Sparse Autoencoder Features Do Not Improve Transferable Detection of Instructed Sandbagging in Gemma-2-9B-it

Abstract

Sandbagging, the strategic underperformance of a model on an evaluation, is a central threat to capability and safety evaluations [@vanderweij2024sandbagging]. A natural hope is that pretrained sparse autoencoders (SAEs), which decompose activations into a sparse and more monosemantic feature basis [@cunningham2023sparse; @rajamanoharan2024jumprelu], yield a sandbagging signal that is cleaner and more domain general than the raw residual stream, and therefore a better foundation for a sandbagging detector. We test this directly on Gemma-2-9B-it [@gemma2024report] using the pretrained Gemma Scope JumpReLU SAEs [@lieberum2024gemmascope]. We elicit instructed sandbagging on two task domains, multiple choice arithmetic and multiple choice factual recall drawn from MMLU [@hendrycks2021mmlu], using three paraphrased instructions per condition so that a detector cannot rely on a single verbatim prompt. We then train logistic probes on identical last token residual activations, comparing the dense residual against its SAE feature encoding, and measure detection within domain, across domains, and across both a new domain and a held out instruction paraphrase at once. Both representations separate honest from sandbagging activations almost perfectly within domain, and both carry a sandbagging direction that transfers strongly to the new domain and the unseen paraphrase, with area under the ROC curve around 0.80 to 0.91 against a shuffled label control near chance. Crucially, the SAE features provide no transfer advantage. At the layer with the strongest transfer, the dense probe reaches a hard transfer AUC of 0.901 while the SAE probe reaches 0.846, a mean paired difference of 0.054 in favor of the dense probe across training subsamples, although the two per test set confidence intervals overlap modestly. The dense probe matches or beats the SAE probe on cross domain transfer at all three layers and on the hardest transfer at the primary and late layers, with the sparse basis only marginally ahead at one early layer. We conclude that, for this task, the pretrained sparse basis offers no transfer advantage over the dense residual it is built from, and that a dense linear probe remains at least as strong a detector and a simpler one.

Introduction

Capability and safety evaluations assume that a model tries its best. A model that deliberately answers below its ability, a behavior called sandbagging, can pass a dangerous capability evaluation while concealing the very capability the evaluation was meant to surface [@vanderweij2024sandbagging]. Because the model that sandbags is by construction not cooperating with the evaluation, behavioral measurement alone is fragile, and a detector that reads the model's internal state is attractive. Linear probes on residual activations are the standard internal state tool and have repeatedly shown that honesty and truthfulness related concepts are linearly readable [@alain2016understanding; @azaria2023internal; @burns2022discovering].

Sparse autoencoders offer a different basis. Rather than the dense residual stream, an SAE re-expresses activations as a sparse combination of an overcomplete dictionary of features, each intended to be more monosemantic than a raw neuron or direction [@cunningham2023sparse]. Gemma Scope releases pretrained JumpReLU SAEs for every layer of Gemma 2 [@lieberum2024gemmascope; @rajamanoharan2024jumprelu], which makes it cheap to ask whether the sparse basis improves a safety relevant detector. The intuition behind the hope is that if a small number of features encode the abstract intention to underperform, separate from the surface form of the instruction and from the specific task, then a probe built on those features should generalize across tasks better than a probe on the entangled dense stream.

We test this intuition with a transfer first design. The question is not only whether sandbagging is detectable, which prior probing work would already predict, but whether the SAE representation detects it more transferably than the dense representation it is derived from. We measure transfer from arithmetic to factual recall, and we add a stricter test in which the detector must generalize to a new domain and to an instruction paraphrase it never saw during training. The stricter test is necessary because a detector trained and tested on a single fixed instruction can separate two fixed prompt strings perfectly without representing sandbagging at all.

Method

Model and SAEs

We study Gemma-2-9B-it [@gemma2024report], a 9 billion parameter instruction tuned model with a residual width of 3584 and 42 layers. We use the pretrained canonical Gemma Scope residual SAEs for the instruction tuned model at layers 9, 20, and 31, each with a dictionary of 16384 features [@lieberum2024gemmascope]. These are JumpReLU autoencoders [@rajamanoharan2024jumprelu], a variant of the gated design [@rajamanoharan2024gated] that uses a learned threshold for strong fidelity at a given sparsity. On our activations the layer 20 SAE fired about 101 features per token on average, confirming the expected sparsity.

Eliciting sandbagging

We construct two task domains as four option multiple choice questions. The arithmetic domain is 250 synthetic problems of two digit addition, two digit subtraction, and single by two digit multiplication, with numerically near distractors. The factual domain is 250 questions sampled across eight MMLU subjects [@hendrycks2021mmlu]. Each question is presented under two conditions. In the honest condition an instruction asks the model to answer correctly. In the sandbagging condition an instruction asks the model to deliberately choose an incorrect answer in order to appear less capable. To prevent a detector from keying on one exact wording, we use three paraphrased instructions per condition, giving 3000 prompts in total across both domains, both conditions, and three paraphrases.

We verify that the instruction changes behavior with a forced choice readout, scoring the model's preferred option from the logits of the four answer letters at the first answer position. Under the honest instruction the model chooses the correct arithmetic option 39.4 percent of the time and the correct factual option 34.2 percent of the time, while under the sandbagging instruction these fall to 12.2 percent and 14.2 percent. The absolute accuracies are modest because the single token forced choice readout gives the model no room to compute, but the gap between conditions is the relevant signal and it is clear in both domains, which indicates that the sandbagging instruction lowers measured accuracy and is acted on rather than merely paraphrased. Because honest accuracy under this single token readout is itself low, the manipulation establishes the direction of the behavioral effect rather than its full magnitude.

Probing

For every prompt we capture the residual stream at the last prompt token from the target layer, taken from the decoder layer output that the Gemma Scope SAE hooks. We then build two representations from the identical activation, the dense 3584 dimensional residual and its 16384 dimensional SAE encoding. On each representation we fit an L2 regularized logistic probe with balanced class weights, standardizing features on the training split only to avoid leakage. The regularization strength is selected once per representation on an arithmetic train and validation split and then held fixed, so the dense and sparse comparisons use the same protocol.

We report three measurements. In distribution detection trains and tests on held out arithmetic questions, split by question group so the honest and sandbagging versions of one question never straddle the split. Domain transfer trains on arithmetic with all three paraphrases and tests on the full factual set. Hard transfer trains on arithmetic using only paraphrases one and two and tests on the factual set using only the third, unseen paraphrase, so the detector must generalize to a new domain and a new instruction wording at the same time. Every measurement uses a ten seed sweep, a two thousand sample bootstrap confidence interval on the test scores, and a shuffled label control task that measures how much apparent separation arises from probe capacity alone [@hewitt2019control]. The headline comparison is the paired SAE minus dense difference in hard transfer AUC at the layer with the strongest transfer, with a five thousand sample bootstrap interval over seeds.

Results

Both representations detect sandbagging almost perfectly within domain. In distribution AUC is 1.000 at every layer for both the dense and the SAE probe, even though the probe must separate three honest

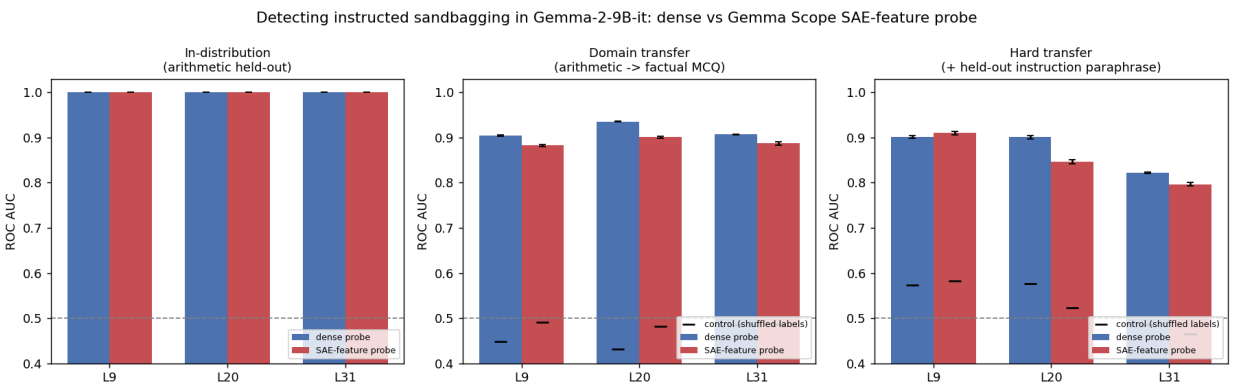

Figure 1: Detection AUC of dense versus SAE-feature probes for instructed sandbagging, per Gemma Scope layer. Left: in-distribution (held-out arithmetic). Middle: domain transfer (train arithmetic, test factual MCQ). Right: hard transfer (train arithmetic paraphrases one and two, test factual MCQ on the unseen paraphrase). Error bars are standard deviation over ten seeds, black dashes are the shuffled-label control, and the dashed gray line is chance.

paraphrases from three sandbagging paraphrases. This confirms that the instructed sandbagging condition leaves a strongly linearly readable trace in the residual stream of Gemma-2-9B-it, consistent with prior internal state probing for honesty related concepts [@azaria2023internal; @burns2022discovering].

The sandbagging direction is also genuinely transferable. Under domain transfer the dense probe reaches AUC 0.904 at layer 9, 0.935 at layer 20, and 0.907 at layer 31, and the SAE probe reaches 0.882, 0.901, and 0.887 at the same layers. Under the stricter hard transfer, which also withholds the instruction paraphrase, the dense probe reaches 0.901 at layer 9, 0.901 at layer 20, and 0.822 at layer 31. The shuffled label control sits between 0.49 and 0.58 across these conditions, far below the real AUC, so the transfer is a property of the sandbagging labels and not of probe capacity. A detector trained only on arithmetic therefore still flags sandbagging on factual questions phrased with an instruction it never saw, which indicates that the direction is to a large degree task general rather than a memorized arithmetic artifact.

The central question, whether the SAE features improve this transfer, is answered in the negative. The dense probe is ahead on cross domain transfer at all three layers, and on hard transfer at layer 20 and layer 31. At layer 20, our primary layer, the dense hard transfer AUC is 0.901 and the SAE hard transfer AUC is 0.846. The mean paired difference of 0.054 in favor of the dense probe is highly consistent across the ten training subsample seeds, with a seed bootstrap interval of 0.051 to 0.057, but that interval holds the factual test set fixed and so measures training stability rather than test sampling. The two per test set bootstrap intervals, 0.870 to 0.930 for the dense probe and 0.809 to 0.887 for the SAE probe, overlap modestly, so we read the result as a clear absence of any SAE advantage rather than a large dense win. The single case in which the SAE probe is nominally ahead is layer 9 hard transfer, where it reaches 0.910 against the dense 0.901, a gap that is small and is not echoed at the other layers or in any domain transfer number. Averaged over the comparison the pretrained sparse basis transfers no better than, and usually slightly worse than, the raw residual it is built from.

The small dense advantage is moreover not specific to sandbagging. On the shuffled label control at layer 20 the dense probe scores 0.054 above the SAE probe, 0.577 against 0.523, a gap that equals the dense advantage on the real task. Because the control task carries no real signal, the dense edge appears just as strongly for random labels as for sandbagging labels, so we do not read it as better encoding of sandbagging. We do not attribute this to a single mechanism, since the dense edge persists even though the SAE code is higher dimensional, and we report it only as an empirical observation. The substantive conclusion is therefore the null one. The SAE features carry no sandbagging specific transfer advantage over the dense residual.

Discussion

The result runs against the intuition that motivated the study. If sandbagging were carried by a few clean monosemantic features, the SAE encoding should have isolated them and transferred at least as well as the dense stream. Instead the dense residual transfers as well or better everywhere. The most likely explanation is that the JumpReLU reconstruction is lossy with respect to exactly the directions a sandbagging probe needs. A pretrained SAE is optimized to reconstruct general activations under a sparsity penalty, not to preserve a specific behavioral axis, and the sparse code can quantize away small but linearly informative components. For a supervised, label driven question such as this one, the dense representation keeps that information while the sparse code partially discards it. This is consistent with the standard guidance that SAEs are tools for unsupervised discovery and decomposition rather than a free improvement to a supervised detector that already has labels.

The practical implication for sandbagging detection is encouraging in one respect and cautionary in another. Encouragingly, a single dense linear probe trained on a cheap synthetic domain transfers to a different domain and a different instruction wording with high AUC, which suggests that a lightweight, cheaply trained monitor for instructed sandbagging is feasible without per task supervision, at least when the sandbagging is induced the same way in training and deployment. Cautiously, reaching for a pretrained SAE in the hope of a better detector is not justified by this evidence, and would in fact cost transfer performance here.

Limitations

The strongest objection is that even our hard transfer may detect the shared semantic content of the sandbagging instructions rather than the act of sandbagging itself. All three paraphrases tell the model to answer incorrectly on purpose, so a probe could generalize through instruction semantics rather than through a behavioral state, and the persistently high in distribution AUC of 1.000 is consistent with that reading. A cleaner future test would probe activations during answer generation, where the model is executing the deception, or would induce sandbagging without lexically describing it. A second limitation is that the forced choice readout understates absolute accuracy, so the manipulation check establishes the direction of the behavioral effect but not its full magnitude. Third, we study one model, one SAE width, and one SAE family, so the conclusion is specific to pretrained Gemma Scope JumpReLU features on Gemma-2-9B-it and may not hold for wider dictionaries, other architectures, or SAEs trained to preserve a behavioral axis. Finally, our control task selectivity is good but not perfect, with control AUC modestly above chance at the early layers, which we attribute to residual probe capacity on standardized high dimensional features.

Conclusion

Instructed sandbagging leaves a strong and surprisingly transferable linear trace in the residual stream of Gemma-2-9B-it, detectable across domains and across unseen instruction wordings by a simple logistic probe. Re-expressing the same activations through pretrained Gemma Scope SAE features does not improve this detector, and the small transfer gap in favor of the dense residual matches the gap seen on shuffled label controls, so it is not specific to sandbagging. For building a sandbagging monitor, the dense residual probe is the stronger and simpler choice, and the value of sparse autoencoders here lies in interpreting what the detector keys on rather than in serving as its input representation.