Building Fast, Evaluating Slow: Pipeline Choices Dominate Autointerpretability Score Variance

Sinie van der Ben [1] **Neele Roch** [1] **Anna Hedström** [1] **Mennatallah El-Assady** [1]

Abstract

Cross-paper comparison of sparse autoencoder (SAE) interpretability often relies on autointerpretability scores. In this evaluation pipeline, a language model (LM) explains each feature, and another LM scores the explanation. For these comparisons to be meaningful, scores must reflect stable properties of the features rather than confounding aspects of the evaluation pipeline. Through systematic experiments across four metrics (simulation, detection, fuzzing, purity), two models (Pythia-160M, Apertus-8B), and four axes of methodological variation, we show that this assumption does not hold. Specifically, we find that **(R1)** methodological variance collectively exceeds architectural variance across all metrics and tested models; **(R2)** each metric exhibits a distinct instability profile, with detection being the most stable and fuzzing unreliable across all conditions; **(R3)** top-k feature rankings do not stay consistent across corpus and draw conditions, masking per-feature instability behind stable mean scores; a failure that cannot be detected by monitoring explanation similarity alone. These findings suggest that cross-paper comparisons based on autointerpretability scores may reflect pipeline differences rather than architectural differences, with implications for the ongoing debate on SAE utility. More broadly, unreliable evaluation slows progress in interpretability research at a time when reliable tools for understanding AI systems are needed. To support evaluation, we contribute a variance decomposition approach, a Stability Check, and a Minimum Reporting Checklist.

[1]ETH Zürich, Zurich, Switzerland. Correspondence to: Sinie van der Ben <sinie.vanderben@example.edu>.

Accepted at the ICML 2026 Workshop on Mechanistic Interpretability, Seoul, South Korea. *Copyright 2026 by the author(s).*

1. Introduction

Autointerpretability scores are widely relied upon as the main instrument for feature interpretability assessment in SAEs (Bricken et al., 2023; Cunningham et al., 2024; Juang et al., 2024; Templeton et al., 2024), and are additionally used in architecture comparisons (Bussmann et al., 2025; Karvonen et al., 2025). However, if these scores reflect evaluation pipeline configurations rather than feature interpretability, such comparisons are potentially meaningless. Proxy metrics such as reconstruction error and sparsity reflect optimization behavior rather than semantic quality (Chanin & Garriga-Alonso, 2025; Minegishi et al., 2025), and SAEs that perform well on these metrics have been shown to detect only a small subset of true features (Korznikov et al., 2026). This has motivated evaluation frameworks that target interpretability more directly (Makelov et al., 2025; Paulo et al., 2025; Puri et al., 2025), yet a fundamental question remains unanswered: *"**Are current autointerpretability evaluation practices stable enough to support cross-paper comparisons, or do they reflect pipeline choices?**"*

In autointerpretability, or automated interpretability, a language model (LM) generates a natural language description of a feature, and a second LM evaluates how well that description predicts feature activations on held-out examples (Bills et al., 2023). This pipeline currently underlies most SAE interpretability evaluations (Cunningham et al., 2024; Karvonen et al., 2025; Templeton et al., 2024). Prior work has identified individual sources of pipeline variation that affect scores, such as sample selection strategies (Juang et al., 2024; Paulo et al., 2025), explainer or scorer model capacity (Juang et al., 2024; Paulo et al., 2025; Puri et al., 2025), and prompt construction (Puri et al., 2025), but no study has systematically quantified their *relative* contribution to total score variance, or assessed whether their combined effect exceeds the variance attributable to architectural differences between SAEs.

After the introduction of SAEs (Bricken et al., 2023; Cunningham et al., 2024), they have been adopted fast in the field of interpretability. Early work demonstrates that identified features can be used to successfully steer model behavior (Templeton et al., 2024), while other applications

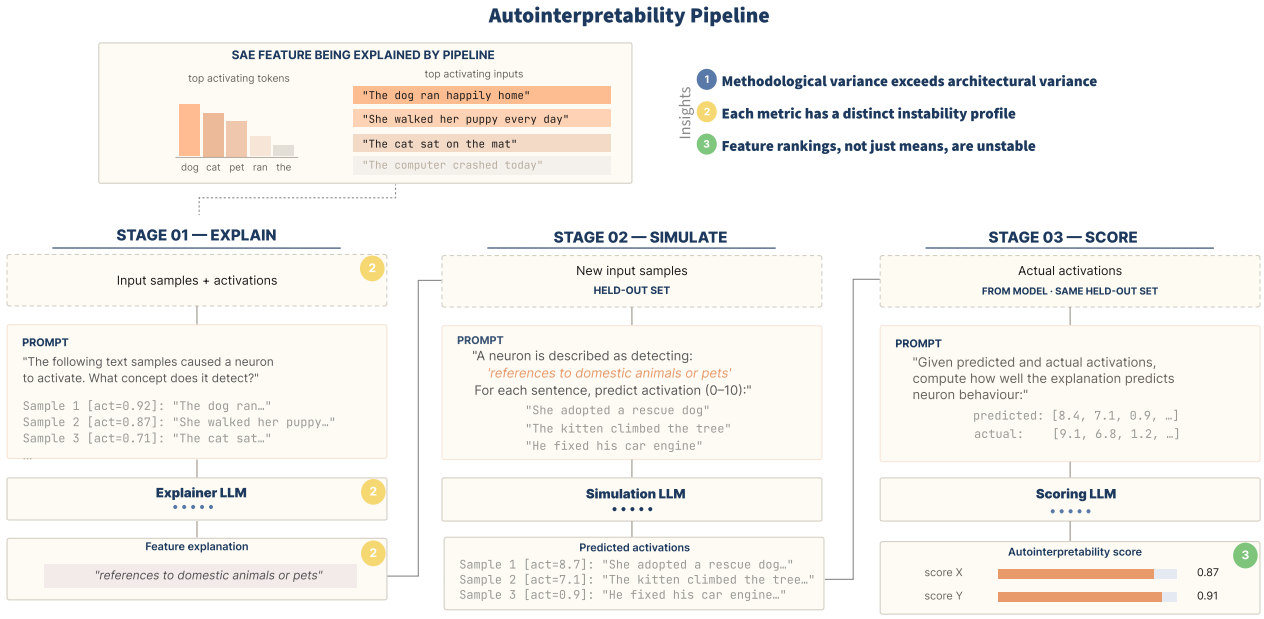

Figure 1. Overview of Automated Interpretability Pipeline with the insights location. **(R1)** is not shown in the figure, but this insight applies to the pipeline as a whole: methodological variance exceeds architectural variance. (R2) shows that each metric exhibits a distinct instability profile, while it is shown through different perturbations of corpora, draws, explainers, and paraphrases. (R3) shows variability in the rankings of feature scores.

include detecting and mitigating emergent misalignment (Wang et al., 2025). Simultaneously, SAE features are sensitive to small input perturbations (Li et al., 2026); SAEs trained on identical data learn different features across random seeds (Paulo & Belrose, 2026), and downstream task performance does not consistently exceed simpler baselines (Kantamneni et al., 2025; Wu et al., 2025). Whether SAEs deliver on interpretability remains an important question for the debate on their usefulness, and that debate requires reliable metrics to resolve.

In this paper, we show that current dominant instruments are unreliable. Through systematic experiments on two models (`Pythia-160M`, `Apertus-8B`) across four metrics (simulation, detection, fuzzing, purity; section 2), we vary four axes of the evaluation pipeline independently and measure instability using ICC (Koo & Li, 2016) and Top-k Jaccard (Kuncheva, 2007) (subsection 2.1). We study metric *reliability* rather than *validity*, under specific pipeline configurations. We do not evaluate whether autointerpretability scores truly capture interpretability, but whether they are consistent enough to support the comparisons the field relies on, which is a precondition for a validity claim. We make the following contributions:

R1 **Methodological variance exceeds architectural variance.** Pipeline choices (corpus, sampling, explainer, and phrasing) collectively account for more score variance than SAE architecture, making cross-architecture comparisons potentially unreliable without standardization (section 3).

R2 **Each metric has a distinct instability profile.** SIMULATION and PURITY are most sensitive to random draw; DETECTION to corpus choice; FUZZING is unreliable across all conditions. No single metric captures the full instability picture, motivating multi-metric evaluation (section 3), confirming statements of earlier research (Puri et al., 2025).

R3 **Feature rankings show instability beyond mean scores.** Mean scores mask per-feature instability, because top-k feature sets identified on one corpus or draw have near-chance overlap with those from another, with direct consequences for any application that selects specific features (section 3).

Additionally, we find that explanation similarity cannot diagnose score instability, as similarity scores remain high across all conditions even when scores vary substantially. Based on these findings, we propose a Stability Check and a Minimum Reporting Checklist (section 4) to support researchers in diagnosing evaluation reliability before interpreting score differences as architectural effects. Our findings do not resolve whether SAEs are useful, but establish what is required to answer that question reliably. A survey of SAE evaluation practices across the literature is provided in Appendix A. Our code is publicly available [1].

[1] https://github.com/sinievanderben/autointerpret_methodology

2. Preliminaries

This section establishes the experimental setup for testing the reliability of autointerpretability evaluation methods. Our goal is to quantify how much the scores depend on pipeline choices, relative to the architectural differences they are meant to capture. To this end, we evaluate the stability of autointerpretability scores across four axes of methodological variation: evaluation CORPUS, random DRAW of activating examples, EXPLAINER model, and explanation PARAPHRASING, and one architectural axis, using three SAEs of varying sizes for both models. Full experimental details are in Appendix B.

Models and SAEs. Experiments were conducted on SAEs trained on `Pythia-160M` (Biderman et al., 2023) and `Apertus-8B Instruct` (Hernández-Cano et al., 2025). For `Pythia-160M`, we used publicly available `BatchTopK` checkpoints at layer 8 from SAEBench (Karvonen et al., 2025) with $k = 160$. For the architecture comparison, we additionally evaluate `Matryoshka` (Bussmann et al., 2025) and `Standard` SAEs (Cunningham et al., 2024). For `Apertus-8B`, we trained a `BatchTopK` SAE with $k = 160$ on layer 16; this is the first work to train and evaluate SAEs on `Apertus`. Training details are provided in Appendix D. We opted for these two models and their SAEs as they vary in size and this allows to exclude the effect of source model size.

The model used to generate and score explanations is `Gemini Flash` (Team et al., 2024), varied across explainer conditions with `GPT4o-mini` (OpenAI et al., 2024). All scoring uses `Gemini Flash` as the fixed scorer, isolating explainer identity from scorer variation. The effect of the scorer has been studied before (Paulo et al., 2025; Puri et al., 2025).

Corpora and Features. We use `Pile` (Gao et al., 2020) as the primary evaluation corpus to match the SAE training distribution. For the corpus experiment, three additional datasets are included: `C4` (Raffel et al., 2023), `Wikipedia` (Foundation), and `GitHub` code (CodeParrot, 2022), spanning a gradient from similar to dissimilar relative to training data. All datasets are sampled to 10,000 sequences of 32 tokens and accessed via `HuggingFace`; corpus statistics are reported in Table 6. Per experiment, we target approximately 250 active features, sampling 300 candidates for `Pythia-160M` and 2,000 for `Apertus-8B` to account for lower activation rates. Feature sampling details are provided in subsection B.3.

Autointerpretability Metrics. We evaluate using four complementary metrics. SIMULATION and FUZZING operate at token level; DETECTION and PURITY at the sequence level. All metrics are LLM-scored, with DETECTION, FUZZING and PURITY having binary scores on different levels. Full metric definitions and scoring prompts are in B.8.1.

- **Simulation** (Bills et al., 2023). An LLM predicts token-level feature activations from the explanation alone. The score is the Pearson correlation between predicted and actual activations, $r \in [-1, 1]$; higher is better, $r = 0$ indicates chance.
- **Detection** (Paulo et al., 2025). An LLM classifies sequences as activating or non-activating. The score is AUC over top-activating vs. random negatives, ranging from 0 to 1; 0.5 indicates chance.
- **Fuzzing** (Paulo et al., 2025). Token-level variant of detection: an LLM classifies whether marked tokens within a sequence correctly represent the feature. Score is AUC over token-level predictions, ranging from 0 to 1.
- **Purity** (Puri et al., 2025). Measures whether high activations are exclusive to the target concept. A scorer LLM assigns (0,1,2) labels to natural samples, and Average Precision is computed over feature activation magnitudes.

To assess the semantic consistency of explanations across conditions, we compute pairwise cosine similarity using `E5-small-v2` embeddings and `BERTScore` O1 (Zhang* et al., 2020). Details are in subsection B.7.

2.1. Quantifying Instability

We quantify within-feature instability using three complementary measures. All three track how much the *same* feature's score changes across conditions rather than differences between features. An overview of the metrics is shown in Table 1 and more details are in Appendix C.

Score Reliability. The **Intraclass Correlation Coefficient** (ICC) (Koo & Li, 2016; Liljequist et al., 2019) measures the proportion of total variance attributable to stable between-feature differences (Eq. 1). We generally interpret ICC ≥ 0.75 as good, 0.50–0.75 as moderate, and < 0.50 as poor reliability, reported with 95% confidence intervals. ICC model specification is in subsection C.1.

$$\mathrm{ICC} = \frac{\sigma^2_{\text{between}}}{\sigma^2_{\text{between}} + \sigma^2_{\text{within}}} \tag{1}$$

Ranking Stability. Whether the same top-k features are identified across conditions is measured by **Top-k Jaccard** (Eq. 2) (Kuncheva, 2007). We report *lift* over chance so results are comparable across pool sizes; lift $= 1$ indicates chance-level selection. Details are in subsection C.2.

$$J_k(A, B) = \frac{|T_k^A \cap T_k^B|}{|T_k^A \cup T_k^B|} \tag{2}$$

Variance Decomposition. We fit a random effects model and estimate variance components via restricted maximum likelihood (REML), testing whether methodological variance exceeds architectural variance (Eq. 3). REML caveats are in subsection C.4. We define *methodological variance* as $\sigma^2_{\text{method}} = \sigma^2_{\text{corpus}} + \sigma^2_{\text{draw}} + \sigma^2_{\text{explainer}} + \sigma^2_{\text{phrasing}}$ and test the central hypothesis:

$$\sigma^2_{\text{method}} > \sigma^2_{\text{arch}} \tag{3}$$

3. Dissecting Autointerpretability Instability through Four Findings

Table 2 and Figure 2 summarise the central findings of this paper. Across all four metrics and both models, the variance introduced by pipeline choices, CORPUS, DRAW, EXPLAINER, and PARAPHRASE, collectively exceeds the variance attributable to SAE architecture. At the same time, no two metrics fail in the same way: they fail for different conditions, and the failure modes are invisible relying on mean scores alone Table 15. Critically, explanation similarity metrics such as cosine similarity and BERTScore (Zhang* et al., 2020) cannot serve as a diagnostic for this instability, as they remain high even when scores vary substantially (subsection E.1 and Table 13). We organize our findings around three insights in the next sections.

R1 Methodological Variance Exceeds Architectural Variance

Pipeline choices account for more score variance than architecture across all metrics and both models.

Figure 2 shows that σ^2_{arch} contributes negligibly to total score variance on `Pythia-160M` across all metrics, and remains small on `Apertus-8B` with a peak of 11% for DETECTION, where CORPUS alone contributes 50%. Our central hypothesis $\sigma^2_{\text{method}} > \sigma^2_{\text{arch}}$ holds for every metric on both models. This is not a marginal result: for SIMULATION on `Pythia-160M`, DRAW alone accounts for 59% of the total variance, while the architecture contributes less than 7%. We show the percentages with confidence intervals in Table 14.

However, low ICC under CORPUS variation does not underlie the same interpretation as low ICC under DRAW or PARAPHRASE. A feature that scores differently on `Wikipedia` than on `Pile` may be legitimately domain-specific rather than unreliably scored. Despite these remarks, poor ICC persists even between `Pile` and `C4`, two general web text corpora, for which domain specificity cannot explain the instability. This instability within similar domains points to corpus sampling as a source of measurement noise rather than feature specialization.

To contextualize this directly, Figure 4 shows Cohen's d effect sizes across three SAE architectures (`BatchTopK`, `Matryoshka`, `Standard`), with all methodological choices held fixed. Unlike within-architecture experiments, cross-architecture comparisons do not permit per-feature analysis via ICC or Jaccard, as features are not matched across architectures; therefore, we compare score distributions instead. On `Pythia-160M`, the largest pairwise effect is `Matryoshka` vs. `Standard` on SIMULATION ($d = -0.44$), falling in the small-to-medium range, yet the same pair shows $d = 0.01$ on FUZZING and $d = -0.10$ on PURITY. On `Apertus-8B`, all pairwise effects fall in the small range. Importantly, effect sizes are inconsistent in direction across metrics because a pair that scores positive and higher on one metric does not systematically score positive and higher on the others. The practical implication is direct: researchers comparing two SAE architectures using autointerpretability scores on a single corpus with a single random draw cannot reliably attribute observed differences in scores to architectural quality.

R2 Each Metric has a Distinct Instability Profile

SIMULATION *and* PURITY *are most sensitive to random draw;* DETECTION *is most sensitive to corpus choice;* FUZZING *is unreliable across all sources.*

SIMULATION. SIMULATION shows high variance across conditions. It has a negative ICC value under the DRAW variation, meaning repeated draws yield feature rankings that are *less consistent than chance* (Table 2). CORPUS instability is similarly severe, with ICC near zero on `Pythia-160M` (0.028), yet SIMULATION becomes more reliable under EXPLAINER variation; the absolute scores are volatile across explainers (Figure 6). Under PARAPHRASE, ICC and Jaccard Lift (Table 2 and Figure 3) are good for both models, confirming that SIMULATION responds to semantic content when evaluation inputs are held fixed.

DETECTION. DETECTION is the most stable metric overall, although still low ICC scores for DRAW and CORPUS, though corpus sensitivity may partly reflect genuine feature domain-specificity rather than pure measurement noise. Under EXPLAINER variation, ICC reaches $0.846\ [0.796, 0.886]$ on `Apertus-8B`. Researchers using different explainer models would obtain similar mean DETECTION scores but disagree substantially on which features are most interpretable (Figure 3). Under PARAPHRASE, DETECTION shows the highest Jaccard lift of any metric ($42.87\times$ at $k = 10$), confirming robustness to surface wording.

FUZZING. FUZZING is unreliable across every variation, with the least reliable result for CORPUS. Despite achieving the most positive results among metrics on DRAW, the ICC and Jaccard Lift remain poor relative to other metrics across

Table 1. Summary of metrics used to evaluate within-feature instability for the four different scores.

Metric	Level	Higher/Lower	Question Answered
ICC	Per-metric	Higher	Is this metric reliable across conditions?
Top-k Jaccard	Per-condition-pair	Higher	Will my top-k features change?
Variance Decomp %	Global	Observational	What contributes more to the variance?

Table 2. ICC with 95% confidence intervals for each source of variation and metric, for `Pythia-160M` and `Apertus-8B`. ICC(1,1) used for DRAW; ICC(2,1) for all other sources. Interpretation follows (Koo & Li, 2016): ≥0.90 excellent, 0.75–0.90 good, 0.50–0.75 moderate, <0.50 poor.

Source	Model	SIMULATION	DETECTION	FUZZING	PURITY
CORPUS	`Pythia`	$0.028\ [-0.018, 0.085]$	$0.151\ [0.087, 0.207]$	$0.076\ [0.007, 0.153]$	$0.122\ [0.063, 0.185]$
	`Apertus`	$0.155\ [0.082, 0.230]$	$0.414\ [0.267, 0.531]$	$0.018\ [-0.089, 0.124]$	$0.110\ [0.027, 0.187]$
DRAW	`Pythia`	$-0.021\ [-0.073, 0.036]$	$0.178\ [0.126, 0.232]$	$0.349\ [0.280, 0.414]$	$-0.010\ [-0.052, 0.038]$
	`Apertus`	$-0.049\ [-0.094, -0.006]$	$0.071\ [0.014, 0.132]$	$0.162\ [0.083, 0.234]$	$-0.078\ [-0.109, -0.045]$
EXPLAINER	`Pythia`	$0.568\ [0.362, 0.728]$	$0.747\ [0.591, 0.854]$	$0.295\ [-0.133, 0.647]$	$0.480\ [0.318, 0.627]$
	`Apertus`	$0.435\ [0.250, 0.615]$	$0.846\ [0.796, 0.886]$	$0.129\ [-0.125, 0.370]$	$0.289\ [0.140, 0.436]$
PARAPHRASE	`Pythia`	$0.860\ [0.783, 0.930]$	$0.824\ [0.699, 0.895]$	$0.240\ [0.034, 0.498]$	$0.650\ [0.523, 0.753]$
	`Apertus`	$0.930\ [0.893, 0.958]$	$0.926\ [0.888, 0.952]$	$0.267\ [0.114, 0.426]$	$0.849\ [0.792, 0.897]$

most conditions, even for PARAPHRASING. A metric that is not robust to paraphrasing responds to surface wording rather than semantic content. Combined with confidence intervals spanning zero for both CORPUS and EXPLAINER, we conclude that FUZZING should not be used as a standalone reliability measure.

PURITY. PURITY measures the semantic coherence of activating examples, as reflected in the high variance for DRAW, which accounts for 67% and 62% of its total variance across both models. This indicates that the semantic coherence of a feature is determined by *which examples happen to be sampled*. Corpus sensitivity is partially expected: distributional shift across corpora may reflect genuine feature specialization or polysemanticity rather than pure measurement noise. Similar to SIMULATION and DETECTION, the highest ICC score is reached for PARAPHRASE.

R3 Feature Rankings show Instability beyond Mean Scores

Mean scores appear stable across conditions, masking per-feature instability. Top-k feature sets identified under one corpus or draw can have near-chance overlap with those identified under a different corpus or draw.

Mean scores are stable across most conditions (Table 15). For all four corpora, the mean SIMULATION score on `Pythia-160M` varies between 0.567 and 0.705, and mean DETECTION between 0.655 and 0.688. Figure 5, however, shows an example of how individual feature scores can be highly unstable. Trajectories cross frequently, and features that rank highly on one paraphrase do not reliably rank highly on another. The distinction matters because downstream applications, such as feature steering, circuit analysis, and manual inspection, often depend on selecting specific features rather than on reading population averages. Figure 3 and Table 16 quantify ranking stability via Jaccard lift, but the interpretation differs by source of variation. We expand on this below.

Draw, Explainer, and Paraphrase. For these three sources, lift should be high, since DRAW, EXPLAINER, and PARAPHRASE are arbitrary pipeline choices that should not affect which features are identified as interpretable. Low lift indicates that feature selection is a lottery rather than a stable property of the evaluation. Under DRAW variation, SIMULATION, DETECTION, and PURITY lift at $k = 10$ and $k = 20$ are near or below chance on both models, meaning that the ten or twenty highest-scoring features on one random draw have no reliable overlap with the top on another draw of the same corpus. PARAPHRASE consistently shows the highest lift of any source (SIMULATION and DETECTION reach $43.87\times$ at $k = 10$ on `Apertus-8B`), confirming that surface wording does not substantially reshuffle feature rankings for most metrics. EXPLAINER lift is above chance for all metrics on both models, consistent with the ICC findings in 3. Interestingly, PURITY has much higher lift on $k = 10$ `Pythia-160M` (20.83) compared to `Apertus-8B` (3.82).

Corpus. The interpretation of corpus lift requires more care. Low lift under CORPUS variation does not simply indicate unreliability; a feature that activates specifically on `Pile`-like text but not on `GitHub` code may be domain-specific rather than unreliably scored. Under this interpretation, corpus-dependent feature rankings reflect genuine

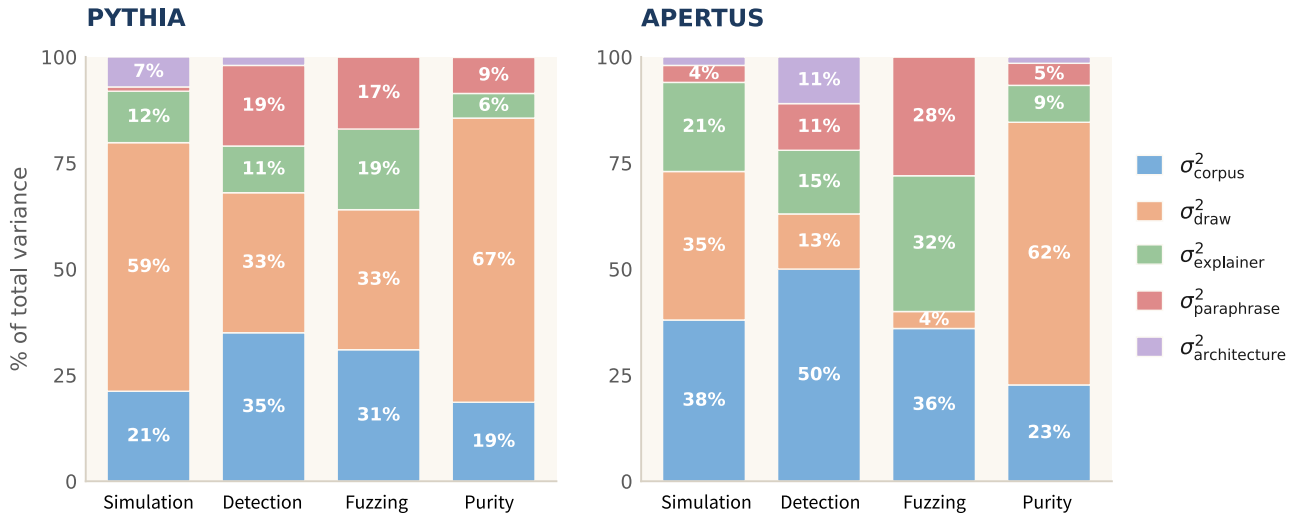

Figure 2. Variance decomposition across five experiments for `Pythia-160M` and `Apertus-8B`, with CORPUS and DRAW showing high dominance across most metrics dominates. σ^2_{arch} remains negligible across all metrics. The full table of percentages can be found in subsection E.2.

properties of the features rather than measurement noise. However, domain specificity does not explain the instability observed across semantically similar corpora. `Pile` and `C4` are both general web text, yet yield a mean SIMULATION high mean delta (9) and below-chance Jaccard lift at $k = 10$ of $0.68\times$ on both models. This instability for domain-independent corpora is difficult to explain through feature specialization. It instead points to corpus sampling as a source of measurement noise that affects which activating examples are retrieved, and thereby which features appear interpretable.

CORPUS variation also determines which features can be evaluated at all. The number of `Pythia-160M` features with at least one activating example ranges from 258 (`Pile`) to 215 (`Wikipedia`). When a hard requirement of exactly ten activating examples is applied, the number of features is reduced by 51% on `Apertus-8B` (Table 7).

4. Towards Reliable Evaluation

We propose two complementary tools for the evaluation pipeline: a *Stability Check* to assess the reliability of the evaluation before interpreting results, and a *Minimum Reporting Checklist* to enable reproducible cross-paper comparisons.

4.1. Stability Check

We propose a *Stability Check*, a test that researchers can run before interpreting autointerpretability scores as evidence of architectural quality, that goes beyond mean scores (section 3, **R3**). The check quantifies the reliability of the evaluation by varying a methodological factor across K conditions. We recommend varying *random draws* as the default, as this is the lowest-cost source of replication and our results show it often shows the highest instability (section 3, **R1**, **R2**). A simpler check of semantic similarity does not work, as cosine similarity and BERTScore remain high even when scores are unstable (Zhang* et al., 2020), making direct stable measurements necessary (subsection E.1).

Table 3. **Stability Check thresholds.** ICC interpretation follows (Koo & Li, 2016); Jaccard lift threshold of $2\times$ indicates feature selection above chance (Kuncheva, 2007).

Level	Measure	Pass	Interpretation if failed
Metric	Mean ICC	≥ 0.5	Scores unreliable across conditions
Feature	Top-k Jaccard lift	$\geq 2\times$	Feature selection at or near chance

How to run. For a chosen factor (such as DRAW, CORPUS, or EXPLAINER), generate K independent evaluation runs varying only that factor. Then compute:

- **Mean ICC** across features: if ICC < 0.50, scores have poor reliability at the metric level. Results should be interpreted with caution and ideally replicated under additional conditions.
- **Top-k Jaccard lift** across condition pairs: if lift is near $1\times$ (chance level), the top-k feature set is unlikely to replicate under a different condition, and feature selection results have to be interpreted with caution.

How to interpret. Table 3 summarises the thresholds. A metric that fails either threshold should not be used or used with caution to make cross-paper claims without method-

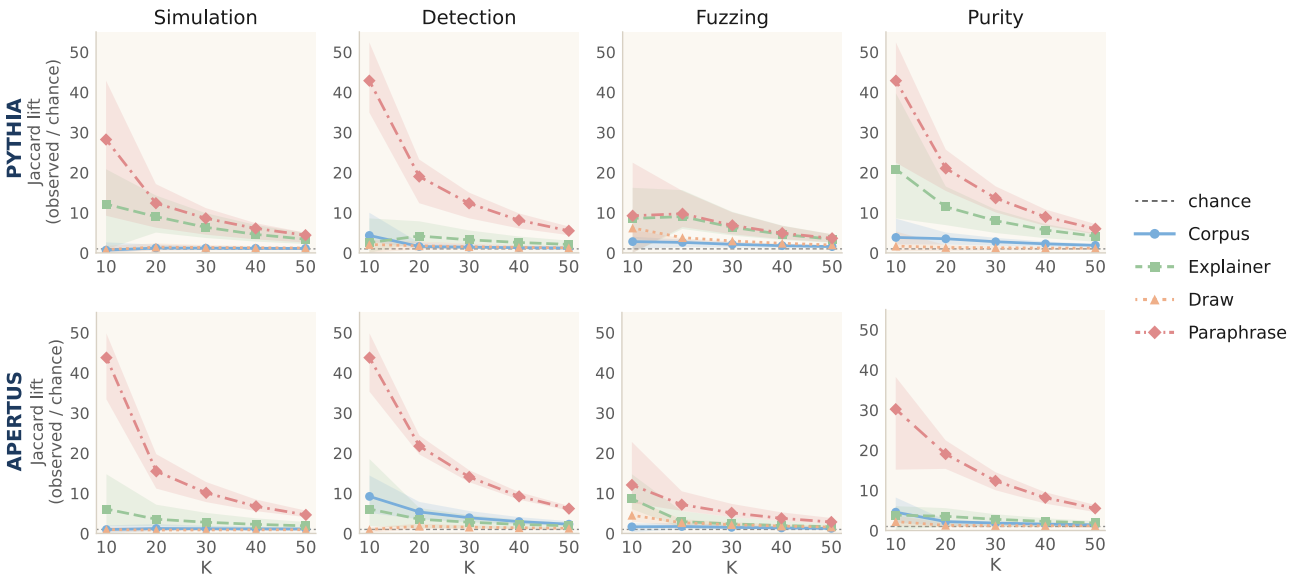

Figure 3. The Jaccard lift for all four metrics across all conditions. PARAPHRASE, which is the red line, shows the highest lift values across all conditions for all metrics.

ological standardization. Passing both thresholds is a minimum condition for reliable comparison, not a guarantee of validity. ICC thresholds should be interpreted in context (Liljequist et al., 2019): although ICC ≥ 0.75 is the standard threshold for good reliability, note that lower values can be acceptable depending on the application and the broader pattern of reported ICC values across conditions. Not all metrics fail in the same way (R2). Moreover, the variance decomposition (Equation 2.1) can be applied if multiple methodological factors are varied simultaneously, but at a higher computational cost scaling with $N_{\text{setup}} \times N_{\text{conditions}} \times N_{\text{features}}$.

4.2. Minimum Reporting Checklist

To enable cross-paper comparison, we propose a *Minimum Reporting Checklist* for autointerpretability evaluation (Table 4). Papers that omit these details cannot be reliably compared to prior work, as our findings demonstrate that each omitted factor is a potential source of substantial score variance (R1).

Each field in the checklist corresponds directly to a source of variation studied in this paper. Reporting the CORPUS and DRAWS addresses the two most harmful sources of instability identified in our experiments (R2, R3). Reporting the EXPLAINER enables replication across experiments using different models. Reporting at least one stability metric, along with the evaluation settings and results, provides a more complete picture for reproducible evaluation.

5. Conclusion

Autointerpretability scores are the dominant means of assessing SAE feature interpretability, and cross-paper comparisons of SAE architectures rely on the reliability of these scores. We asked whether *current autointerpretability evaluation practices are reliable enough to support cross-paper comparisons of SAEs, or do pipeline choices determine the outcome?* To answer this, we systematically varied four axes of the evaluation pipeline, across two models (`Pythia-160M` and `Apertus-8B`) and four metrics, measuring instability via ICC, Top-k Jaccard lift, and REML-based variance decomposition.

Our findings are summarised in threefold. **(R1)** Methodological variance collectively exceeds architectural variance in all metrics and models, with pipeline choices accounting for more score variance than SAE architectures. **(R2)** Each metric has a distinct instability profile, with DETECTION being most stable overall, while FUZZING is unreliable across all sources of variation. **(R3)** Mean scores mask per-feature instability, as top-k feature rankings have near-chance overlap across corpus and draw conditions, with consequences for applications that depend on selecting specific features. Additional results show that explanation similarity cannot diagnose score instability, as similarity scores remain high even when autointerpretability scores vary, ruling out this cheap diagnostic.

Several limitations bound the scope of our conclusions. Our experiments use a fixed scorer model (`Gemini Flash`) and a fixed layer (layer 8 for `Pythia-160M`, layer 16 for `Apertus-8B`); scorer variation and layer-dependence are

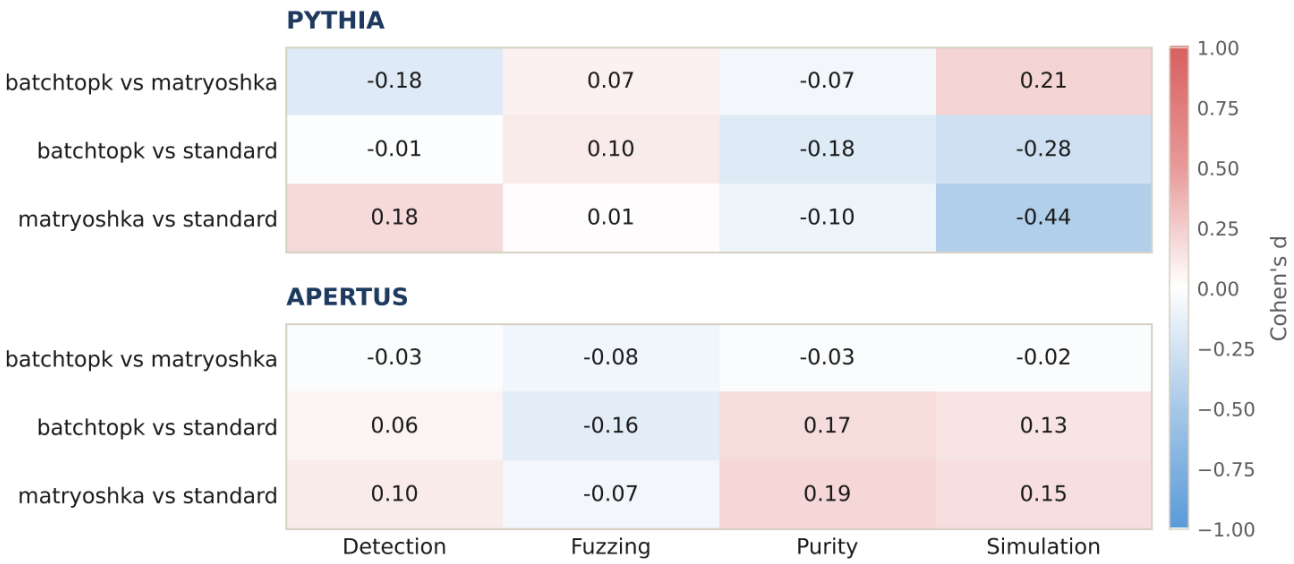

Figure 4. Cohen's d effect sizes between SAE architecture pairs on `Pythia-160M` (top) and `Apertus-8B` (bottom). We interpret Cohen's d effect sizes as follows: $d = 0.2$ indicates a small effect, $d = 0.5$ a medium effect, and $d = 0.8$ a large effect (Sullivan & Feinn, 2012). Most effects are small to negligible and inconsistent in direction across metrics.

Table 4. **Minimum Reporting Checklist**, where each field corresponds to a source of variation identified in our experiments.

Field	What to report
CORPUS	Name, size, and domain characteristics
SAMPLING	Number of activating examples and selection method (e.g., top-10 from a pool of 100)
DRAWS	Number of independent draws; if only one draw is used, acknowledge as a limitation
EXPLAINER	Model name and version
STABILITY	At least one stability metric: ICC and/or top-k Jaccard lift

additional axes of instability we did not study and that can influence scores (Puri et al., 2025). Feature samples are limited to 100–267 per condition, and variance estimates from the decomposition should be interpreted as approximate, given the partially crossed experimental design. Additionally, including more metrics can provide a more robust evaluation (Paulo et al., 2025; Puri et al., 2025). Finally, different numbers of corpora, draws, explainers, and paraphrases could have been tested to provide a richer picture of the influence of these choices. Under the specific pipeline configurations tested, the methodological variance component exceeds the architectural variance component. We do not claim this generalizes to all scorers, layers, or SAE architectures.

These findings have a direct bearing on the ongoing research on SAE utility (Kantamneni et al., 2025). To support more reliable evaluation, we provide a *Stability Check* and a *Minimum Reporting Checklist* (section 4), allowing researchers to diagnose evaluation reliability before interpreting score differences as architectural effects, and to report enough methodological detail to make cross-paper comparison reproducible.

Future work could examine alternative evaluations that do not rely solely on LLM-judged explanations. Causal and interventionist evaluations (Makelov et al., 2025; Marks et al., 2025) validate features through their downstream effects on model behavior. Similar lines of work use SAE features for steering interventions (Chalnev et al., 2024), whose effects can be measured directly on model outputs rather than mediated by an explanation step. On the other hand, the discussion about the usefulness of SAEs (Li et al., 2026; Kantamneni et al., 2025; Wu et al., 2025) underscores the need for evaluations with established reliability. We view causal, interventionist, and explanation-based evaluations as complementary: explanation-based scoring scales to millions of features in ways that causal methods currently do not, while interventionist evaluations provide the behavioral grounding that explanation quality alone cannot establish. Establishing the reliability of each approach and identifying the conditions under which they agree are important directions for future work.

As we see it, the goal is not to abandon autointerpretability, as it currently remains the most scalable approach to feature-

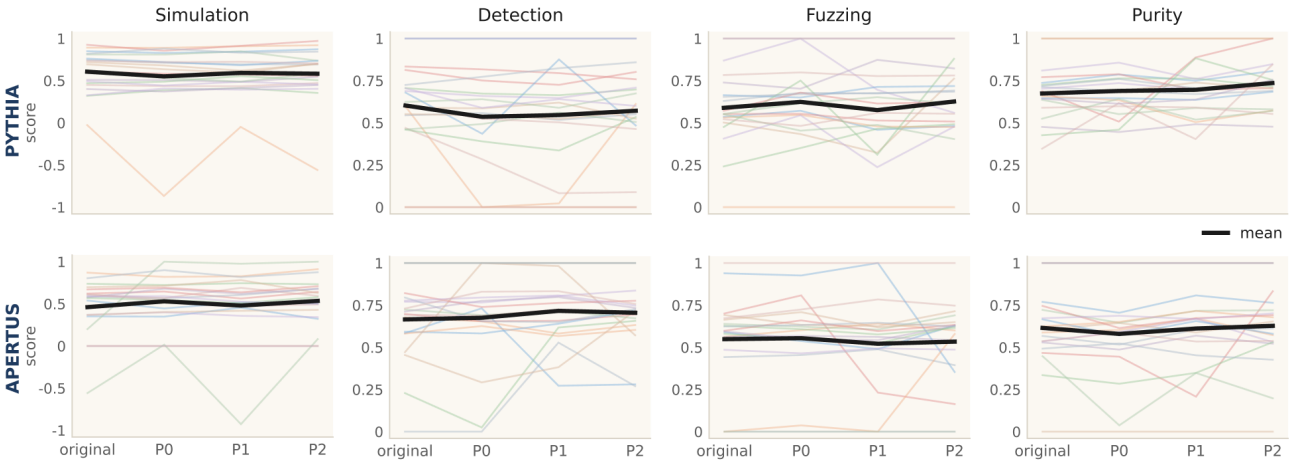

Figure 5. Metric score trajectories of 20 features across three paraphrases. SIMULATION appears the least volatile, consistent with the ICC values in Table 2.

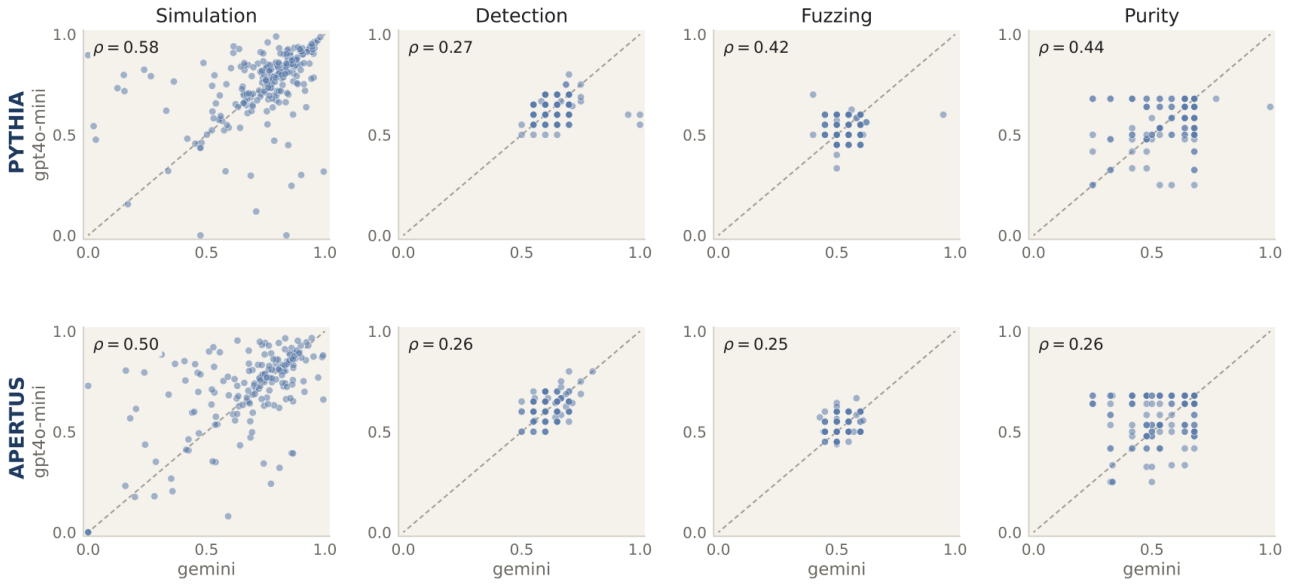

Figure 6. Per-feature metric scores for `Gemini Flash` vs.`GPT4o-mini`; each point represents one feature.

level interpretation. The goal is to use it with the caution its instability warrants, and to build the shared reporting standards that would allow the field to accumulate reliable evidence.

6. Acknowledgements

We would like to thank Raphaël Baur and Steffen Holter for their valuable feedback on the final drafts of the paper. Anna Hedström is supported by the ETH AI Center Fellowship. This work was supported under project ID a127 as part of the Swiss AI Initiative, through a grant from the ETH Domain and computational resources provided by the Swiss National Supercomputing Centre (CSCS) under the Alps infrastructure.

References

Biderman, S., Schoelkopf, H., Anthony, Q., Bradley, H., O'Brien, K., Hallahan, E., Khan, M. A., Purohit, S., Prashanth, U. S., Raff, E., et al. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, pp. 2397–2430, 2023.

Bills, S., Cammarata, N., Mossing, D., Tillman, H., Gao, L., Goh, G., Sutskever, I., Leike, J., Wu, J., and Saunders, W. Language models can explain neurons in language models, 2023. URL https:

//openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A., Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Bussmann, B., Nabeshima, N., Karvonen, A., and Nanda, N. Learning multi-level features with matryoshka sparse autoencoders. In *Proceedings of the Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=m25T5rAy43.

Chalnev, S., Siu, M., and Conmy, A. Improving steering vectors by targeting sparse autoencoder features. *arXiv preprint arXiv:2411.02193*, 2024.

Chanin, D. and Garriga-Alonso, A. Sparse but wrong: Incorrect L0 leads to incorrect features in sparse autoencoders. *arXiv preprint arXiv:2508.16560*, 2025. URL https://arxiv.org/abs/2508.16560.

Chanin, D., Wilken-Smith, J., Dulka, T., Bhatnagar, H., and Bloom, J. A is for absorption: Studying feature splitting and absorption in sparse autoencoders, 2024. URL https://arxiv.org/abs/2409.14507.

CodeParrot. GitHub Code dataset. Hugging Face Datasets, 2022. URL https://huggingface.co/datasets/codeparrot/github-code. Accessed: March 2025.

Cunningham, H., Huben, R., Smith, L. R., Ewart, A., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=F76bwRSLeK.

Foundation, W. Wikimedia downloads. URL https://dumps.wikimedia.org.

Gao, L., Biderman, S., Black, S., Golding, L., Hoppe, T., Foster, C., Phang, J., He, H., Thite, A., Nabeshima, N., Presser, S., and Leahy, C. The pile: An 800gb dataset of diverse text for language modeling, 2020. URL https://arxiv.org/abs/2101.00027.

Gao, L., la Tour, T. D., Tillman, H., Goh, G., Troll, R., Radford, A., Sutskever, I., Leike, J., and Wu, J. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=tcsZt9ZNKD.

Hernández-Cano, A., Hägele, A., Huang, A. H., Romanou, A., Solergibert, A.-J., Pasztor, B., Messmer, B., Garbaya, D., Ďurech, E. F., Hakimi, I., García Giraldo, J., Ismayilzada, M., Foroutan, N., Moalla, S., Chen, T., Sabolčec, V., Xu, Y., Aerni, M., AlKhamissi, B., Altemir Marinas, I., Amani, M. H., Ansaripour, M., Badanin, I., Benoit, H., Boros, E., Browning, N., Bösch, F., Böther, M., Canova, N., Challier, C., Charmillot, C., Coles, J., Deriu, J., Devos, A., Drescher, L., Dzenhaliou, D., Ehrmann, M., Fan, D., Fan, S., Gao, S., Gila, M., Grandury, M., Hashemi, D., Hoyle, A., Jiang, J., Klein, M., Kucharavy, A., Kucherenko, A., Lübeck, F., Machacek, R., Manitaras, T., Marfurt, A., Matoba, K., Matrenok, S., Mendonça, H., Mohamed, F. R., Montariol, S., Mouchel, L., Najem-Meyer, S., Ni, J., Oliva, G., Pagliardini, M., Palme, E., Panferov, A., Paoletti, L., Passerini, M., Pavlov, I., Poiroux, A., Ponkshe, K., Ranchin, N., Rando, J., Sauser, M., Saydaliev, J., Sayfiddinov, M. A., Schneider, M., Schuppli, S., Scialanga, M., Semenov, A., Shridhar, K., Singhal, R., Sotnikova, A., Sternfeld, A., Tarun, A. K., Teiletche, P., Vamvas, J., Yao, X., Zhao, H., Ilic, A., Klimovic, A., Krause, A., Gulcehre, C., Rosenthal, D., Ash, E., Tramèr, F., VandeVondele, J., Veraldi, L., Rajman, M., Schulthess, T., Hoefler, T., Bosselut, A., Jaggi, M., and Schlag, I. Apertus: Democratizing open and compliant LLMs for global language environments, 2025. URL https://arxiv.org/abs/2509.14233.

Juang, C., Paulo, G., Drori, J., and Belrose, N. Open source automated interpretability for sparse autoencoder features. EleutherAI Blog, July 2024.

Kantamneni, S., Engels, J., Rajamanoharan, S., Tegmark, M., and Nanda, N. Are sparse autoencoders useful? a case study in sparse probing. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=rNfzT8YkgO.

Karvonen, A., Rager, C., Lin, J., Tigges, C., Bloom, J. I., Chanin, D., Lau, Y.-T., Farrell, E., McDougall, C. S., Ayonrinde, K., Till, D., Wearden, M., Conmy, A., Marks, S., and Nanda, N. SAEBench: A comprehensive benchmark for sparse autoencoders in language model interpretability. In *Proceedings of the Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=qrU3yNfX0d.

Koo, T. K. and Li, M. Y. A guideline of selecting and reporting intraclass correlation coefficients for reliability research. *Journal of chiropractic medicine*, 15(2):155–163, 2016.

Korznikov, A., Galichin, A., Dontsov, A., Rogov, O., Oseledets, I., and Tutubalina, E. Sanity checks for sparse autoencoders: Do saes beat random baselines? *arXiv preprint arXiv:2602.14111*, 2026.

Kuncheva, L. I. A stability index for feature selection. In *Artificial intelligence and applications*, pp. 421–427, 2007.

Li, A. J., Srinivas, S., Bhalla, U., and Lakkaraju, H. Evaluating adversarial robustness of concept representations in sparse autoencoders. In Demberg, V., Inui, K., and Marquez, L. (eds.), *Proceedings of the 19th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 5940–5957, Rabat, Morocco, March 2026. Association for Computational Linguistics. ISBN 979-8-89176-380-7. doi: 10.18653/v1/2026.eacl-long.279. URL https://aclanthology.org/2026.eacl-long.279/.

Liljequist, D., Elfving, B., and Skavberg Roaldsen, K. Intraclass correlation–a discussion and demonstration of basic features. *PloS one*, 14(7):e0219854, 2019.

Makelov, A., Lange, G., and Nanda, N. Towards principled evaluations of sparse autoencoders for interpretability and control. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=1Njl73JKjB.

Marks, S., Rager, C., Michaud, E. J., Belinkov, Y., Bau, D., and Mueller, A. Sparse feature circuits: Discovering and editing interpretable causal graphs in language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=I4e82CIDxv.

Minegishi, G., Furuta, H., Iwasawa, Y., and Matsuo, Y. Rethinking evaluation of sparse autoencoders through the representation of polysemous words. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=HpUs2EXjOl.

OpenAI, :, Hurst, A., Lerer, A., Goucher, A. P., Perelman, A., Ramesh, A., Clark, A., Ostrow, A., Welihinda, A., Hayes, A., Radford, A., Mądry, A., Baker-Whitcomb, A., Beutel, A., Borzunov, A., Carney, A., Chow, A., Kirillov, A., Nichol, A., Paino, A., Renzin, A., Passos, A. T., Kirillov, A., Christakis, A., Conneau, A., Kamali, A., Jabri, A., Moyer, A., Tam, A., Crookes, A., Tootoochian, A., Tootoonchian, A., Kumar, A., Vallone, A., Karpathy, A., Braunstein, A., Cann, A., Codispoti, A., Galu, A., Kondrich, A., Tulloch, A., Mishchenko, A., Baek, A., Jiang, A., Pelisse, A., Woodford, A., Gosalia, A., Dhar, A., Pantuliano, A., Nayak, A., Oliver, A., Zoph, B., Ghorbani, B., Leimberger, B., Rossen, B., Sokolowsky, B., Wang, B., Zweig, B., Hoover, B., Samic, B., McGrew, B., Spero, B., Giertler, B., Cheng, B., Lightcap, B., Walkin, B., Quinn, B., Guarraci, B., Hsu, B., Kellogg, B., Eastman, B., Lugaresi, C., Wainwright, C., Bassin, C., Hudson, C., Chu, C., Nelson, C., Li, C., Shern, C. J., Conger, C., Barette, C., Voss, C., Ding, C., Lu, C., Zhang, C., Beaumont, C., Hallacy, C., Koch, C., Gibson, C., Kim, C., Choi, C., McLeavey, C., Hesse, C., Fischer, C., Winter, C., Czarnecki, C., Jarvis, C., Wei, C., Koumouzelis, C., Sherburn, D., Kappler, D., Levin, D., Levy, D., Carr, D., Farhi, D., Mely, D., Robinson, D., Sasaki, D., Jin, D., Valladares, D., Tsipras, D., Li, D., Nguyen, D. P., Findlay, D., Oiwoh, E., Wong, E., Asdar, E., Proehl, E., Yang, E., Antonow, E., Kramer, E., Peterson, E., Sigler, E., Wallace, E., Brevdo, E., Mays, E., Khorasani, F., Such, F. P., Raso, F., Zhang, F., von Lohmann, F., Sulit, F., Goh, G., Oden, G., Salmon, G., Starace, G., Brockman, G., Salman, H., Bao, H., Hu, H., Wong, H., Wang, H., Schmidt, H., Whitney, H., Jun, H., Kirchner, H., de Oliveira Pinto, H. P., Ren, H., Chang, H., Chung, H. W., Kivlichan, I., O'Connell, I., O'Connell, I., Osband, I., Silber, I., Sohl, I., Okuyucu, I., Lan, I., Kostrikov, I., Sutskever, I., Kanitscheider, I., Gulrajani, I., Coxon, J., Menick, J., Pachocki, J., Aung, J., Betker, J., Crooks, J., Lennon, J., Kiros, J., Leike, J., Park, J., Kwon, J., Phang, J., Teplitz, J., Wei, J., Wolfe, J., Chen, J., Harris, J., Varavva, J., Lee, J. G., Shieh, J., Lin, J., Yu, J., Weng, J., Tang, J., Yu, J., Jang, J., Candela, J. Q., Beutler, J., Landers, J., Parish, J., Heidecke, J., Schulman, J., Lachman, J., McKay, J., Uesato, J., Ward, J., Kim, J. W., Huizinga, J., Sitkin, J., Kraaijeveld, J., Gross, J., Kaplan, J., Snyder, J., Achiam, J., Jiao, J., Lee, J., Zhuang, J., Harriman, J., Fricke, K., Hayashi, K., Singhal, K., Shi, K., Karthik, K., Wood, K., Rimbach, K., Hsu, K., Nguyen, K., Gu-Lemberg, K., Button, K., Liu, K., Howe, K., Muthukumar, K., Luther, K., Ahmad, L., Kai, L., Itow, L., Workman, L., Pathak, L., Chen, L., Jing, L., Guy, L., Fedus, L., Zhou, L., Mamitsuka, L., Weng, L., McCallum, L., Held, L., Ouyang, L., Feuvrier, L., Zhang, L., Kondraciuk, L., Kaiser, L., Hewitt, L., Metz, L., Doshi, L., Aflak, M., Simens, M., Boyd, M., Thompson, M., Dukhan, M., Chen, M., Gray, M., Hudnall, M., Zhang, M., Aljubeh, M., Litwin, M., Zeng, M., Johnson, M., Shetty, M., Gupta, M., Shah, M., Yatbaz, M., Yang, M. J., Zhong, M., Glaese, M., Chen, M., Janner, M., Lampe, M., Petrov, M., Wu, M., Wang, M., Fradin, M., Pokrass, M., Castro, M., de Castro, M. O. T., Pavlov, M., Brundage, M., Wang, M., Khan, M., Murati, M., Bavarian, M., Lin, M., Yesildal, M., Soto, N., Gimelshein, N., Cone, N., Staudacher, N., Summers, N., LaFontaine, N., Chowdhury, N., Ryder, N., Stathas, N., Turley, N., Tezak, N., Felix, N., Kudige, N., Keskar, N., Deutsch, N., Bundick, N., Puckett, N., Nachum, O., Okelola, O., Boiko, O., Murk, O., Jaffe, O., Watkins, O., Godement, O., Campbell-Moore, O., Chao, P., McMillan, P., Belov, P., Su, P., Bak, P., Bakkum, P., Deng, P., Dolan, P., Hoeschele, P., Welinder, P., Tillet,

P., Pronin, P., Tillet, P., Dhariwal, P., Yuan, Q., Dias, R., Lim, R., Arora, R., Troll, R., Lin, R., Lopes, R. G., Puri, R., Miyara, R., Leike, R., Gaubert, R., Zamani, R., Wang, R., Donnelly, R., Honsby, R., Smith, R., Sahai, R., Ramchandani, R., Huet, R., Carmichael, R., Zellers, R., Chen, R., Chen, R., Nigmatullin, R., Cheu, R., Jain, S., Altman, S., Schoenholz, S., Toizer, S., Miserendino, S., Agarwal, S., Culver, S., Ethersmith, S., Gray, S., Grove, S., Metzger, S., Hermani, S., Jain, S., Zhao, S., Wu, S., Jomoto, S., Wu, S., Shuaiqi, Xia, Phene, S., Papay, S., Narayanan, S., Coffey, S., Lee, S., Hall, S., Balaji, S., Broda, T., Stramer, T., Xu, T., Gogineni, T., Christianson, T., Sanders, T., Patwardhan, T., Cunninghman, T., Degry, T., Dimson, T., Raoux, T., Shadwell, T., Zheng, T., Underwood, T., Markov, T., Sherbakov, T., Rubin, T., Stasi, T., Kaftan, T., Heywood, T., Peterson, T., Walters, T., Eloundou, T., Qi, V., Moeller, V., Monaco, V., Kuo, V., Fomenko, V., Chang, W., Zheng, W., Zhou, W., Manassra, W., Sheu, W., Zaremba, W., Patil, Y., Qian, Y., Kim, Y., Cheng, Y., Zhang, Y., He, Y., Zhang, Y., Jin, Y., Dai, Y., and Malkov, Y. Gpt-4o system card, 2024. URL https://arxiv.org/abs/2410.21276.

Paulo, G. and Belrose, N. Sparse autoencoders trained on the same data learn different features. In *Proceedings of the International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10010650.

Paulo, G. S., Mallen, A. T., Juang, C., and Belrose, N. Automatically interpreting millions of features in large language models. In *Proceedings of the Forty-second International Conference on Machine Learning*, 2025. URL https://icml.cc/virtual/2025/poster/45936.

Puri, B., Jain, A., Golimblevskaia, E., Kahardipraja, P., Wiegand, T., Samek, W., and Lapuschkin, S. FADE: Why bad descriptions happen to good features. In Che, W., Nabende, J., Shutova, E., and Pilehvar, M. T. (eds.), *Findings of the Association for Computational Linguistics: ACL 2025*, pp. 17138–17160, Vienna, Austria, 2025. Association for Computational Linguistics. doi: 10.18653/v1/2025.findings-acl.881. URL https://aclanthology.org/2025.findings-acl.881/.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023. URL https://arxiv.org/abs/1910.10683.

Rajamanoharan, S., Conmy, A., Smith, L., Lieberum, T., Varma, V., Kramar, J., Shah, R., and Nanda, N. Improving sparse decomposition of language model activations with gated sparse autoencoders. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=zLBlin2zvW.

Sullivan, G. M. and Feinn, R. Using effect size—or why the P value is not enough. *Journal of Graduate Medical Education*, 4(3):279–282, sep 2012. doi: 10.4300/JGME-D-12-00156.1.

Team, G., Georgiev, P., Lei, V. I., Burnell, R., Bai, L., Gulati, A., Tanzer, G., Vincent, D., Pan, Z., Wang, S., Mariooryad, S., Ding, Y., Geng, X., Alcober, F., Frostig, R., Omernick, M., Walker, L., Paduraru, C., Sorokin, C., Tacchetti, A., Gaffney, C., Daruki, S., Sercinoglu, O., Gleicher, Z., Love, J., Voigtlaender, P., Jain, R., Surita, G., Mohamed, K., Blevins, R., Ahn, J., Zhu, T., Kawintiranon, K., Firat, O., Gu, Y., Zhang, Y., Rahtz, M., Faruqui, M., Clay, N., Gilmer, J., Co-Reyes, J., Penchev, I., Zhu, R., Morioka, N., Hui, K., Haridasan, K., Campos, V., Mahdieh, M., Guo, M., Hassan, S., Kilgour, K., Vezer, A., Cheng, H.-T., de Liedekerke, R., Goyal, S., Barham, P., Strouse, D., Noury, S., Adler, J., Sundararajan, M., Vikram, S., Lepikhin, D., Paganini, M., Garcia, X., Yang, F., Valter, D., Trebacz, M., Vodrahalli, K., Asawaroengchai, C., Ring, R., Kalb, N., Soares, L. B., Brahma, S., Steiner, D., Yu, T., Mentzer, F., He, A., Gonzalez, L., Xu, B., Kaufman, R. L., Shafey, L. E., Oh, J., Hennigan, T., van den Driessche, G., Odoom, S., Lucic, M., Roelofs, B., Lall, S., Marathe, A., Chan, B., Ontanon, S., He, L., Teplyashin, D., Lai, J., Crone, P., Damoc, B., Ho, L., Riedel, S., Lenc, K., Yeh, C.-K., Chowdhery, A., Xu, Y., Kazemi, M., Amid, E., Petrushkina, A., Swersky, K., Khodaei, A., Chen, G., Larkin, C., Pinto, M., Yan, G., Badia, A. P., Patil, P., Hansen, S., Orr, D., Arnold, S. M. R., Grimstad, J., Dai, A., Douglas, S., Sinha, R., Yadav, V., Chen, X., Gribovskaya, E., Austin, J., Zhao, J., Patel, K., Komarek, P., Austin, S., Borgeaud, S., Friso, L., Goyal, A., Caine, B., Cao, K., Chung, D.-W., Lamm, M., Barth-Maron, G., Kagohara, T., Olszewska, K., Chen, M., Shivakumar, K., Agarwal, R., Godhia, H., Rajwar, R., Snaider, J., Dotiwalla, X., Liu, Y., Barua, A., Ungureanu, V., Zhang, Y., Batsaikhan, B.-O., Wirth, M., Qin, J., Danihelka, I., Doshi, T., Chadwick, M., Chen, J., Jain, S., Le, Q., Kar, A., Gurumurthy, M., Li, C., Sang, R., Liu, F., Lamprou, L., Munoz, R., Lintz, N., Mehta, H., Howard, H., Reynolds, M., Aroyo, L., Wang, Q., Blanco, L., Cassirer, A., Griffith, J., Das, D., Lee, S., Sygnowski, J., Fisher, Z., Besley, J., Powell, R., Ahmed, Z., Paulus, D., Reitter, D., Borsos, Z., Joshi, R., Pope, A., Hand, S., Selo, V., Jain, V., Sethi, N., Goel, M., Makino, T., May, R., Yang, Z., Schalkwyk, J., Butterfield, C., Hauth, A., Goldin, A., Hawkins, W., Senter, E., Brin, S., Woodman, O., Ritter, M., Noland, E., Giang, M., Bolina, V., Lee, L., Blyth, T., Mackinnon, I., Reid, M., Sarvana, O., Sil-

ver, D., Chen, A., Wang, L., Maggiore, L., Chang, O., Attaluri, N., Thornton, G., Chiu, C.-C., Bunyan, O., Levine, N., Chung, T., Eltyshev, E., Si, X., Lillicrap, T., Brady, D., Aggarwal, V., Wu, B., Xu, Y., McIlroy, R., Badola, K., Sandhu, P., Moreira, E., Stokowiec, W., Hemsley, R., Li, D., Tudor, A., Shyam, P., Rahimtoroghi, E., Haykal, S., Sprechmann, P., Zhou, X., Mincu, D., Li, Y., Addanki, R., Krishna, K., Wu, X., Frechette, A., Eyal, M., Dafoe, A., Lacey, D., Whang, J., Avrahami, T., Zhang, Y., Taropa, E., Lin, H., Toyama, D., Rutherford, E., Sano, M., Choe, H., Tomala, A., Safranek-Shrader, C., Kassner, N., Pajarskas, M., Harvey, M., Sechrist, S., Fortunato, M., Lyu, C., Elsayed, G., Kuang, C., Lottes, J., Chu, E., Jia, C., Chen, C.-W., Humphreys, P., Baumli, K., Tao, C., Samuel, R., dos Santos, C. N., Andreassen, A., Rakićević, N., Grewe, D., Kumar, A., Winkler, S., Caton, J., Brock, A., Dalmia, S., Sheahan, H., Barr, I., Miao, Y., Natsev, P., Devlin, J., Behbahani, F., Prost, F., Sun, Y., Myaskovsky, A., Pillai, T. S., Hurt, D., Lazaridou, A., Xiong, X., Zheng, C., Pardo, F., Li, X., Horgan, D., Stanton, J., Ambar, M., Xia, F., Lince, A., Wang, M., Mustafa, B., Webson, A., Lee, H., Anil, R., Wicke, M., Dozat, T., Sinha, A., Piqueras, E., Dabir, E., Upadhyay, S., Boral, A., Hendricks, L. A., Fry, C., Djolonga, J., Su, Y., Walker, J., Labanowski, J., Huang, R., Misra, V., Chen, J., Skerry-Ryan, R., Singh, A., Rijhwani, S., Yu, D., Castro-Ros, A., Changpinyo, B., Datta, R., Bagri, S., Hrafnkelsson, A. M., Maggioni, M., Zheng, D., Sulsky, Y., Hou, S., Paine, T. L., Yang, A., Riesa, J., Rogozinska, D., Marcus, D., Badawy, D. E., Zhang, Q., Wang, L., Miller, H., Greer, J., Sjos, L. L., Nova, A., Zen, H., Chaabouni, R., Rosca, M., Jiang, J., Chen, C., Liu, R., Sainath, T., Krikun, M., Polozov, A., Lespiau, J.-B., Newlan, J., Cankara, Z., Kwak, S., Xu, Y., Chen, P., Coenen, A., Meyer, C., Tsihlas, K., Ma, A., Gottweis, J., Xing, J., Gu, C., Miao, J., Frank, C., Cankara, Z., Ganapathy, S., Dasgupta, I., Hughes-Fitt, S., Chen, H., Reid, D., Rong, K., Fan, H., van Amersfoort, J., Zhuang, V., Cohen, A., Gu, S. S., Mohananey, A., Ilic, A., Tobin, T., Wieting, J., Bortsova, A., Thacker, P., Wang, E., Caveness, E., Chiu, J., Sezener, E., Kaskasoli, A., Baker, S., Millican, K., Elhawaty, M., Aisopos, K., Lebsack, C., Byrd, N., Dai, H., Jia, W., Wiethoff, M., Davoodi, E., Weston, A., Yagati, L., Ahuja, A., Gao, I., Pundak, G., Zhang, S., Azzam, M., Sim, K. C., Caelles, S., Keeling, J., Sharma, A., Swing, A., Li, Y., Liu, C., Bostock, C. G., Bansal, Y., Nado, Z., Anand, A., Lipschultz, J., Karmarkar, A., Proleev, L., Ittycheriah, A., Yeganeh, S. H., Polovets, G., Faust, A., Sun, J., Rrustemi, A., Li, P., Shivanna, R., Liu, J., Welty, C., Lebron, F., Baddepudi, A., Krause, S., Parisotto, E., Soricut, R., Xu, Z., Bloxwich, D., Johnson, M., Neyshabur, B., Mao-Jones, J., Wang, R., Ramasesh, V., Abbas, Z., Guez, A., Segal, C., Nguyen, D. D., Svensson, J., Hou, L., York, S., Milan, K., Bridgers, S., Gworek, W., Tagliasacchi, M., Lee-Thorp, J., Chang, M., Guseynov, A., Hartman, A. J., Kwong, M., Zhao, R., Kashem, S., Cole, E., Miech, A., Tanburn, R., Phuong, M., Pavetic, F., Cevey, S., Comanescu, R., Ives, R., Yang, S., Du, C., Li, B., Zhang, Z., Iinuma, M., Hu, C. H., Roy, A., Bijwadia, S., Zhu, Z., Martins, D., Saputro, R., Gergely, A., Zheng, S., Jia, D., Antonoglou, I., Sadovsky, A., Gu, S., Bi, Y., Andreev, A., Samangooei, S., Khan, M., Kocisky, T., Filos, A., Kumar, C., Bishop, C., Yu, A., Hodkinson, S., Mittal, S., Shah, P., Moufarek, A., Cheng, Y., Bloniarz, A., Lee, J., Pejman, P., Michel, P., Spencer, S., Feinberg, V., Xiong, X., Savinov, N., Smith, C., Shakeri, S., Tran, D., Chesus, M., Bohnet, B., Tucker, G., von Glehn, T., Muir, C., Mao, Y., Kazawa, H., Slone, A., Soparkar, K., Shrivastava, D., Cobon-Kerr, J., Sharman, M., Pavagadhi, J., Araya, C., Misiunas, K., Ghelani, N., Laskin, M., Barker, D., Li, Q., Briukhov, A., Houlsby, N., Glaese, M., Lakshminarayanan, B., Schucher, N., Tang, Y., Collins, E., Lim, H., Feng, F., Recasens, A., Lai, G., Magni, A., Cao, N. D., Siddhant, A., Ashwood, Z., Orbay, J., Dehghani, M., Brennan, J., He, Y., Xu, K., Gao, Y., Saroufim, C., Molloy, J., Wu, X., Arnold, S., Chang, S., Schrittwieser, J., Buchatskaya, E., Radpour, S., Polacek, M., Giordano, S., Bapna, A., Tokumine, S., Hellendoorn, V., Sottiaux, T., Cogan, S., Severyn, A., Saleh, M., Thakoor, S., Shefey, L., Qiao, S., Gaba, M., yiin Chang, S., Swanson, C., Zhang, B., Lee, B., Rubenstein, P. K., Song, G., Kwiatkowski, T., Koop, A., Kannan, A., Kao, D., Schuh, P., Stjerngren, A., Ghiasi, G., Gibson, G., Vilnis, L., Yuan, Y., Ferreira, F. T., Kamath, A., Klimenko, T., Franko, K., Xiao, K., Bhattacharya, I., Patel, M., Wang, R., Morris, A., Strudel, R., Sharma, V., Choy, P., Hashemi, S. H., Landon, J., Finkelstein, M., Jhakra, P., Frye, J., Barnes, M., Mauger, M., Daun, D., Baatarsukh, K., Tung, M., Farhan, W., Michalewski, H., Viola, F., de Chaumont Quitry, F., Lan, C. L., Hudson, T., Wang, Q., Fischer, F., Zheng, I., White, E., Dragan, A., baptiste Alayrac, J., Ni, E., Pritzel, A., Iwanicki, A., Isard, M., Bulanova, A., Zilka, L., Dyer, E., Sachan, D., Srinivasan, S., Muckenhirn, H., Cai, H., Mandhane, A., Tariq, M., Rae, J. W., Wang, G., Ayoub, K., FitzGerald, N., Zhao, Y., Han, W., Alberti, C., Garrette, D., Krishnakumar, K., Gimenez, M., Levskaya, A., Sohn, D., Matak, J., Iturrate, I., Chang, M. B., Xiang, J., Cao, Y., Ranka, N., Brown, G., Hutter, A., Mirrokni, V., Chen, N., Yao, K., Egyed, Z., Galilee, F., Liechty, T., Kallakuri, P., Palmer, E., Ghemawat, S., Liu, J., Tao, D., Thornton, C., Green, T., Jasarevic, M., Lin, S., Cotruta, V., Tan, Y.-X., Fiedel, N., Yu, H., Chi, E., Neitz, A., Heitkaemper, J., Sinha, A., Zhou, D., Sun, Y., Kaed, C., Hulse, B., Mishra, S., Georgaki, M., Kudugunta, S., Farabet, C., Shafran, I., Vlasic, D., Tsitsulin, A., Ananthanarayanan, R., Carin, A., Su, G., Sun, P., V, S., Carvajal, G., Broder, J., Comsa, I., Repina, A., Wong, W., Chen, W. W., Hawkins, P., Filonov, E., Loher, L., Hirnschall, C., Wang, W., Ye, J., Burns, A., Cate, H., Wright,

D. G., Piccinini, F., Zhang, L., Lin, C.-C., Gog, I., Kulizhskaya, Y., Sreevatsa, A., Song, S., Cobo, L. C., Iyer, A., Tekur, C., Garrido, G., Xiao, Z., Kemp, R., Zheng, H. S., Li, H., Agarwal, A., Ngani, C., Goshvadi, K., Santamaria-Fernandez, R., Fica, W., Chen, X., Gorgolewski, C., Sun, S., Garg, R., Ye, X., Eslami, S. M. A., Hua, N., Simon, J., Joshi, P., Kim, Y., Tenney, I., Potluri, S., Thiet, L. N., Yuan, Q., Luisier, F., Chronopoulou, A., Scellato, S., Srinivasan, P., Chen, M., Koverkathu, V., Dalibard, V., Xu, Y., Saeta, B., Anderson, K., Sellam, T., Fernando, N., Huot, F., Jung, J., Varadarajan, M., Quinn, M., Raul, A., Le, M., Habalov, R., Clark, J., Jalan, K., Bullard, K., Singhal, A., Luong, T., Wang, B., Rajayogam, S., Eisenschlos, J., Jia, J., Finchelstein, D., Yakubovich, A., Balle, D., Fink, M., Agarwal, S., Li, J., Dvijotham, D., Pal, S., Kang, K., Konzelmann, J., Beattie, J., Dousse, O., Wu, D., Crocker, R., Elkind, C., Jonnalagadda, S. R., Lee, J., Holtmann-Rice, D., Kallarackal, K., Liu, R., Vnukov, D., Vats, N., Invernizzi, L., Jafari, M., Zhou, H., Taylor, L., Prendki, J., Wu, M., Eccles, T., Liu, T., Kopparapu, K., Beaufays, F., Angermueller, C., Marzoca, A., Sarcar, S., Dib, H., Stanway, J., Perbet, F., Trdin, N., Sterneck, R., Khorlin, A., Li, D., Wu, X., Goenka, S., Madras, D., Goldshtein, S., Gierke, W., Zhou, T., Liu, Y., Liang, Y., White, A., Li, Y., Singh, S., Bahargam, S., Epstein, M., Basu, S., Lao, L., Ozturel, A., Crous, C., Zhai, A., Lu, H., Tung, Z., Gaur, N., Walton, A., Dixon, L., Zhang, M., Globerson, A., Uy, G., Bolt, A., Wiles, O., Nasr, M., Shumailov, I., Selvi, M., Piccinno, F., Aguilar, R., McCarthy, S., Khalman, M., Shukla, M., Galic, V., Carpenter, J., Villela, K., Zhang, H., Richardson, H., Martens, J., Bosnjak, M., Belle, S. R., Seibert, J., Alnahlawi, M., McWilliams, B., Singh, S., Louis, A., Ding, W., Popovici, D., Simicich, L., Knight, L., Mehta, P., Gupta, N., Shi, C., Fatehi, S., Mitrovic, J., Grills, A., Pagadora, J., Munkhdalai, T., Petrova, D., Eisenbud, D., Zhang, Z., Yates, D., Mittal, B., Tripuraneni, N., Assael, Y., Brovelli, T., Jain, P., Velimirovic, M., Akbulut, C., Mu, J., Macherey, W., Kumar, R., Xu, J., Qureshi, H., Comanici, G., Wiesner, J., Gong, Z., Ruddock, A., Bauer, M., Felt, N., GP, A., Arnab, A., Zelle, D., Rothfuss, J., Rosgen, B., Shenoy, A., Seybold, B., Li, X., Mudigonda, J., Erdogan, G., Xia, J., Simsa, J., Michi, A., Yao, Y., Yew, C., Kan, S., Caswell, I., Radebaugh, C., Elisseeff, A., Valenzuela, P., McKinney, K., Paterson, K., Cui, A., Latorre-Chimoto, E., Kim, S., Zeng, W., Durden, K., Ponnapalli, P., Sosea, T., Choquette-Choo, C. A., Manyika, J., Robenek, B., Vashisht, H., Pereira, S., Lam, H., Velic, M., Owusu-Afriyie, D., Lee, K., Bolukbasi, T., Parrish, A., Lu, S., Park, J., Venkatraman, B., Talbert, A., Rosique, L., Cheng, Y., Sozanschi, A., Paszke, A., Kumar, P., Austin, J., Li, L., Salama, K., Perz, B., Kim, W., Dukkipati, N., Baryshnikov, A., Kaplanis, C., Sheng, X., Chervonyi, Y., Unlu, C., de Las Casas, D., Askham, H., Tunyasuvunakool, K., Gimeno, F., Poder, S., Kwak, C., Miecnikowski, M., Mirrokni, V., Dimitriev, A., Parisi, A., Liu, D., Tsai, T., Shevlane, T., Kouridi, C., Garmon, D., Goedeckemeyer, A., Brown, A. R., Vijayakumar, A., Elqursh, A., Jazayeri, S., Huang, J., Carthy, S. M., Hoover, J., Kim, L., Kumar, S., Chen, W., Biles, C., Bingham, G., Rosen, E., Wang, L., Tan, Q., Engel, D., Pongetti, F., de Cesare, D., Hwang, D., Yu, L., Pullman, J., Narayanan, S., Levin, K., Gopal, S., Li, M., Aharoni, A., Trinh, T., Lo, J., Casagrande, N., Vij, R., Matthey, L., Ramadhana, B., Matthews, A., Carey, C., Johnson, M., Goranova, K., Shah, R., Ashraf, S., Dasgupta, K., Larsen, R., Wang, Y., Vuyyuru, M. R., Jiang, C., Ijazi, J., Osawa, K., Smith, C., Boppana, R. S., Bilal, T., Koizumi, Y., Xu, Y., Altun, Y., Shabat, N., Bariach, B., Korchemniy, A., Choo, K., Ronneberger, O., Iwuanyanwu, C., Zhao, S., Soergel, D., Hsieh, C.-J., Cai, I., Iqbal, S., Sundermeyer, M., Chen, Z., Bursztein, E., Malaviya, C., Biadsy, F., Shroff, P., Dhillon, I., Latkar, T., Dyer, C., Forbes, H., Nicosia, M., Nikolaev, V., Greene, S., Georgiev, M., Wang, P., Martin, N., Sedghi, H., Zhang, J., Banzal, P., Fritz, D., Rao, V., Wang, X., Zhang, J., Patraucean, V., Du, D., Mordatch, I., Jurin, I., Liu, L., Dubey, A., Mohan, A., Nowakowski, J., Ion, V.-D., Wei, N., Tojo, R., Raad, M. A., Hudson, D. A., Keshava, V., Agrawal, S., Ramirez, K., Wu, Z., Nguyen, H., Liu, J., Sewak, M., Petrini, B., Choi, D., Philips, I., Wang, Z., Bica, I., Garg, A., Wilkiewicz, J., Agrawal, P., Li, X., Guo, D., Xue, E., Shaik, N., Leach, A., Khan, S. M., Wiesinger, J., Jerome, S., Chakladar, A., Wang, A. W., Ornduff, T., Abu, F., Ghaffarkhah, A., Wainwright, M., Cortes, M., Liu, F., Maynez, J., Terzis, A., Samangouei, P., Mansour, R., Kępa, T., Aubet, F.-X., Algymr, A., Banica, D., Weisz, A., Orban, A., Senges, A., Andrejczuk, E., Geller, M., Santo, N. D., Anklin, V., Merey, M. A., Baeuml, M., Strohman, T., Bai, J., Petrov, S., Wu, Y., Hassabis, D., Kavukcuoglu, K., Dean, J., and Vinyals, O. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context, 2024. URL https://arxiv.org/abs/2403.05530.

Templeton, A., Conerly, T., Marcus, J., Lindsey, J., Bricken, T., Chen, B., Pearce, A., Citro, C., Ameisen, E., Jones, A., Cunningham, H., Turner, N. L., McDougall, C., MacDiarmid, M., Freeman, C. D., Sumers, T. R., Rees, E., Batson, J., Jermyn, A., Carter, S., Olah, C., and Henighan, T. Scaling monosemanticity: Extracting interpretable features from claude 3 sonnet. *Transformer Circuits Thread*, 2024. URL https://transformer-circuits.pub/2024/scaling-monosemanticity/index.html.

Wang, M., la Tour, T. D., Watkins, O., Makelov, A., Chi, R. A., Miserendino, S., Wang, J., Rajaram, A., Heidecke, J., Patwardhan, T., et al. Persona features control emergent misalignment. *arXiv preprint arXiv:2506.19823*, 2025.

Wu, Z., Arora, A., Geiger, A., Wang, Z., Huang, J., Jurafsky, D., Manning, C. D., and Potts, C. Axbench: Steering LLMs? even simple baselines outperform sparse autoencoders. In *Forty-second International Conference on Machine Learning*, 2025. URL https://openreview.net/forum?id=K2CckZjNy0.

Zhang*, T., Kishore*, V., Wu*, F., Weinberger, K. Q., and Artzi, Y. Bertscore: Evaluating text generation with bert. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeHuCVFDr.

A. Related Work

Evaluation Gap The field has recognized that improvements in the sparsity-reconstruction trade-off often do not directly translate to improved monosemanticity (Minegishi et al., 2025; Karvonen et al., 2025). The intuition behind reconstruction error as a proxy is that a low error indicates the SAE faithfully captures the original model's activations (Bricken et al., 2023; Rajamanoharan et al., 2024). Similarly, sparsity is assumed to promote interpretability by forcing the model to represent each input using few active latent units (Cunningham et al., 2024). However, SAEs can be optimized along this trade-off and still produce polysemantic, unstable, or irrelevant features (Karvonen et al., 2025; Minegishi et al., 2025). More broadly, most metrics used in practice are *proxy* metrics: reconstruction error, sparsity, and loss recovery reflect optimization behavior rather than semantic quality, and optimizing for sparsity can introduce failure modes such as feature absorption (Chanin et al., 2024) and dead latents (Gao et al., 2025).

A first step toward a more principled approach was taken by (Makelov et al., 2025), who proposed a framework of faithfulness checks and feature-editing experiments relative to a supervised baseline. The authors explicitly acknowledge that their framework depends on a particular parametrization of task-relevant variables, which makes comparisons across settings difficult. This is a limitation that reflects the broader challenge that interpretability evaluation is conditional on task, variables, and the conceptual lens. SAEBench (Karvonen et al., 2025) extends this toward standardization, comparing SAE variants across reconstruction fidelity, feature disentanglement, interpretability, and concept detection. They show that different model choices, such as dictionary size, influence evaluation metrics. A key finding is that these metrics cannot be unified into a single score, making evaluation fundamentally multi-objective: metric choice depends on the researcher's priority.

Autointerpretability Interpretability evaluation in practice relies predominantly on *automated interpretability* pipelines: a LM generates natural language explanations for features, and a scorer evaluates those explanations by testing whether they predict feature activations in new contexts. The approach was originally developed for neuron-level interpretation (Bills et al., 2023) and subsequently adapted to SAE features (Bricken et al., 2023; Cunningham et al., 2024). Recent work has extended the pipeline in several directions. (Paulo et al., 2025) scale explanation generation to millions of features, while (Puri et al., 2025) examines failure modes of automated description. Additionally, (Puri et al., 2025) shows that explanations may not faithfully capture feature behavior and that scoring procedures can reward misleading descriptions.

Known Sources of Variation Several studies have identified factors that influence autointerpretability scores, motivating the systematic analysis we present in this paper. We organize these into three groups.

- *SAE and model properties.* Interpreting larger SAEs is more difficult, with larger dictionaries yielding lower scores (Puri et al., 2025). Scores vary across layers, with later layers tending to be more polysemantic and harder to explain (Puri et al., 2025). SAEs with more features tend to score higher than individual neurons, suggesting that sparsity aids interpretability (Paulo et al., 2025). Score variation on detection across SAE architectures is documented in (Karvonen et al., 2025).
- *Example selection.* The choice of which activating examples are shown to the explainer affects explanation quality. Using only top-activating examples yields concise, specific descriptions but fails to capture the full activation distribution (Paulo et al., 2025). Random sampling yields interpretations that span a more diverse set of activating contexts, whereas stratified sampling can produce descriptions that are overly broad (Paulo et al., 2025). Including only activating contexts increases specificity but reduces sensitivity; including non-activating contexts has the opposite effect (Paulo et al., 2025). Providing a larger number of examples yields slightly higher scores overall (Paulo et al., 2025; Puri et al., 2025), though this improvement may reflect increased generality rather than increased accuracy (Paulo et al., 2025).
- *Explainer and scorer properties.* Larger explainer models produce better feature descriptions and higher scores (Paulo et al., 2025; Puri et al., 2025). For simpler metrics such as detection and fuzzing, using a smaller scorer model leads to lower scores, suggesting that scorer capacity interacts with metric complexity (Paulo et al., 2025). Prompt construction affects results (Puri et al., 2025), as does the use of few-shot prompting (Puri et al., 2025). Notably, (Paulo et al., 2025) finds that training dataset variation has little effect on scores, suggesting that corpus choice at training time is less influential than corpus choice at evaluation time, which is a distinction our experiments make explicit.

What remains absent from prior work is a systematic quantification of the *relative* contribution of these sources to total score variance, and an assessment of whether their combined effect exceeds the variance attributable to architectural differences. This paper addresses both questions, with additional focus on corpus, draws, explainers, phrasing, and architecture.

Table 5. Experimental design. Each experiment isolates one source of variation.

Experiment	Factor Varied	Variation
Corpus	Evaluation corpus	4 corpora
Sampling	Random draw of examples	5 draws
Explainer	Explainer LLM	2 explainer models
Phrasing	Explanation wording	3 paraphrases
Architecture	Architecture influence	3 architectures

B. Experiments

The following sections contain additional information to help with reproducibility of the experiments.

B.1. Experiment Overview

An overview of the number of variations for each condition can be found in Table 5.

Exp. 1 — Corpus dependence. We generated independent explanations and scores for the same set of features on four different corpora spanning a gradient of distributional distance from the training data: the Pile (in-distribution), C4 (similar), Wikipedia (moderate), and GitHub code (dissimilar). All datsets are listed in Table 6. This tests whether autointerpretability scores reflect stable feature properties or are confounded by the choice of evaluation corpus. It is expected that more similar corpora have similar scores on the same features.

Exp. 2a — Sampling instability. For a fixed corpus, we generated multiple explanations per feature with the same explainer by drawing different random subsets of activating examples. This isolates the variance introduced by which examples happen to be sampled, holding all other factors constant. We use five independent draws per feature to balance statistical and computational considerations.

Exp. 2b — Explainer model dependence. We explained the same features using two different explainer LLMs: `Gemini Flash` and `GPT4o-mini`. All explanations were scored with a fixed, but different model. Both are small, efficient models positioned below the frontier tier of their respective families, and are broadly comparable in general capability. As other papers have looked into the effect of the size of the explainers ((Paulo et al., 2025; Puri et al., 2025), this is not the main focus. As the chosen models are comparable in size, we investigated the effects of their differences, which should be minimal in theory. We restricted our comparison to two closed-source models representing the most common practical choice; whether instability patterns generalize to open-source explainers remains an open question.

Exp. 2c — Paraphrase sensitivity. To test whether scoring measures semantics or primarily surface wording, we generated three paraphrases of each explanation and scored all versions on the same held-out sequences. If scores vary substantially across paraphrases, this indicates that the scorer is responding to wording rather than meaning. We generate three paraphrases per explanation, again balancing statistical and computational considerations. Critically, paraphrase represents the most semantically conservative perturbation in our experimental design, as it focuses solely on the explanation itself, not on the input to or generation of the explanation.

Exp. 3 — Cross-architecture comparison. To contextualize the methodological instability, we compared score distributions across three SAE architectures trained on the same model and data: a standard ReLU SAE, a Matryoshka SAE, and a BatchTopK SAE. All other methodological choices were fixed. This provides a reference point: if methodological variance (Exp. 1–2) exceeds architectural variance, this could limit the validity of cross-SAE comparisons in the literature. We investigate SAEs trained on smaller and larger models (Pythia160M vs. Apertus-8b Instruct). Earlier work has shown that architectures can have a different autointerpretability score (Karvonen et al., 2025), but we extend this to multiple metrics.

B.2. Corpora

All corpora are sampled to 10,000 sequences, tokenized with the model-specific tokenizer (Pythia tokenizer for Pythia-160M; Apertus tokenizer for Apertus-8B), and truncated to 32 tokens following the findings of (Paulo et al., 2025). All

Table 6. Corpus statistics. All corpora sampled to 10,000 sequences of 32 tokens.

Corpus	Domain	HuggingFace identifier	License	Experiments
The Pile	General web text (mixed)	`monology/pile-uncopyrighted`	Other	All
C4	Cleaned web text	`allenai/c4`	ODC-BY	Exp. 1
Wikipedia	Encyclopedic text	`wikimedia/wikipedia`	CC-BY-SA-3.0	Exp. 1
GitHub Code	Source code	`codeparrot/github-code`	Other	Exp. 1

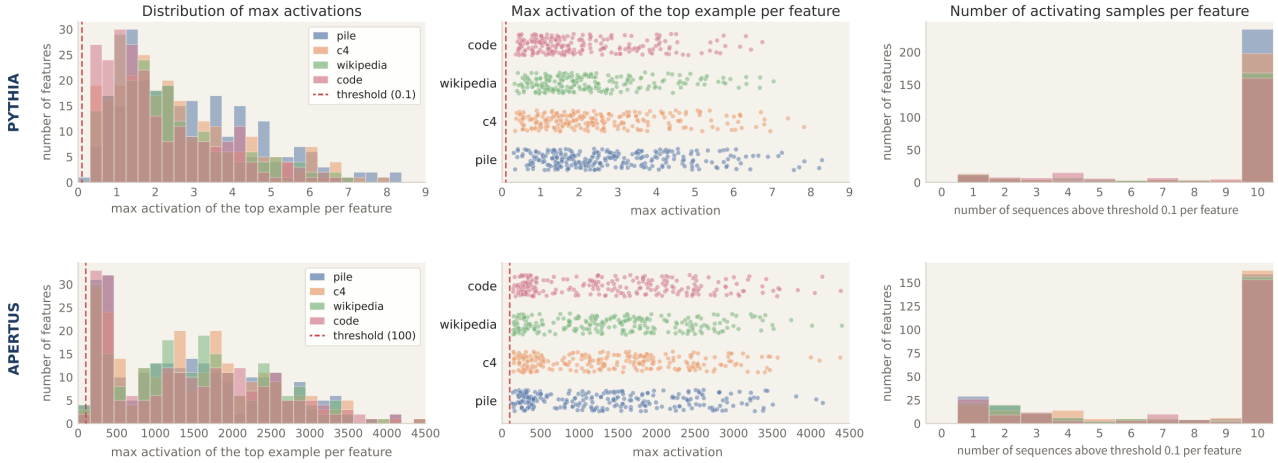

Figure 7. Activation diagnostics for features, considering all four datasets. The first column of figures shows the distribution of the highest-activating sample for each feature. The second shows the spread of features between the minimum activation threshold and the maximum activation detected for a feature. The final one shows how many activating samples there are per feature.

datasets were accessed through HuggingFace. Table 6 summarises corpus statistics.

The four corpora were chosen to span different degrees of similarity to the SAE training distribution, from similar to dissimilar. Both SAEs are trained on the Pile.

B.3. Feature Sampling

Features are selected at random from the full SAE feature pool per experiment, targeting to retrieve 250 active features per condition. To reach this number, we randomly sample 300 features for Pythia and 2000 features for Apertus. A feature is considered *active* if its maximum activation over the evaluation corpus exceeds a model-specific threshold: 0.1 for Pythia-160M and 100 for Apertus-8B, reflecting the difference in activation scales between the two models. Figure 7 shows that these thresholds yield sufficient activating sequences.

For each active feature, the 10 highest-activating sentences are extracted for explanation generation. Detection scoring uses 10 randomly sampled negative sentences (where the feature does not activate) per feature. We have opted for the standard 10 highest-activating sequences, as prior work has already shown their influence, and we have chosen a number that is feasible regarding computing costs and efficiency (Paulo et al., 2025; Puri et al., 2025). However, the requirement of at least one activating sample per feature can yield explanations with fewer than 10 samples. An example of this is shown in Table 7 for the CORPUS condition.

B.3.1. Feature Evaluation and Activation Coverage

There are differences in the number of features that can be evaluated per methodological condition, which is shown in for the CORPUS condition in Table 7. This also introduces a small selection bias, as the choice of corpus determines which features can be scored.

Table 7. Number of features **with at least one** activating example per corpus vs. **with 10 activating samples per corpus**. *Overlap* denotes features activating in all four corpora simultaneously and constitutes the evaluation set for Exp. 1. *Distinct* indicates the number of different features in total

		Corpus					
Model	**Threshold**	**Pile**	**C4**	**Wiki.**	**Code**	**Overlap**	**Distinct**
Pythia-160M	≥ 1	258	236	215	223	190	258
	10	143	134	131	133	121	143
Apertus-8B	≥ 1	244	244	234	221	189	264
	10	130	130	128	128	128	130

B.3.2. Draw Sampling (Exp. 2a)

For Exp. 2b, each of the five draws samples 10 sequences uniformly at random from a pool of 100 top-activating sequences, yielding an independent explanation and score profile for each draw. This experiment is close to the experiments on top-activating and stratified sampling reported in (Paulo et al., 2025).

B.4. Explainer Prompt

System prompt — Explanation generation

```
You are a neuron interpreter for a sparse autoencoder (SAE)
trained on a language model.
Your task: Given examples of text where a specific SAE feature
activates strongly, describe what concept, pattern, or behavior
the feature detects.
Guidelines:
- Be specific and concise (1-2 sentences)
- Focus on the common pattern across all examples
- Mention token-level patterns if relevant
- If the pattern is unclear, say so honestly
- Describe the concept directly (e.g. "Mathematical equations
  and formulas" or "Python source code and programming
  constructs"). Do NOT start with "This feature activates
  for/when/on" or similar phrases.
```

User prompt — Explanation generation

```
Here are text examples where SAE feature {feature_id} activates
strongly. Each example shows the text and the activation
strength.
{examples}
Based on these examples, describe what concept or pattern
feature {feature_id} detects. {word_limit_instruction}starting
directly with the concept (not with "This feature
activates...").
```

where `{word_limit_instruction}` resolves to:

- *Default:* "Answer in 1–2 sentences, "
- *W10 condition:* "Answer in at most 10 words, "

B.5. Explainers (Exp. 2b)

Exp. 2b compares two explainer LLMs: `Gemini Flash` (primary) and `GPT4o-mini`(comparison). Both models receive identical inputs, which are the same ten highest-activating sequences with activating tokens highlighted, and are prompted to generate a natural language description of the feature. All scoring in Exp. 2b uses `Gemini Flash` as the fixed scorer, isolating the effect of explainer model identity from scorer variation.

B.6. Paraphrase Generation (Exp. 2c)

Three paraphrases are generated per explanation. The paraphrase prompt instructs the model to preserve the semantic meaning of the original explanation while varying the surface wording, sentence structure and vocabulary. Each original explanation and its three paraphrases are scored indepdently on the same held-out sequences, producing four observations per feature per metric.

Paraphrases are generated by the same LLM as the scorer, which is Gemini Flash. This could mean that our paraphrase stability result may overstate semantic invariance because both the paraphraser and scorer share representational biases. A future test would use paraphrases from a structurally different model, which we view this as an important follow-up.

System prompt

```
You are a precise language model assistant. Your task is to
rephrase a given feature description in multiple ways, preserving
the exact same meaning each time. Vary only vocabulary and sentence
structure - do NOT change the concept described. Return ONLY a JSON
array of strings, one per paraphrase. No extra text.
```

User prompt

```
Original description: {explanation}
Generate {n} paraphrases of the above description. Keep the exact
same meaning. Return ONLY a JSON array of {n} strings.
```

B.7. Explanation Similarity

To measure semantic consistency of explanations across conditions, we compute two similarity measures:

- **Cosine similarity** of E5-small-v2 embeddings (`intfloat/e5-small-v2`). Each explanation is embedded using mean pooling over token embeddings, and pairwise cosine similarity is computed across all condition pairs for each feature.
- **BERTScore O1** (Zhang* et al., 2020) using RoBERTa-large as the reference model. BERTScore captures token-level lexical overlap weighted by contextual embeddings, providing a complementary measure less sensitive to synonym substitution than cosine similarity.

Both measures are all reported as mean pairwise within-experiment comparisons. The two measures are highly consistent across all experiments and models (Table 13).

B.8. Scoring Procedure

In the following section, we extend with details on the metrics used and the scoring prompts.

B.8.1. Metric Details

The first three metrics follow a two-phase pipeline: 1) the explainer LLM generates a natural language description of the feature from its top-activating samples and 2) a scorer LLM evaluates the explanation against held-out sequences. The last metric, Purity, does not depend on any LLM.

- SIMULATION (Bills et al., 2023). A scorer LLM is shown the explanation and a set of sequences and is asked to predict the feature's activation at each token position on a scale of 1 to 10. The simulation score is the Pearson correlation between predicted and actual activation values at the token level, ranging from -1 (inverse prediction) to 1 (perfect prediction). A score of 0 indicates that predictions are no better than chance.
- DETECTION (Paulo et al., 2025). The scorer LLM is shown the explanation and is asked to classify sequences as either activating or non-activating. The detection score is the AUC of a binary classifier, ranging from 0 to 1, with 0.5 indicating chance performance. Positive sequences are the top 10 activating sequences; negative sequences are 10 randomly sampled sequences from the corpus. Detection is "cheaper" compared to simulation, as it requires only a sequence-level binary decision, rather than token-level activation prediction/ classification of activating and non-activating samples.
- FUZZING (Paulo et al., 2025). Similar to detection, but then on the token level within activating sequences. The scorer is shown an explanation and sequences with some tokens masked, and asked to predict whether the original token is likely to activate the feature. The fuzzing score is the AUC of these token-level predictions, ranging from 0 to 1. Fuzzing is more sensitive to whether the explanation correctly identifies which tokens are relevant.
- PURITY (Puri et al., 2025). Measures whether high activations are exclusive to the target concept at the sequence level. Natural samples are rated by a scorer LLM on a three-point scale (0 = not expressed, 1 = partial, 2 = clearly expressed); ambiguous cases are discarded, leaving binary concept/non-concept labels. Average Precision (AP) is then computed using feature activation magnitudes as the ranking signal, ranging from 0 to 1. A high Purity score indicates that strongly activating sequences are specific to the target concept, while a low score indicates that unrelated sequences also produce high activations.

B.8.2. SCORING PROMPTS

We use four scoring prompts corresponding to the four metrics. All prompts are sent to `Gemini Flash` as the scorer. The `{explanation}` placeholder is replaced with the generated feature explanation; other placeholders are filled with the relevant sequences.

Simulation Implements the scoring procedure of (Bills et al., 2023). The scorer predicts token-level activation magnitudes on a 0–10 scale; the simulation score is the Pearson correlation between predicted and actual activations.

System prompt — Simulation

```
You are a neuron activation simulator for a sparse autoencoder
(SAE) trained on a language model.

Given a description of what an SAE feature detects and multiple
pieces of text, predict how strongly the feature would activate
on each text.

Return a JSON array of numbers between 0 and 10:
- 0 means the feature would not activate at all
- 10 means the feature would activate very strongly

Return ONLY the JSON array, nothing else.
Example: [3.5, 0.0, 8.2, 1.0]
```

User prompt — Simulation

```
Feature description: {explanation}

Texts:
{texts}

For each text, predict the feature activation (0-10).
Return ONLY a JSON array of {num_texts} numbers.
```

Detection Implements the detection procedure of (Paulo et al., 2025). The scorer performs binary classification of sequences as activating (1) or non-activating (0); the detection score is the AUC over positive and negative sequences.

System prompt — Detection

```
You are an intelligent and meticulous linguistics researcher.
You will be given a certain feature of text, such as "male
pronouns" or "text with negative sentiment". You will then be
given several text examples. Your task is to determine which
examples possess the feature.
For each example in turn, return 1 if the sentence is correctly
labeled or 0 if the tokens are mislabeled. You must return your
response in a valid Python list. Do not return anything else
besides a Python list.
```

User prompt — Detection

```
feature interpretation: {explanation}
Text examples:
{texts}
```

Fuzzing Implements the token-level fuzzing procedure of (Paulo et al., 2025). Activating tokens are marked with « and » delimiters; the scorer classifies whether the marked tokens correctly represent the feature. The fuzzing score is the AUC over correctly and incorrectly marked token spans.

System prompt — Fuzzing

```
You are an intelligent and meticulous linguistics researcher.
You will be given a certain feature of text, such as "male
pronouns" or "text with negative sentiment". You will be given
a few examples of text that contain this feature. Portions of
the sentence which strongly represent this feature are between
tokens << and >>.
Some examples might be mislabeled. Your task is to determine
if every single token within << and >> is correctly labeled.
Consider that all provided examples could be correct, none of
the examples could be correct, or a mix. An example is only
correct if every marked token is representative of the feature.
For each example in turn, return 1 if the sentence is correctly
labeled or 0 if the tokens are mislabeled. You must return your
response in a valid Python list. Do not return anything else
besides a Python list.
```

User prompt — Fuzzing

```
feature interpretation: {explanation}
Text examples:
{examples}
```

Purity Implements the FADE sequence rating procedure of (Puri et al., 2025). The scorer rates each sequence on a 0–2 scale according to how clearly the concept is expressed; the purity score is the average precision over rated sequences.

System prompt — Purity

```
You are tasked with building a database of sequences that best
represent a specific concept.
To create this, you will review a dataset of varying sequences
and rate each one according to how much the concept is
expressed.
For each sequence, assign a rating based on this scale:
0: The concept is not expressed.
1: The concept is vaguely or partially expressed.
2: The concept is clearly and unambiguously present.
Use conservative ratings. If uncertain, choose a lower rating
to avoid including irrelevant sequences in your database.
If no sequence expresses the concept, rate all sequences as 0.
Each sequence is identified by a unique ID. Provide your
ratings as a Python dictionary with sequence IDs as keys and
their ratings as values.
Example output: {"14": 0, "15": 2, "20": 1, "27": 0}
Output only the dictionary - no additional text, comments,
or symbols.
```

User prompt — Purity

```
Concept: {explanation}

Sequences:
{sequences}
```

B.8.3. Paraphrase Generation Prompt

Paraphrases are generated with the following prompts. The system prompt instructs the model to preserve semantic meaning while varying surface form; the user prompt specifies the original explanation and the number of paraphrases required ($n = 3$). Responses are parsed from the returned JSON array.

System prompt — Paraphrase generation

```
You are a precise language model assistant. Your task is to
rephrase a given feature description in multiple ways,
preserving the exact same meaning each time. Vary only
vocabulary and sentence structure - do NOT change the concept
described. Return ONLY a JSON array of strings, one per
paraphrase. No extra text.
```

User prompt — Paraphrase generation

```
Original description: {explanation}
Generate {n} paraphrases of the above description. Keep the
exact same meaning. Return ONLY a JSON array of {n} strings.
```

C. Instability Quantification

C.1. ICC specification

We use the Intraclass Correlation Coefficient (ICC) to quantify the proportion of total score variance attributable to stable **between-feature** difference, rather than within feature noise (Koo & Li, 2016).

There is a difference between ICC(1,1) and ICC(2,1):

- **Draw — ICC(1,1).** Random draws of activating examples are not a fixed, replicable set of conditions. Each draw is a unique random sample from the activation pool. We therefore treat draws as a one-way random effects model (ICC(1,1)), in which the conditions themselves are sampled from a larger population of possible draws.
- **Corpus, Explainer, Paraphrase — ICC(2,1).** These sources involve a fixed, replicable set of conditions (four specific corpora; two specific explainer models; four specific explanation versions). We therefore treat them as a two-way mixed effects model (ICC(2,1)), in which conditions are fixed and features are random.

However, even though these thresholds are stated, this does not indicate that anything with less than 0.75 is directly unreliable (Liljequist et al., 2019). It can be informative to look at other studies that report ICC, to receive an indication of the range of ICC values. We report the ICC for each metric (simulation, detection, fuzzing, purity) and for each source of variation (corpus, draw, explainer).

C.2. Top-*k* Jaccard Lift

Raw Jaccard values are not directly comparable across experiments because the feature pool size N differs between experiments. We therefore normalise by the expected Jaccard under random selection:

$$J_k^{\text{chance}} = \frac{k}{2N - k} \tag{4}$$

and report *lift* as the ratio of observed to chance Jaccard:

$$\text{Lift}_k = \frac{J_k^{\text{observed}}}{J_k^{\text{chance}}} \tag{5}$$

Lift $= 1$ indicates rankings no better than random; lift > 1 indicates that the same features tend to appear in the top-k set across conditions more than expected by chance. We report mean J_k and lift across all condition pairs for $k \in \{10, 20, 30, 40, 50\}$.

C.3. Cohen's d effect sizes

Unlike within-architecture experiments where the same features are observed across conditions, cross-architecture comparison does not permit per-feature reliability analysis: features are not matched across architectures and ICC or Jaccard metrics are therefore not applicable. We instead compare score *distributions* via mean scores and Cohen's d effect sizes. Formally, for metric m and architectures A and B, we compute:

$$\begin{aligned} d &= \frac{\bar{x}_{m,A} - \bar{x}_{m,B}}{s_{\text{pooled}}}, \\ s_{\text{pooled}} &= \sqrt{\frac{(n_A - 1)s_{m,A}^2 + (n_B - 1)s_{m,B}^2}{n_A + n_B - 2}} \end{aligned} \tag{6}$$

where $\bar{x}_{m,A}$, $s_{m,A}$, and n_A denote the sample mean, standard deviation, and number of features for architecture A, respectively. The small and inconsistent effect sizes observed do not prove that no architectural differences exist at the feature level; they show only that differences are not detectable at the distribution level with our sample sizes. This is nonetheless informative, because if architectural differences were large, they would be expected to show as a distributional shift that is detectable even without per-feature correspondence.

C.4. Variance Decomposition

To quantify the contribution of each pipeline choice to metric variability, we fit a variance components model via Restricted Maximum Likelihood (REML). For score s_{ijklm} of feature i under corpus j, draw k, explainer l, and architecture m:

$$s_{ijklm} = \mu + \alpha_i + \beta_j + \gamma_k + \delta_l + \epsilon_m \tag{7}$$

where μ is the grand mean and $\alpha_i, \beta_j, \gamma_k, \delta_l, \epsilon_m$ are independent mean-zero random effects for feature identity, corpus, draw, explainer, and architecture respectively:

$$\begin{aligned} \alpha_i &\sim \mathcal{N}(0, \sigma_\alpha^2), \quad \beta_j \sim \mathcal{N}(0, \sigma_\beta^2), \\ \gamma_k &\sim \mathcal{N}(0, \sigma_\gamma^2), \quad \delta_l \sim \mathcal{N}(0, \sigma_\delta^2), \\ \epsilon_m &\sim \mathcal{N}(0, \sigma_\epsilon^2) \end{aligned} \tag{8}$$

The total variance decomposes as:

$$\sigma_{\text{total}}^2 = \sigma_\alpha^2 + \sigma_\beta^2 + \sigma_\gamma^2 + \sigma_\delta^2 + \sigma_\epsilon^2 \tag{9}$$

The proportion of variance attributable to each factor f is then:

$$\rho_f = \frac{\sigma_f^2}{\sigma_{\text{total}}^2} \tag{10}$$

Variance components are estimated per metric from partially overlapping subsets, as not all pipeline combinations are fully crossed.

D. SAE Training Details

D.1. Pythia SAE models

There are three pretrained SAEs accessed in this study (Karvonen et al., 2025). For most experiments, we opt for BatchTopK SAE with k = 160. For the experiment comparing architectures, we additionally investigate a Matryoshka SAE with k = 160 and a ReLU SAE with a sparsity penalty of 0.03. All SAEs can be found at `adamkarvonen/saebench_pythia-160m-deduped_width-2pow14_date-0108`. Select the folders `trainer_3`. The three pretrained SAEs from SAEBench (Karvonen et al., 2025) that are used in this study are all trained on layer 8 of PYTHIA-160M. Table 8 summarises their hyperparameters.

Table 8. SAE hyperparameters for Pythia-160M models from SAEBench. All models available at `adamkarvonen/saebench_pythia-160m-deduped_width-2pow14_date-0108`.

Architecture	Trainer	k / penalty	Dict. size	Layer
BatchTopK	`trainer_3`	$k = 160$	$2^{14} = 16{,}384$	8
Matryoshka	`trainer_3`	$k = 160$	$2^{14} = 16{,}384$	8
Standard	`trainer_5`	$\lambda = 0.03$	$2^{14} = 16{,}384$	8

For most experiments we use the BatchTopK SAE with $k = 160$ as the primary evaluation target. For the architecture comparison (Exp. 3), we additionally evaluate the Matryoshka SAE and Standard (ReLU) SAE.

The time for one experiment on PYTHIA-160M is roughly . The experiments can be executed on a CPU.

D.2. Apertus SAE Training Details

This paper is the first to train and evaluate SAEs on Apertus (Hernández-Cano et al., 2025), a Swiss multilingual large language model trained on German, French, Italian, Romansh, and English text. SAEs were trained on layer 16 of Apertus-8B using activations collected from the Pile (uncopyrighted subset). Table 9 summarises the training hyperparameters.

Table 9. Apertus-8B SAE training hyperparameters across the three architectures evaluated. All SAEs are trained on activations from layer 16 of Apertus-8B (residual stream, $d = 4{,}096$) using the Pile (uncopyrighted) corpus.

Hyperparameter	**BatchTopK**	**Matryoshka**	**Standard**
Architecture	BatchTopK	Matryoshka (BatchTopK)	ReLU + L1
Layer	16	16	16
Residual stream dim	4,096	4,096	4,096
Dictionary size	65,536 ($\times$16)	51,200 ($\times$12.5)	65,536 ($\times$16)
Matryoshka groups	—	[6,144, 14,336, 30,720]	—
Sparsity mechanism	BatchTopK ($k = 160$)	BatchTopK ($k = 160$)	L1 ($\lambda = 4 \times 10^{-4}$)
Training steps	140,000	300,000	200,000
Batch size	4,096	4,096	4,096
Learning rate	0.001	0.001	0.001
LR warmup steps	1,000	1,000	1,000
LR decay	$\times$0.8 every 10,000	$\times$0.9 every 20,000	$\times$0.8 every 10,000
Minimum LR	1×10^{-6}	1×10^{-6}	1×10^{-6}
Auxiliary penalty	0.03125	0.03125	—
Top-k auxiliary	128	128	—
Dead latent window	50 batches	50 batches	50 batches

Table 10. SAE training and evaluation metrics across configurations.

	Rel. RMSE $\downarrow$		**Loss Recovered** $\uparrow$		**Feature Sparsity** $\uparrow$	
SAE	Train	Eval	Train	Eval	Train	Eval
ReLU (L16)	0.00111	0.00107	0.99999	0.99999	0.9377	0.9377
BatchTopK k=160	0.04125	0.04124	0.99830	0.99829	0.9976	0.9976
Matryoshka k=160	0.03345	0.04176	0.99888	0.99824	0.9969	0.9968

The size of Matryoshka is slightly smaller compared to the ReLU and BatchTopK variant. This was done with efficiency and computational limitatios

A key difference between Apertus and Pythia-160M is the activation scale: Apertus features activate with magnitudes in the hundreds to thousands, compared to single digits for Pythia-160M. The activation threshold for feature inclusion is therefore set to 100 for Apertus (vs. 0.1 for Pythia-160M). Figure 7 confirms that both thresholds yield comparable activation coverage within each model.

The large dictionary size ($d_{\text{SAE}} = 65{,}536$) combined with batch-level sparsity enforcement ($k = 160$) results in an expected per-feature activation rate of approximately:

$$P(\text{feature activates}) = \frac{k}{d_{\text{SAE}}} = \frac{160}{65{,}536} \approx 0.24\% \tag{11}$$

per token. Over a typical evaluation sequence of 32 tokens, a randomly sampled feature is therefore expected to activate on fewer than one token per sequence on average, which explains the low feature evaluability observed for BatchTopK on Apertus (Table 7). Therefore, we randomly sample 2000 features to extract around 250 activating features.

Our trained SAEs are publicly released at https://huggingface.co/REPLACE-WITH-SAE-REPO.

D.2.1. Training and Evaluation Scores

As these SAEs were trained by us, we report the train and evaluation scores. The best checkpoint was taken as the final model checkpoint.

D.3. Time and Compute

The differences in LLM calls for the two models stems from the fact that not every condition has the exact amount of features, even though we try in both cases to reach roughly 250 features.

Table 11. PYTHIA-160M: LLM calls, time and compute estimates. The code is executing on a CPU.

Experiment	LLM Calls	Time (h)
Exp1: Cross-Dataset Generalization	4,705	4.7
Exp2a: Sampling Instability	5,970	6.0
Exp2b (w=5): Cross-Explainer	2,490	2.5
Exp2b (w=10): Cross-Explainer	2,670	2.7
Exp2c: Paraphrase Stability	4,165	4.2
Exp4: Cross-SAE Architecture	3,665	3.7
Total	**23,665**	**23.7**

Table 12. APERTUS-8B: LLM calls, time and compute estimates. The code is executed using GH200 GPU on a supercomputer. The latency originates from the LLM calls, not from the GPU.

Experiment	LLM Calls	Time (h)
Exp1: Cross-Dataset Generalization	5,035	5.1
Exp2a: Sampling Instability	3,975	4.0
Exp2b (w=5): Cross-Explainer	1,780	1.8
Exp2b (w=10): Cross-Explainer	2,550	2.6
Exp2c: Paraphrase Stability	5,120	5.1
Exp4: Cross-SAE Architecture	4,190	4.2
Total	**22,650**	**27.4**

E. Extended Results

E.1. Similarity Values

Cosine similarity and BERTSCORE *of explanations remain high and narrowly distributed across all conditions, even when scores vary substantially, invalidating explanation similarity as a proxy for stability.*

Trivially, to combat instability researchers could monitor explanation similarity as a cheap diagnostic: if two explanations are semantically close, their scores should agree. Table 13 falsifies this assumption systematically across all four experiments and both models. Mean cosine similarity ranges from 0.840 and 0.824 for (CORPUS, 0.942 to 0.949 for (PARAPHRASE, and the ordering PARAPHRASE $>$ DRAW $>$ EXPLAINER $\approx$ CORPUS holds consistently. However, the score instability does not follow this ordering (tab:icc). DRAW produces the most severe instability (negative ICC for SIMULATION and PURITY) while maintaining high explanation similarity (0.892 cosine on `Pythia-160M`).

The dissociation is most striking for DRAW on feature 2208 (Appendix F), where all five draws produce near-identical explanations (cosine $= 0.964$, BERTScore $= 0.944$), yet SIMULATION scores range from -0.177 to $+0.729$. DETECTION remains fixed at 0.650 throughout. The explanation is essentially the same; while the score differs. This rules out explanation drift as the mechanism behind instability under draw variation, and points instead to the held-out evaluation sequences themselves as the source of noise.

Explanation similarity therefore cannot substitute for the direct stability measurement provided by ICC and Jaccard lift. A pipeline that produces consistent explanations may still produce unreliable scores, and this failure is invisible without running the stability checks we propose in section 4.

Table 13. Mean pairwise explanation similarity per experiment and model. Cosine similarity computed using `E5-small-v2`; BERTScore using `RoBERTa-large`. Both reported as mean ± std across all pairwise comparisons within each experiment.

			Cosine similarity		BERTScore O1	
Model	**Experiment**	n	**Mean**	**Std**	**Mean**	**Std**
`Pythia-160M`	CORPUS	258	0.840	0.028	0.844	0.020
	DRAW	239	0.892	0.036	0.874	0.031
	EXPLAINER	266	0.853	0.036	0.849	0.022
	PARAPHRASE	267	0.942	0.037	0.924	0.038
`Apertus-8B`	CORPUS	264	0.824	0.024	0.832	0.019
	DRAW	159	0.878	0.029	0.864	0.026
	EXPLAINER	256	0.841	0.034	0.841	0.020
	PARAPHRASE	256	0.949	0.022	0.924	0.016

Table 14. Variance decomposition (% of total variance) per source and metric, for `Pythia-160M` and `Apertus-8B`, with 95% confidence intervals in brackets. The dominant source per (model, metric) is shown in **bold**.

Model	**Metric**	σ^2_{corpus}	σ^2_{draw}	$\sigma^2_{\text{explainer}}$	$\sigma^2_{\text{paraphrase}}$	σ^2_{arch}
`Pythia-160M`	SIMULATION	38 [29, 48]	**50 [46, 54]**	5 [4, 7]	4 [2, 6]	2 [1, 5]
	DETECTION	**60 [47, 72]**	20 [16, 24]	11 [6, 18]	8 [5, 12]	1 [0, 3]
	FUZZING	**48 [32, 68]**	11 [10, 13]	11 [8, 15]	29 [15, 46]	0 [0, 2]
	PURITY	24 [20, 28]	**63 [57, 68]**	7 [5, 9]	7 [5, 10]	0 [0, 1]
`Apertus-8B`	SIMULATION	39 [31, 48]	**47 [43, 52]**	5 [4, 7]	5 [3, 8]	3 [1, 5]
	DETECTION	**46 [37, 57]**	17 [15, 19]	9 [7, 10]	13 [8, 20]	15 [9, 23]
	FUZZING	29 [17, 44]	8 [8, 9]	6 [5, 7]	**57 [38, 80]**	0 [0, 1]
	PURITY	23 [20, 26]	**62 [56, 68]**	9 [7, 11]	5 [4, 7]	1 [1, 3]

E.2. Variance Decomposition values

E.3. ICC Heatmap

E.4. Mean scores and standard deviation per condition

E.5. Corpus Results

Figure 9 shows the delta in scores between each condition. This shows that most of the deltas are relatively similar, even when the datasets differ. The expectation was that more divergent datasets would yield higher deltas, but this does not appear to be the case. This could be due to the inherent design of the metric or due to the possible polysemanticity of features.

E.6. Jaccard Lift Table

E.7. Draw Score Instability

DRAW contributes to a large portion of the variance, as we saw in section 3. Below, we provide a figure that clearly shows the trajectory of 20 randomly picked features for the scores per draw for each feature (Figure 10).

E.8. Explainer Sensitivity

To test whether `GPT4o-mini`'s tendency to produce longer explanations confounds the comparison, we additionally ran a length-constrained condition in which both explainers were restricted to a maximum of ten words.

Figure 11 compares mean scores between unrestricted and length-constrained conditions for both explainers. Restricting explanation length to ten words reduces SIMULATION and DETECTION scores for both explainers on `Apertus-8B` but has minimal effect on `Pythia-160M`. FUZZING shows a small differences between explainers for both models, which could indicate that the score is least sensitive to explanation detail, compared to other metrics.

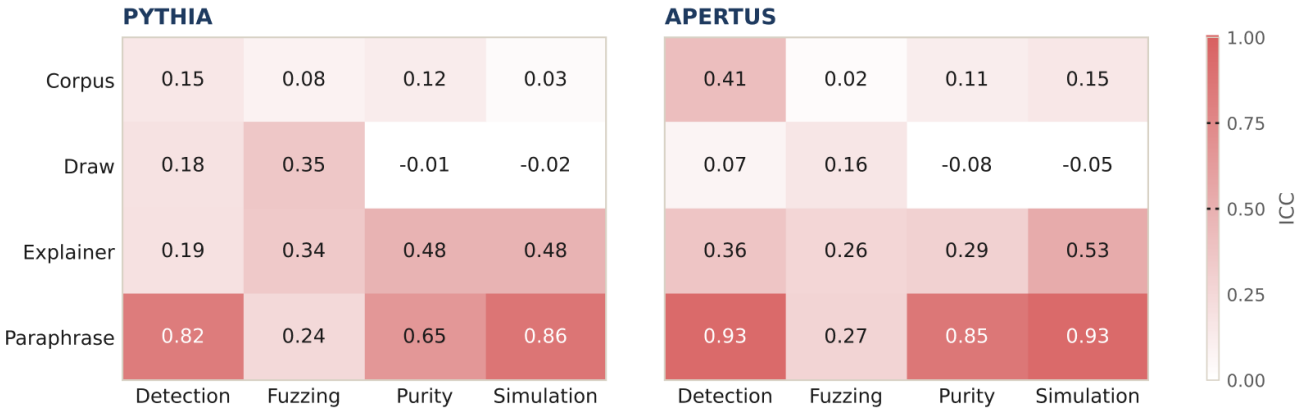

Figure 8. ICC heatmap with the delta of the mean for each corpus and metric

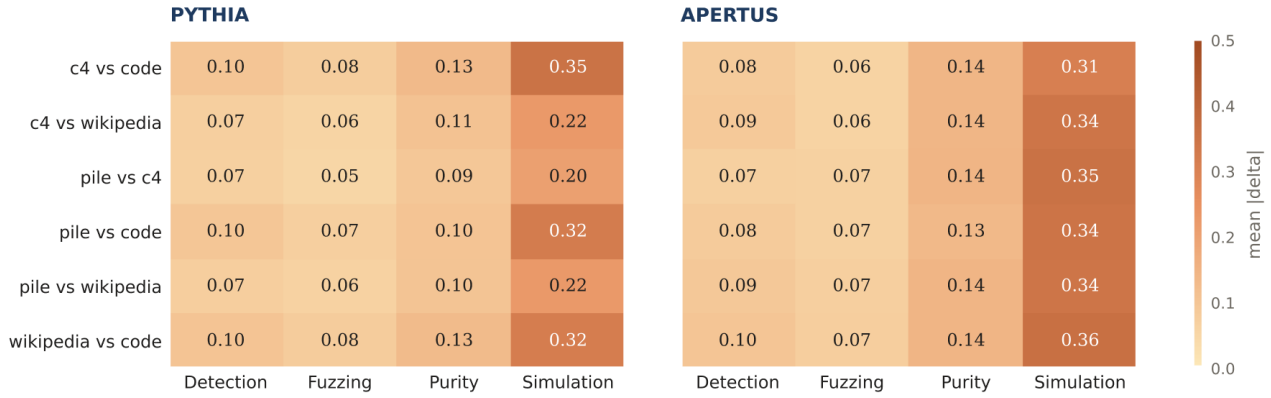

Figure 9. Corpus heatmap with the delta of the mean for each corpus combination and metric

E.9. Paraphrase Sensitivity

Mean pairwise cosine similarity across the four versions (original + three paraphrases) is 0.942 for Pythia-160M and 0.949 for Apertus-8B (Table 13), which is substantially higher than cross-explainer similarity (≈0.853) or cross-corpus similarity (≈0.841).

Figure 5 shows per-feature score trajectories across the original explanation and three paraphrases. The trajectories of all metrics show substantial crossing and non-monotonic patterns despite the high explanation similarity (mean cosine $= 0.945$, Table 13). Next to that, Figure 12 shows the score of the original paraphrase vs. the mean score of the alternative paraphrases. The lines for SIMULATION and DETECTION are relatively similar to $y = x$, but varies the most for FUZZING.

E.9.1. Paraphrase Generator Model

For the original experiment, we keep the explainer, scorer and paraphrases model fixed to `Gemini Flash`. However, we conducted a small experiment, by changing the paraphrase generator to `GPT4o-mini`, to observe the effects of the paraphase model.

Table 17 reports ICC(2,1) and Jaccard overlap at top-10 and top-20 thresholds for two paraphrase models across four metrics. Overall, SIMULATION and DETECTION scores show high reliability under paraphrasing, with ICC(2,1) values exceeding 0.8 for both `GPT4o-mini` and `Gemini Flash`. FUZZING exhibits substantially lower agreement, with an ICC(2,1) of 0.328 for `GPT4o-mini` and a confidence interval crossing zero, indicating near-chance reliability. PURITY falls in an intermediate range, with `Gemini Flash` yielding higher agreement (0.650) than `GPT4o-mini` (0.459).

At the rank level, Jaccard lift reveals a consistent pattern: `Gemini Flash` paraphrases produce more stable feature

Table 15. Mean scores and standard deviations per condition, metric, and model across all experiments. As mentioned earlier, the mean and variance cover up the instability of different conditions.

		Simulation		Detection		Fuzzing		Purity	
Exp.	**Condition**	**mean**	**std**	**mean**	**std**	**mean**	**std**	**mean**	**std**
Pythia-160M									
Corpus	Pile	0.705	0.255	0.655	0.086	0.559	0.091	0.628	0.110
	C4	0.674	0.302	0.673	0.113	0.572	0.091	0.628	0.115
	Wikipedia	0.663	0.310	0.671	0.116	0.567	0.098	0.623	0.123
	Code	0.567	0.415	0.688	0.136	0.533	0.097	0.621	0.146
Draw	Draw 0	0.206	0.347	0.611	0.053	0.536	0.043	0.554	0.208
	Draw 1	0.273	0.292	0.617	0.057	0.538	0.044	0.623	0.199
	Draw 2	0.522	0.244	0.629	0.054	0.543	0.046	0.485	0.102
	Draw 3	0.081	0.212	0.618	0.063	0.534	0.045	0.394	0.109
	Draw 4	−0.124	0.228	0.616	0.051	0.542	0.054	0.302	0.122
Explainer	`Gemini Flash`	0.740	0.187	0.643	0.058	0.555	0.055	0.621	0.106
	`GPT4o-mini`	0.753	0.174	0.637	0.047	0.550	0.051	0.615	0.098
Paraphrase	Original	0.670	0.295	0.671	0.103	0.563	0.108	0.637	0.107
	Paraphrase 1	0.649	0.314	0.663	0.114	0.565	0.090	0.610	0.139
	Paraphrase 2	0.635	0.323	0.660	0.111	0.567	0.089	0.605	0.157
	Paraphrase 3	0.634	0.325	0.663	0.110	0.563	0.091	0.606	0.154
Architecture	BatchTopK	0.682	0.302	0.657	0.095	0.554	0.104	0.613	0.130
	Matryoshka	0.602	0.435	0.678	0.126	0.546	0.110	0.622	0.131
	Standard	0.756	0.221	0.658	0.083	0.545	0.066	0.633	0.095
Apertus-8B									
Corpus	Pile	0.532	0.418	0.694	0.146	0.523	0.115	0.618	0.153
	C4	0.517	0.426	0.680	0.139	0.515	0.091	0.583	0.148
Draw	Draw 0	0.151	0.329	0.604	0.051	0.520	0.036	0.559	0.216
	Draw 1	0.205	0.276	0.599	0.049	0.523	0.039	0.610	0.223
	Draw 2	0.459	0.306	0.600	0.047	0.522	0.036	0.460	0.114
	Draw 3	0.085	0.226	0.603	0.047	0.526	0.037	0.398	0.117
	Draw 4	−0.128	0.262	0.601	0.050	0.524	0.037	0.283	0.093
Explainer	`Gemini Flash`	0.672	0.225	0.617	0.052	0.520	0.040	0.564	0.122
	`GPT4o-mini`	0.709	0.201	0.623	0.053	0.525	0.044	0.589	0.108
Paraphrase	Original	0.492	0.437	0.703	0.157	0.526	0.107	0.629	0.157
	Paraphrase 1	0.482	0.463	0.700	0.158	0.522	0.101	0.619	0.159
	Paraphrase 2	0.494	0.456	0.698	0.159	0.528	0.107	0.624	0.160
	Paraphrase 3	0.501	0.449	0.699	0.156	0.514	0.106	0.624	0.163
Architecture	BatchTopK	0.508	0.436	0.700	0.152	0.523	0.123	0.623	0.151
	Matryoshka	0.675	0.241	0.619	0.055	0.525	0.040	0.575	0.123
	Standard	0.637	0.259	0.618	0.068	0.527	0.039	0.554	0.138

rankings than `GPT4o-mini` across all metrics, particularly at the top-10 threshold. DETECTION under `Gemini Flash` achieves a top-10 Jaccard of $43.76\times$, compared to $34.9\times$ for `GPT4o-mini`. FUZZING again shows the weakest rank stability across both models. Taken together, these results suggest confirms that reliability varying substantially across metrics and paraphrase models.

F. Qualitative Examples

F.1. Exp.1 examples - Corpus

Feature 3070 on Pythia160M Simulation scores drop on the Wikipedia corpus (range $= 0.369$), accompanied by a qualitative shift in the generated explanation from product and service descriptions on the Pile, C4, and Code corpora to disclosures and directives on Wikipedia. Despite moderate overall explanation similarity (cosine $= 0.831$, BERTScore $= 0.833$), the Wikipedia corpus retrieves a sufficiently different set of activating examples to produce a divergent explanation and substantially lower scores across simulation and purity.

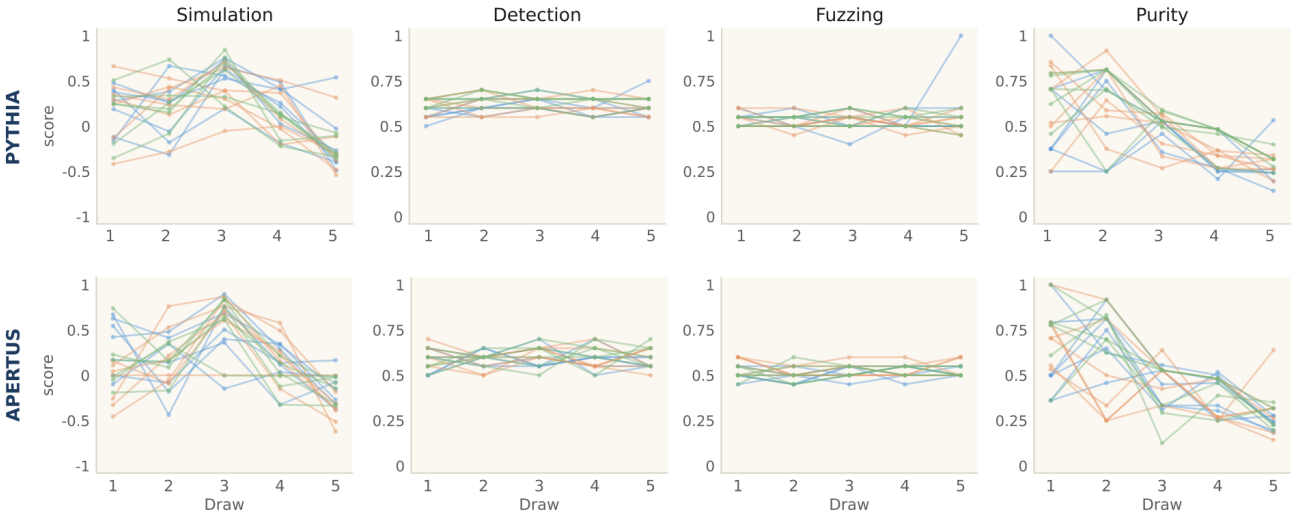

Figure 10. The score across draws for each feature. One line represents one feature. Both SIMULATION and PURITY show a high variation in scores for each draw.

Example: Corpus instability (Feature 3070, BatchTopK, Pythia-160M)

Pile "Descriptions of products, services, or technical processes"
Sim: 0.813 Det: 0.650 Fuzz: 0.550 Pur: 0.639

C4 "Promotional or descriptive text for businesses, products"
Sim: 0.741 Det: 0.650 Fuzz: 0.500 Pur: 0.679

Wikipedia "Disclosures, revelations, or directives"
Sim: 0.444 Det: 0.550 Fuzz: 0.550 Pur: 0.417

Code "Assembly language comments, often related to low-level"
Sim: 0.743 Det: 0.650 Fuzz: 0.550 Pur: 0.639

Feature 7223 on Pythia160M Simulation scores remain stable across all four corpora (range $= 0.187$) despite a notable semantic shift on the Code corpus, where the explainer describes computer viruses rather than biological pathogens (cosine $= 0.864$, BERTScore $= 0.840$). That the scorer assigns similar simulation scores to both interpretations suggests the feature captures a broader concept of "virus" or "infection" that spans both domains and that this conceptual stability is recoverable across corpora even when the explanation surface form diverges.

Example: Corpus stability (Feature 7223, BatchTopK, Pythia-160M)

Pile "Specific bacterial species, often in the context of infection"
Sim: 0.874 Det: 0.600 Fuzz: 0.550 Pur: 0.639

C4 "Mentions of specific diseases, viruses, bacteria, or. . . "
Sim: 0.881 Det: 0.650 Fuzz: 0.550 Pur: 0.679

Wikipedia "Diseases, infections, and medical conditions"
Sim: 0.694 Det: 0.550 Fuzz: 0.550 Pur: 0.583

Code "Descriptions or source code comments pertaining to computer viruses, often including their names, authors, or technical details"
Sim: 0.793 Det: 0.650 Fuzz: 0.600 Pur: 0.679

Feature 63699 on Apertus-8B Despite divergent explanations across corpora, ranging from headings and titles on the Pile to assembly language on Code, simulation scores remain stable (range $= 0.146$, cosine $= 0.836$, BERTScore $= 0.828$). Detection and fuzzing are similarly consistent throughout. This feature illustrates that corpus explanation divergence does not necessarily produce score instability: the scorer assigns comparable scores to qualitatively different descriptions of the same underlying feature.

Figure 11. Mean metric scores for unrestricted and W10 (10-word-constrained) explanations, per explainer and model. Restricting explanation length reduces Simulation scores for both explainers but has minimal effect on Detection and Fuzzing.

Example: Corpus stability (Feature 63699, BatchTopK, Apertus-8B)

Pile "Headings, titles, or introductory sections"
Sim: 0.677 Det: 0.600 Fuzz: 0.550 Pur: 0.533

C4 "Descriptions of physical properties, processes, or characteristics of objects"
Sim: 0.773 Det: 0.600 Fuzz: 0.550 Pur: 0.533

Wikipedia "Attribution of statements, observations, or beliefs "
Sim: 0.823 Det: 0.600 Fuzz: 0.500 Pur: 0.639

Code "Assembly language code, including comments (`;` or...)"
Sim: 0.822 Det: 0.650 Fuzz: 0.550 Pur: 0.679

Feature 4081 on Apertus-8B Simulation scores vary substantially across corpora (range $= 0.327$), accompanied by qualitatively divergent explanations that share no obvious common concept, ranging from titles and headings on the Pile to passive voice constructions on Wikipedia and assembly language on Code (cosine $= 0.814$, BERTScore $= 0.836$). Purity varies most dramatically, dropping to 0.250 on C4.

Example: Corpus instability (Feature 4081, BatchTopK, Apertus-8B)

Pile "Titles, headings, or introductory sentences that summarize"
Sim: 0.675 Det: 0.600 Fuzz: 0.500 Pur: 0.533

C4 "This feature activates at the end of a phrase"
Sim: 0.811 Det: 0.650 Fuzz: 0.500 Pur: 0.250

Wikipedia "Passive voice constructions where an action is performed 'by'"
Sim: 0.636 Det: 0.600 Fuzz: 0.500 Pur: 0.533

Code "Assembly language source code, particularly comments"
Sim: 0.484 Det: 0.600 Fuzz: 0.500 Pur: 0.325

F.2. Exp. 2a examples - Draws

Feature 13746 on Pythia160M Across five independent draws from the same corpus, the explainer consistently identifies this feature as related to formal or technical writing (cosine similarity $= 0.883$). Yet simulation scores range from -0.316 to 0.707, which is a large spread of over one unit. Detection and fuzzing remain stable throughout (variance < 0.003), while purity varies substantially (variance $= 0.038$).

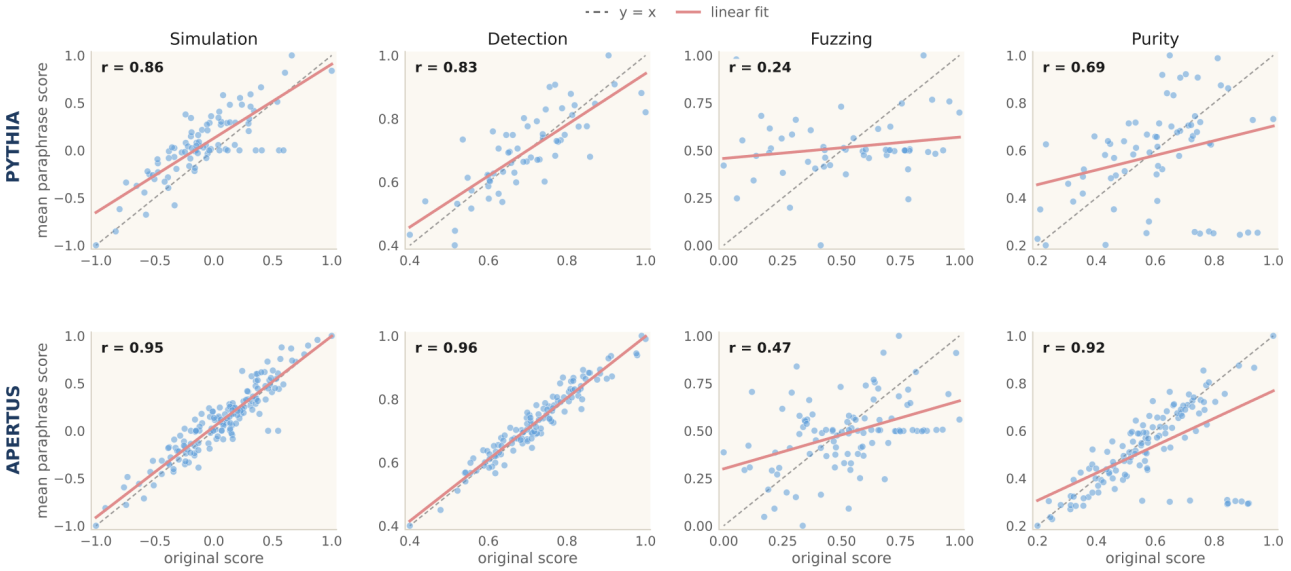

Figure 12. The score of the original paraphrase vs. the average of the other paraphrases.

Exp. 2a, Feature 13746, BatchTopK, Pythia-160M)

Draw 0 "Academic paper sections like *Abstract* and *Introduction*"
Sim: −0.316 Det: 0.600 Fuzz: 0.500 Pur: 0.778

Draw 1 "Formal or technical documents, including scientific paper abstracts, legal"
Sim: 0.401 Det: 0.650 Fuzz: 0.500 Pur: 0.250

Draw 2 "Technical questions or discussions, often from forums"
Sim: 0.707 Det: 0.600 Fuzz: 0.550 Pur: 0.556

Draw 3 "Formal or technical documents, such as academic papers"
Sim: 0.451 Det: 0.500 Fuzz: 0.550 Pur: 0.476

Draw 4 "Technical discussions, problem-solving, or academic content"
Sim: −0.155 Det: 0.650 Fuzz: 0.550 Pur: 0.276

Feature 2208 on Pythia160M This feature produces the highest explanation similarity of any example in this section (cosine = 0.964, BERTScore = 0.944): all five draws generate near-identical descriptions of algebraic equations and variable assignments. Yet simulation scores span a range of 0.906 — from −0.177 to 0.729, while detection remains fixed at 0.650 throughout. This is an illustration in our data of the dissociation between explanation similarity and score stability: explanations that are essentially paraphrases of each other produce simulation scores that vary by nearly the full metric range.

Example: Sampling instability (Feature 2208, BatchTopK, Pythia160M)

Draw 0 "Algebraic equations, variable assignments, and mathematical expressions"
Sim: +0.251 Det: 0.650 Fuzz: 0.550 Pur: 0.395

Draw 1 "Algebraic equations, variable assignments, and mathematical expressions"
Sim: +0.729 Det: 0.650 Fuzz: 0.600 Pur: 0.793

Draw 2 "Algebraic equations, variable assignments, and mathematical problem"
Sim: +0.498 Det: 0.650 Fuzz: 0.550 Pur: 0.443

Draw 3 "Algebraic equations, variable assignments, and mathematical expressions"
Sim: −0.008 Det: 0.650 Fuzz: 0.550 Pur: 0.415

Draw 4 "Mathematical equations and algebraic expressions, often involving multiple variables"
Sim: −0.177 Det: 0.650 Fuzz: 0.550 Pur: 0.303

Feature 45056 on Apertus-8B Simulation scores span the full metric range across five draws (range = 1.001, from −0.472 to +0.529), with explanations that shift between related but distinct concepts, such as proper nouns, biographical

information, names and titles, probably depending on which examples are sampled (cosine $= 0.869$, BERTScore $= 0.846$). Detection and fuzzing are relatively stable, though detection drops to 0.500 on Draw 2, indicating chance-level sequence classification. Purity varies substantially (range $= 0.389$), suggesting that the semantic coherence of activating examples itself depends on which examples are drawn.

Example: Sampling instability (Feature 45056, BatchTopK, Apertus-8B)

Draw 0 "Proper nouns, often capitalized, appearing at the beginning"
Sim: -0.125 Det: 0.600 Fuzz: 0.550 Pur: 0.411

Draw 1 "Biographical information, often including names"
Sim: $+0.166$ Det: 0.600 Fuzz: 0.550 Pur: 0.700

Draw 2 "Specific names or titles followed by a brief description "
Sim: $+0.529$ Det: 0.500 Fuzz: 0.500 Pur: 0.333

Draw 3 "Proper nouns, often names of people, places"
Sim: $+0.148$ Det: 0.650 Fuzz: 0.500 Pur: 0.525

Draw 4 "Specific names, places, or technical terms"
Sim: -0.472 Det: 0.650 Fuzz: 0.550 Pur: 0.318

Table 16. Top-k Jaccard lift (observed/chance) per metric and source of variation at $k = 10$ and $k = 20$, for `Pythia-160M` and `Apertus-8B`, with 95% confidence intervals in brackets. Lift $= 1\times$ indicates feature selection no better than random. Values below $2\times$ are shown in **bold** to indicate near-chance selection. † Point estimate falls outside BCa (bias-corrected and accelerated) CI due to skewness in the pairwise Jaccard distribution; the CI should be interpreted as the primary reliability indicator.

Pythia-160M					
Source	k	**SIMULATION**	**DETECTION**	**FUZZING**	**PURITY**
CORPUS	10	**0.68× [0.00, 2.63]**	4.34× [2.85, 10.01]	2.81× [1.00, 4.03]	3.85× [2.09, 8.53]
	20	**1.21× [0.80, 2.05]**	**1.60× [0.96, 2.55]**	2.61× [1.42, 3.32]	3.52× [2.08, 5.08]
DRAW	10	**0.82× [0.00, 1.78]**	2.09× [0.00, 3.21]	6.19× [3.46, 5.72]†	**1.68× [0.41, 5.33]**
	20	**1.43× [0.59, 2.37]**	**1.76× [0.90, 2.82]**	3.80× [2.68, 3.62]†	**1.35× [0.54, 1.94]**
EXPLAINER	10	12.15× [0.88, 20.83]	2.56× [0.00, 8.58]	8.58× [0.00, 16.20]	20.83× [8.58, 39.76]
	20	9.03× [5.05, 14.28]	4.20× [2.64, 7.87]	9.03× [5.95, 15.62]	11.46× [6.91, 16.52]
PARAPHRASE	10	28.22× [9.25, 42.87]	42.87× [34.93, 52.40]	9.25× [0.00, 22.46]	42.87× [22.46, 52.40]
	20	12.37× [6.28, 17.13]	19.00× [12.37, 23.25]	9.75× [6.42, 15.42]	21.03× [15.42, 25.70]
Apertus-8B					
Source	k	**SIMULATION**	**DETECTION**	**FUZZING**	**PURITY**
CORPUS	10	**0.97× [0.00, 2.01]**	9.22× [5.09, 14.42]	**1.65× [0.32, 2.50]**	4.57× [1.97, 8.22]
	20	**1.23× [0.64, 2.32]**	5.35× [3.52, 7.85]	**1.73× [1.08, 2.61]**	2.22× [1.13, 2.61]
DRAW	10	**0.97× [0.32, 1.62]**	**0.99× [0.18, 1.54]**	4.52× [2.90, 8.63]	2.22× [1.03, 3.70]
	20	**0.85× [0.55, 1.08]**	**1.81× [1.13, 2.44]**	2.68× [1.79, 3.55]	**1.29× [0.81, 1.85]**
EXPLAINER	10	6.07× [1.81, 14.74]	6.07× [1.81, 18.52]	8.60× [6.07, 14.74]	3.82× [0.00, 6.07]
	20	3.54× [1.86, 7.16]	3.54× [1.35, 6.33]	2.95× [1.35, 5.57]	3.54× [0.98, 5.57]
PARAPHRASE	10	43.76× [33.39, 49.80]	43.76× [35.29, 49.80]	12.07× [7.59, 22.78]	30.20× [15.15, 38.23]
	20	15.47× [11.16, 19.73]	21.72× [19.67, 24.40]	7.17× [5.61, 10.50]	19.03× [15.34, 22.46]

Table 17. An overview of the results of changing the paraphrase model.

	Metric	ICC(2,1)	Jaccard Top-10	Jaccard Top-20
GPT4o-mini	Simulation	0.797 [0.685, 0.884]	17.5× [2.8, 22.5]	9.7× [3.7, 12.4]
	Detection	0.868 [0.784, 0.916]	34.9× [9.2, 42.9]	19.0× [9.3, 23.3]
	Fuzzing	0.328 [−0.164, 0.679]	17.5× [5.8, 34.9]	7.5× [4.5, 13.8]
	Purity	0.459 [0.333, 0.601]	28.2× [9.2, 34.9]	13.8× [6.4, 23.3]
Gemini Flash	Simulation	0.860 [0.783, 0.930]	43.76× [33.39, 49.80]	15.47× [11.16, 19.73]
	Detection	0.824 [0.699, 0.895]	43.76× [35.29, 49.80]	21.72× [19.67, 24.40]
	Fuzzing	0.240 [0.034, 0.498]	12.07× [7.59, 22.78]	7.17× [5.61, 10.50]
	Purity	0.650 [0.523, 0.753]	30.20× [15.15, 38.23]	19.03× [15.34, 22.46]