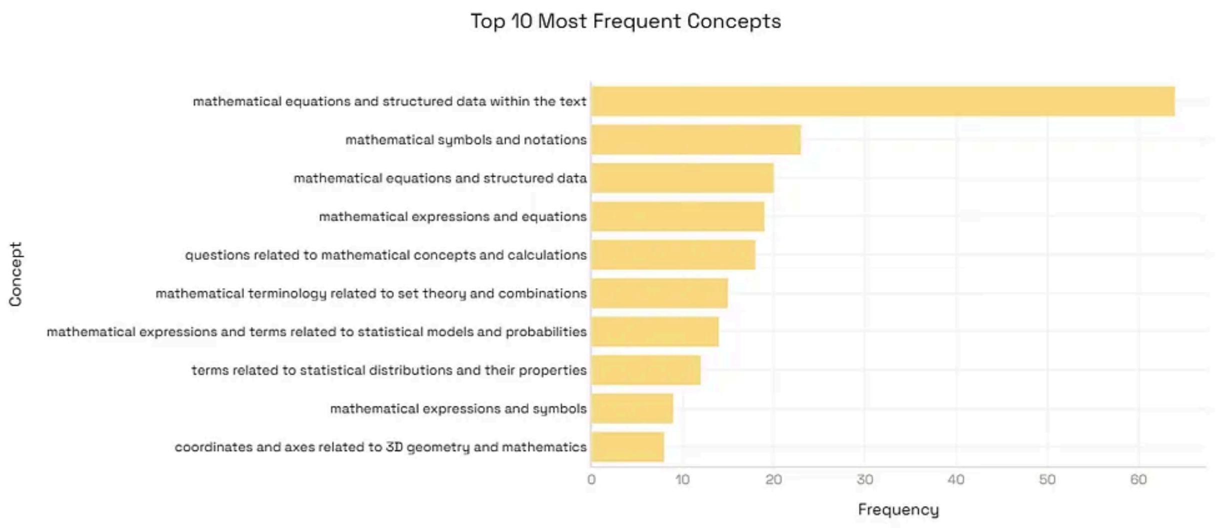
One of the major aims of the project was to obtain early findings that can help inform us on how to explore a wide range of potential alternative interpretability approaches; we believe that pursuing this project provides us with helpful insights on flaws to address. In our Appendices, we present some negative results to assist the community in developing improvements.

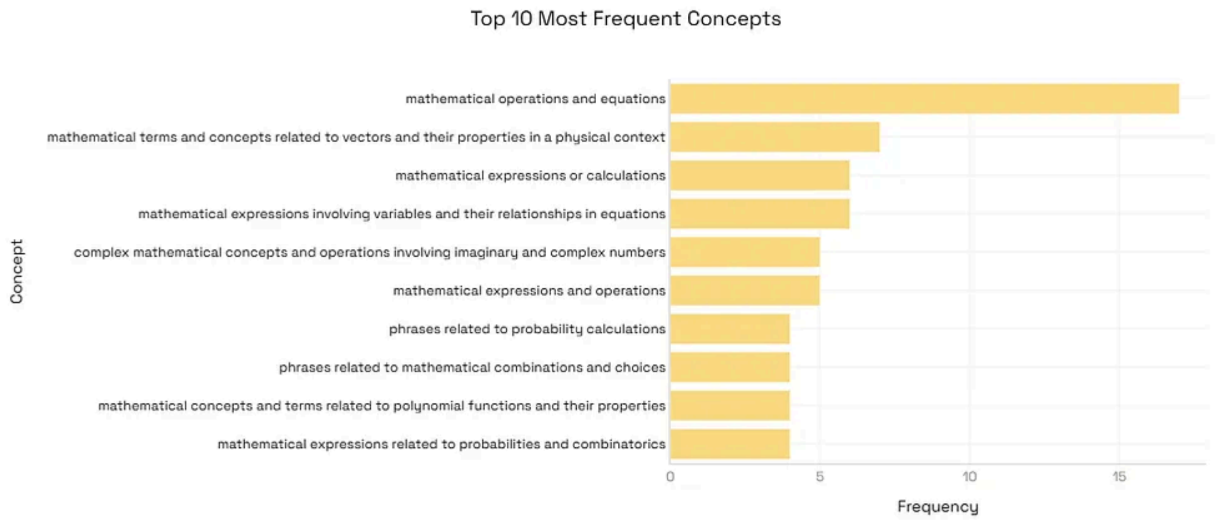
In the long-term, we plan to use this project to justify more sophisticated approaches that employ interpretability for routing. In general, our plan is to use mechanistic interpretability observations to complement black-box prior performance data; in other words, we do not seek to solely use model internals for similarity-based model routing, but rather, to enhance it for OOD tasks, or to reduce the number of representative tasks required for good routing.

Appendix: Results with incomplete/flawed experimental setups that require more work

Results: Subject Wise breakdown of UGmathbench

We further broke down results for UGMathBench on its six broad mathematical subjects: *Algebra, Arithmetic, Combinatorics, Complex Analysis, Geometry, and Probability*. For each input, a judge LLM took in both features and model outputs, and selected a model only if both the features were relevant, and the model was correct.

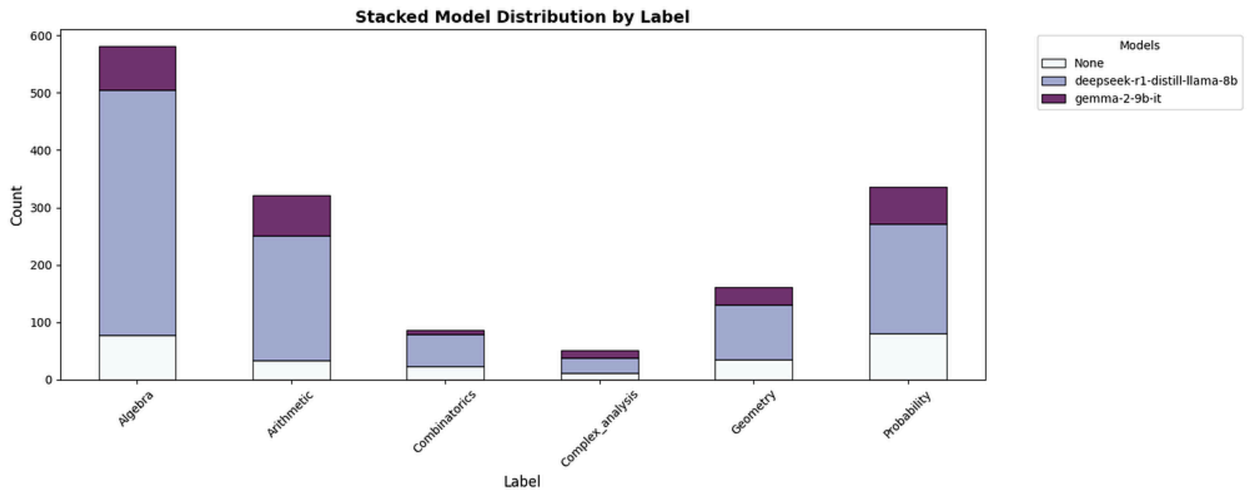




However this experiment had a major flaw: clearly, it is possible that both models could have correct outputs and relevant features. Then, forcing the judge to pick one model is completely arbitrary. Second, this is incorrectly framed as a “routing approach”; routing should be done to optimize model performance and cost, and not route just because SAE features are relevant (which isn’t shown yet to be correlated with performance or cost; more experiments are required).

Lastly, this experiment also has issues with ambiguous signals, as it does not separate whether routing relies on SAE features or outputs. This is flawed due to the joint evaluation of accuracy and task relevance**:** the judge assesses both accuracy and task relevance within the same prompt. This conflation risks confusing the LLM-as-a-Judge, introduces variability in outcomes, and makes it difficult to measure in isolation how much task relevance alone contributes to accuracy.

Nonetheless, we report the distributions of chosen models for each category, aggregated over approximately 1,500 examples:



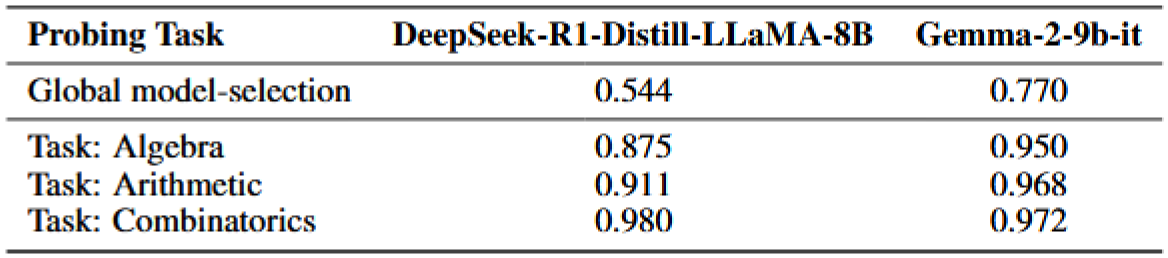
From a broader lens, DeepSeek R is the preferred choice across all categories. The largest margins appear in Algebra (73.5%) and Arithmetic (67.6%), where DeepSeek-R accounts for a majority of routed examples. Gemma-2, though selected less frequently, demonstrated consistent specialization: it is disproportionately chosen across most domains.

Appendix: Probing to assess if SAE Activations linearly encode routing decisions

Having found some evidence that SAE Features can provide a differentiating signal between models, we thought of validating these trends using the raw SAE activations themselves, looking beyond just their descriptions (which tend to be noisy).

We probe SAE latents to assess whether they (i) validate the routing trends surfaced by our method (global model–selection probing) and (ii) provide signal about downstream task categories (task probing). The central question is: *do these activations, on their own, encode interpretable information aligned with model choice and problem type?*

The setup for this was simple enough: cache SAE Activations and then max pool them across token positions. For model-selection probing: we use these to train a binary classification model for either models: did the given problem route to the selected model or not? For task probing, we use a binary classification model for each task category: does the given problem belong to the given task category or not? For the model we used Logistic Regression.



Probing Task	DeepSeek-R1-Distill-LLaMA-8B	Gemma-2-9b-it
Global model-selection	0.544	0.770
Task: Algebra	0.875	0.950
Task: Arithmetic	0.911	0.968
Task: Combinatorics	0.980	0.972

SAE latent probing accuracies—Logistic Regression

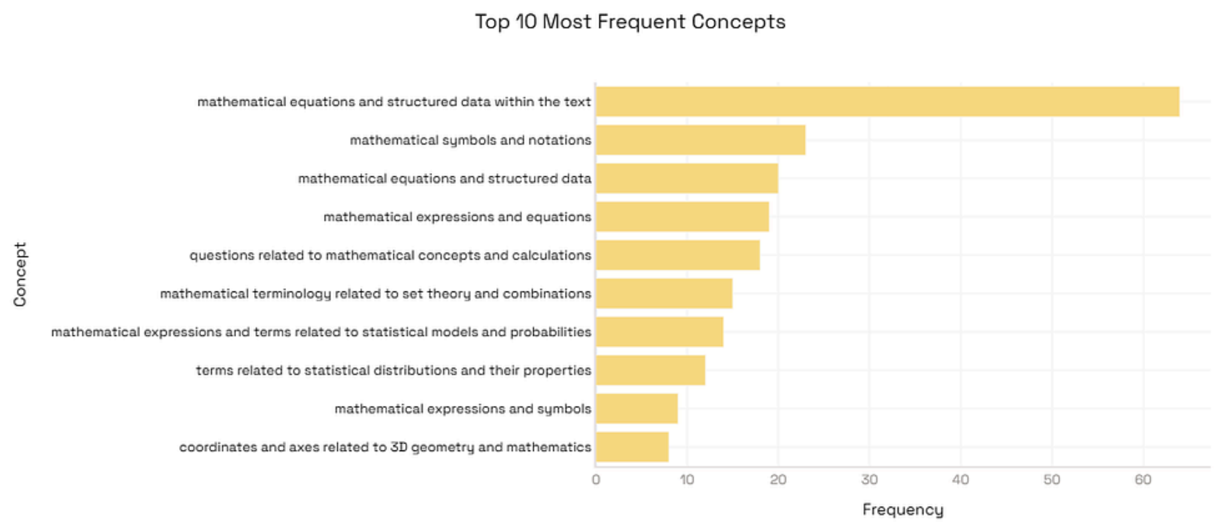
> The task probing results are not spurious, they also tend to have an AUC score of ≥ 0.95. While for model-selection probing, these scores remained low around 0.55—slightly better than a random model

For global model-selection probing, Gemma's latents achieve substantially higher accuracy (0.770) than DeepSeek (0.544). For task probing, both models demonstrate strong performance (≥ 0.875). These patterns might hint at meaningful structure in SAE representations related to both routing preferences and task categories, but would require more robust probing techniques for validation. The results might be subpar due to improper hyperparameter tuning and only suggest correlation not causation of SAE Activations and routing preference/task relevance.

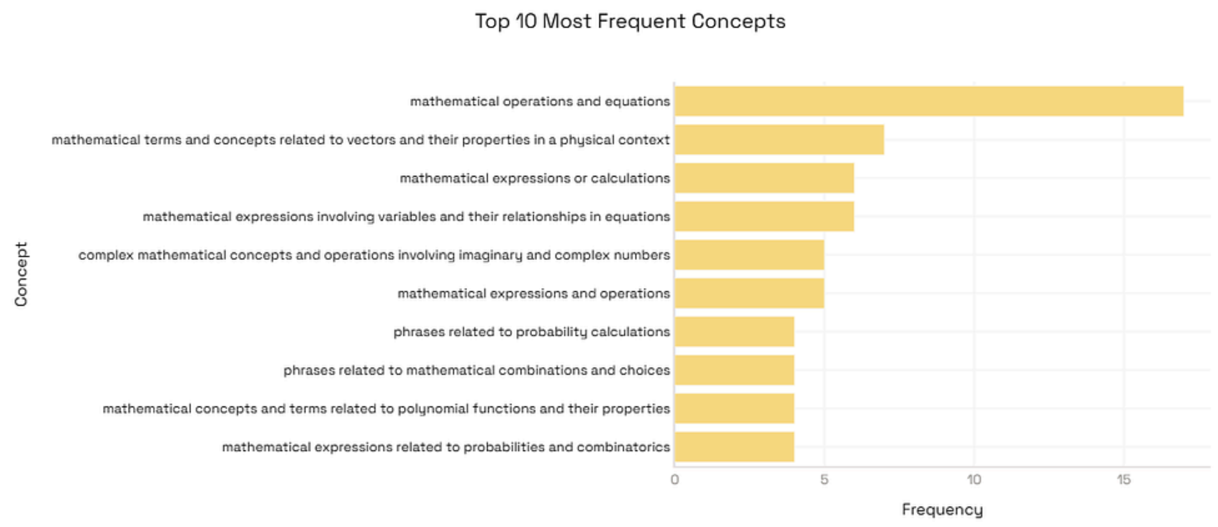
Appendix: SAE concept-wise breakdown

To further probe routing behavior, we analyze the concepts most associated with judge decisions. Here, the feature counts have been normalized by subject split size, to account for the natural skew for certain subject's features, due to their samples accounting for larger proportion of the dataset.

Interpreting these results is too subjective; overall, we do not find a clear semantic division of labor between the two models.



Most frequent concepts guiding the judge to prefer DeepSeek-R



Most frequent concepts guiding the judge to prefer Gemma-2

Appendix: Additional Next Steps to Address Limitations

We describe potential next steps for addressing additional limitations that were not stated above:

Judge Limitations: One caveat is that we rely on the Judge to synthesize supporting SAE features when making a routing choice. This introduces subjectivity and weakens comparability, as the Judge often consolidates multiple relevant features under a single label (e.g. ”*features relating to Probability”)*

I ran an informal sanity check on this by comparing embedding similarity between Judge's supporting features for a decision and the actual SAE Features descriptions provided to the Judge. I found good representation for most samples, but an objective analysis remains important.

This reinforces the importance of developing more objective metrics to quantify the task relevance of SAE features.

Alternate Judgement Strategies: We observed that judgement can be sensitive to LLMs used. This might be due to output verbosity, which might make a “think” model seem to have more feature appropriateness. There is scope to explore other techniques like NLI, or structured output to make judgement more robust.

Here is how such a scheme might look like:

1. Accuracy Judgement: We can pull out Accuracy Judgement into a separate LLM call, this allows for a more focused signal, without risking response verbosity to be considered into the feature conformity task
2. Conformity Judgment: This is where we can make the objective a lot more robust. Currently, the LLM synthesizes reasons from the feature list provided of both models. A better method could be to score each feature individually against the task labels—to which they must conform. There are two ways to do this: NLI and Structured Output (LLM Scoring).
3. Thresholding: Once we have conformity scores across LLMs, we do not need an LLM to synthesize Judgement. We can simply apply thresholding, and a rubric based Judge can be employed to more comprehensively judge “conformity” on relevance and completeness.

Lack of Causal Experiments: Current work operates on SAE Feature descriptions, but through initial experiment we have found these are noisy i.e. even math-features can activate for non-math prompts. Hence, its important to try out causal ablations/perturbations to verify and validate downstream effects. This helps us also comment on the usefulness of SAE Features, which are currently in question, as per probing results. Hence, it is a useful addition to current work.

Tentative steps may include finding highest impacting, Zero-ablation features. We can zero-ablate top-K SAE Features, and measure the resulting impact on output logits in terms of KL-Divergence. Early results may show that highest activating feature don’t necessarily lead to the greatest impact. We can identify if the conforming features in our judgement overlap with the features that also cause the greatest impact.

Acknowledgements

I thank Apart Research and Martian for generously providing support and direction for this project, with particular thanks to Michael Lan at Martian for his helpful mentorship, guidance and massive help writing this blog.