

Master's thesis
Master's Programme in Data Science

Label-Space-Aware Evaluation of Sparse Autoencoder Feature Monosemanticity

Yunjing Ma

May 21, 2026

Supervisor(s): Professor Michael Mathioudakis

Examiner(s): Professor Kai Puolamäki
Professor Michael Mathioudakis

University of Helsinki
Faculty of Science

P. O. Box 68 (Pietari Kalmin katu 5)
00014 University of Helsinki

HELSINGIN YLIOPISTO — HELSINGFORS UNIVERSITET — UNIVERSITY OF HELSINKI

Tiedekunta — Fakultet — Faculty	Koulutusohjelma — Utbildningsprogram — Degree programme
Faculty of Science	Master's Programme in Data Science

Tekijä — Författare — Author

Yunjing Ma

Työn nimi — Arbetets titel — Title

Label-Space-Aware Evaluation of Sparse Autoencoder Feature Monosemanticity

Työn laji — Arbetets art — Level	Aika — Datum — Month and year	Sivumäärä — Sidantal — Number of pages
Master's thesis	May 21, 2026	69

Tiivistelmä — Referat — Abstract

Sparse autoencoders (SAEs) are increasingly used to recover interpretable features from the dense activation spaces of large language models, but it remains difficult to evaluate whether their learned features are genuinely monosemantic. This thesis proposes a label-space-aware evaluation framework that treats class alignment as a scalable operational proxy for SAE feature monosemanticity. At the same time, it avoids making the stronger claim that class alignment alone is sufficient to prove interpretability. The framework combines two complementary stages. First, normalized Shannon entropy and KL divergence measure how each feature's activation mass is distributed across a predefined label space. Second, reverse precision/recall validation tests whether selected class-aligned features can recover examples of their corresponding classes. Using Gemma Scope residual-stream SAEs for Gemma 2 2B, the study evaluates 15 SAE configurations across three layers and five sparsity levels, with both document-level topic labels and token-level PII entity labels. The results show that label granularity is a key condition for meaningful entropy-based evaluation. Document-level labels produce weak and compressed alignment signals, whereas fine-grained token-level labels reveal substantially clearer class-aligned structure. The analysis also shows that sparsity and layer depth affect the strength and reliability of class alignment. It further suggests that low entropy alone may be misleading if prior information and reverse-validation checks are not considered. Overall, this thesis supports a label-space-aware approach to SAE feature evaluation. In this approach, information-theoretic metrics and reverse precision/recall validation are used as scalable intermediate tools for identifying candidate class-aligned features, not as final proof of monosemanticity.

ACM Computing Classification System (CCS):
Computing methodologies → Artificial intelligence → Natural language processing → Natural language generation
Computing methodologies → Machine learning → Learning paradigms → Unsupervised learning

Avainsanat — Nyckelord — Keywords

sparse autoencoders, large language models, interpretability, monosemanticity

Säilytyspaikka — Förvaringsställe — Where deposited

Muita tietoja — Övriga uppgifter — Additional information

The source code for this thesis is available at https://github.com/Yunjing-Elio-Ma/LAEM.

Contents

1 Introduction **1**

1.1 Background and motivation 1
1.2 From neurons to features 2
1.3 Sparse autoencoders 2
1.4 Research gap 3
1.5 Research questions and scope 4
1.6 Thesis approach 5
1.7 Contributions 5
1.8 Thesis outline 7

2 Literature Review **9**

2.1 Foundations of mechanistic interpretability 9
2.2 Superposition and polysemanticity 10
2.3 Sparse autoencoder architectures 11
2.3.1 Standard L_1 SAEs 11
2.3.2 Gated SAEs 12
2.3.3 Top-K and Batch Top-K SAEs 12
2.3.4 JumpReLU and ProLU SAEs 12
2.3.5 Advanced SAE architectures and feature absorption 13
2.4 Evaluation of sparse autoencoders 13
2.4.1 Structural proxy metrics 14
2.4.2 Automated interpretability methods 14
2.4.3 Probing-based evaluation 15
2.4.4 Causal disentanglement metrics 16
2.4.5 Feature monosemanticity scores 16
2.4.6 Distributional and information-theoretic perspectives 17

3 Experimental Setup **19**

3.1 Base language model 19
3.2 SAE suite 20

3.3 Datasets and label spaces . . . 21
3.3.1 AG News . . . 22
3.3.2 DBpedia14 . . . 22
3.3.3 PII-masking-300k . . . 22
3.3.4 Ethical considerations of datasets . . . 23
3.4 Activation extraction and feature statistics . . . 24
3.5 Experimental design . . . 24

4 Methodology and Implementation 27
4.1 Information-theoretic framework . . . 27
4.1.1 Motivation: From monosemanticity to class alignment . . . 27
4.1.2 Symbols and notation . . . 28
4.1.3 Entropy and KL divergence . . . 29
4.2 Reverse precision/recall validation framework . . . 31
4.2.1 Motivation: From feature-to-class to class-to-feature . . . 31
4.2.2 P/R evaluation framework . . . 32
4.2.3 Candidate pools and ranking groups . . . 33
4.3 Implementation details . . . 35
4.3.1 Implementation of H/KL computation . . . 35
4.3.2 Implementation of P/R validation . . . 36

5 Results and Analysis 39
5.1 Information-theoretic results . . . 39
5.1.1 Label granularity and the dynamic range of H/KL . . . 39
5.1.2 Effect of sparsity . . . 41
5.1.3 Effect of layer depth . . . 42
5.1.4 Joint distribution of H and density . . . 44
5.1.5 The spurious low-entropy trap . . . 46
5.1.6 Summary of H/KL results . . . 47
5.2 Reverse P/R validation . . . 47
5.2.1 Main validation results . . . 48
5.2.2 Validating H/KL against low-frequency, high-frequency, and random baselines . . . 48
5.2.3 Choice of primary P/R metrics . . . 49
5.2.4 Sparsity and layer effects in P/R validation . . . 50
5.2.5 Summary of P/R validation results . . . 52

6 Conclusion 55
6.1 Summary of the study . . . 55

6.2 Main conclusions 56
6.3 Contributions 57
6.4 Implications for SAE evaluation 57
6.5 Limitations 58
6.6 Future work 59
6.7 Final remarks 60

Bibliography **61**

Appendix A Full P/R Results Table **67**
A.1 Full P/R Results for 15 SAEs on PII-noO 67

Appendix B Disclaimer on the Use of AI Tools **69**

1. Introduction

1.1 Background and motivation

The rapid development of large language models (LLMs) in recent years has accelerated progress in natural language processing, complex reasoning, and multimodal generation. With larger parameter scales, advanced transformer architectures, and extensive training data, these foundation models have shown emerging abilities. In some tasks, their performance can approach the human level [21, 44]. Despite these gains, the internal computational basis of such abilities is still not well understood. This is because deep neural networks, especially large transformer-based models, often rely on complex internal representations and computations that are difficult to interpret [8, 46]. This opacity creates challenges not only for scientific understanding, but also for reliability, controllability, and safety.

One source of this difficulty is that LLMs encode information through dense, high-dimensional continuous vectors that do not directly correspond to human-interpretable concepts [31]. As a result, strong behavioral performance does not necessarily explain how the model works internally. It may still be unclear which representations support the behavior. It may also be unclear whether the model relies on robust abstractions, brittle heuristics, hidden biases, or unsafe patterns [9]. Traditional evaluation methods mainly rely on external behavioral testing and input-output benchmarking, but these tests cannot directly reveal how internal representations are structured [28]. For this reason, there is a growing focus on analyzing model internals rather than only relying on external behavior.

This broader motivation has contributed to the rise of mechanistic interpretability, a branch of explainable artificial intelligence (AI). Mechanistic interpretability aims to make model behavior more interpretable by analyzing internal representations in a more direct and component-level way. Instead of treating explanation as a post-hoc description of model behavior, it seeks to identify the representational units and mechanisms of that behavior [8, 36]. By doing so, the complex transformer computations can be understood in much simpler, human-readable terms.

1.2 From neurons to features

A central question in mechanistic interpretability is what the basic units of representation should be. Early work often treated a single neuron as the fundamental unit and assumed that each neuron matched a specific feature [5]. Later empirical studies show that this assumption rarely holds for LLMs. Most neurons are not monosemantic, meaning that they do not represent a single, well-isolated concept [10,14]. Instead, models show strong polysemanticity, where one neuron can activate for several unrelated patterns. For instance, previous studies reported neurons in small transformer models that respond to Arabic script, HTTP requests, and academic citations at the same time. Because of this entanglement, neuron-level interpretation methods are no longer sufficient for understanding the concepts that drive model behavior.

A widely used explanation for why polysemanticity is so common is the superposition hypothesis [14]. In natural language, the number of meaningful features, such as concepts, syntactic rules, and semantic nuances, can be much larger than the number of neural dimensions available to the network. To maximize representational capacity and predictive accuracy within a limited parameter budget, the neural network does not assign one feature to one orthogonal direction [8]. Instead, it represents many features as dense vectors that are close to orthogonal but not perfectly separated within a shared latent space. This mechanism allows the model to behave like a much larger and sparser network, which greatly increases its capacity. However, this compression also creates geometric interference because multiple features share the same representational space [14]. As a result, the neuron basis used by the network does not align with the conceptual features that researchers want to study.

This observation has shifted the research focus from individual neurons to feature-level methods, which aim to recover a more meaningful basis from dense activation spaces. Therefore, the main task is no longer just to examine raw neuron activations, but to identify latent features that better reflect the structure of model computation.

1.3 Sparse autoencoders

Sparse autoencoders (SAEs) have emerged as one of the most promising tools for this purpose. In recent years, they have shown strong results in addressing neural superposition and have become a key tool in mechanistic interpretability [16,19]. SAEs function as unsupervised, weak dictionary-learning algorithms that are attached to the frozen layers of a pre-trained LLM. During the forward pass, an SAE extracts the dense activation vectors from a specific component, typically the residual stream or the MLP output [10]. It then maps these activations into an overcomplete latent space with a

much higher dimensionality than the original representation.

A key part of this approach is the use of sparsity constraints. Using mathematical penalties or hard sparsity constraints, such as L_1 regularization or Top-K routing, the SAE is encouraged to keep most latent units inactive for any given input [16]. This promotes the recovery of sparse feature directions in the activation space, rather than encoding many concepts in a dense superposed form [10,19]. When successfully trained, the SAE may produce a feature dictionary whose units are closer to monosemanticity, which means that each feature activates more selectively for a specific, fine-grained, human-interpretable pattern. By reconstructing the original activations from these sparse features, researchers can study mechanisms, such as grammatical structure and semantic regularities. In some cases, they can also use feature directions for causal steering.

At the same time, SAE research has revealed two broad types of challenge. The first concerns architecture and optimization, including issues such as shrinkage bias, dead latents, and feature absorption [13, 16, 19]. The second concerns evaluation: even when an SAE reconstructs activations well and achieves strong sparsity, it remains difficult to determine whether its learned features truly correspond to clear, stable, and interpretable concepts [22]. This thesis focuses primarily on the latter.

1.4 Research gap

Although SAEs have quickly developed from a promising interpretability method into a widely used tool for analyzing language model activations, important gaps remain in how their learned features are evaluated [10, 19]. Recent work has improved SAE training at scale, introduced stronger architectures, and developed broader benchmark suites for comparing reconstruction, disentanglement, and intervention-oriented performance [22]. However, these advances also show that the main open problem is not only how to train better SAEs. It is also how to evaluate what their learned features actually mean.

This evaluation problem is closely tied to the main motivation for using SAEs: they are expected to learn features that are more monosemantic than raw neurons. Ideally, a feature should respond to one stable and human-interpretable pattern [10]. However, it is hard to prove monosemanticity just by looking at activation data. A few highly active examples might make a feature seem interpretable, but this does not mean the feature is truly reliable, selective, or meaningful across a large dataset. Therefore, this thesis views monosemanticity as the ultimate interpretability goal, but focuses on a measurable proxy: class alignment. In many practical settings, one tractable way to operationalize feature meaning is to ask whether a feature is selectively associated

with a known semantic class [23]. Here, class alignment measures how much of a feature's activation is concentrated on one or a few predefined categories. While it is not the same as monosemanticity, class alignment offers a scalable and objective way to evaluate a key aspect of it.

Against this background, the first gap concerns the limitations of existing structural proxy metrics. Traditional SAE evaluation has relied heavily on the sparsity-fidelity trade-off [16,22]. These metrics are useful for quantifying reconstruction quality and representational efficiency, but they cannot directly establish whether a feature corresponds to a single, clear, human-understandable concept. A stronger reconstruction-sparsity balance does not automatically imply stronger interpretability. This creates the need for an evaluation criterion that is closer to feature meaning than reconstruction loss, while still being scalable across many features.

The second gap concerns the interpretation and boundary conditions of selectivity-based metrics. Measures such as entropy, concentration, and related distributional statistics may provide useful evidence about how strongly a feature is concentrated with respect to a chosen label space [5, 22]. However, such metrics should not be equated directly with monosemanticity [10, 26]. A feature may be strongly concentrated in one topic but still respond to several different local patterns within that topic. In contrast, a truly interpretable feature may appear across multiple topics and therefore may not show strong topic-level concentration. This issue becomes especially important when the label space is coarser than the granularity of the learned feature.

Taken together, these two gaps show that SAE feature monosemanticity requires a careful evaluation framework. The problem has two aspects. First, monosemanticity must be operationalized in a way that is scalable and empirically checkable. Second, the conditions under which this operationalization is informative must be made explicit. This thesis addresses this problem by studying class alignment as a constrained operational proxy for monosemanticity. The following section turns these two gaps into the research questions (RQs) that guide the thesis.

1.5 Research questions and scope

Guided by the two gaps above, the thesis addresses three RQs. The first gap motivates RQ1 and RQ3, which focus on how SAE feature monosemanticity can be operationalized and validated beyond structural proxy metrics. The second gap motivates RQ2, which examines the conditions under which this operationalization is informative.

- **RQ1:** How can SAE feature monosemanticity be evaluated under a scalable and empirically grounded framework?

- **RQ2:** How do label granularity, SAE sparsity, and layer depth affect SAE feature monosemanticity?

- **RQ3:** Can features identified as monosemantic also function as useful detectors for the corresponding classes?

The scope of the thesis is intentionally controlled. The experiments use one selected open-source LLM, one SAE family, one activation-site type, and a fixed SAE feature width. The empirical comparisons vary label granularity, selected residual-stream layer, and sparsity level while keeping the rest of the evaluation pipeline fixed. This design narrows the scope of the conclusions, but it makes the comparisons more interpretable.

The thesis does not aim to compare SAE training algorithms, train new SAEs, evaluate multiple base language models, or provide a full causal account of model behavior. It also does not claim that class alignment is identical to monosemanticity. Instead, class alignment is used as a measurable operational proxy, and the thesis studies both its usefulness and its limitations.

1.6 Thesis approach

To answer these RQs, the thesis uses a two-stage evaluation approach. The first stage measures class alignment in the feature-to-class direction. For each SAE feature, its activation mass is summarized over a predefined label space, producing a feature-level estimate of how concentrated or diffuse its class distribution is. This stage is used to study the measurement problem in RQ1 and the condition-dependence problem in RQ2.

The second stage validates the resulting feature rankings in the reverse direction. Instead of only asking which classes a feature appears to align with, the thesis also asks whether features selected for a target class can recover instances of that class. This stage addresses RQ3 using reverse precision/recall (P/R) validation: features ranked by low normalized Shannon entropy (H) or high Kullback-Leibler divergence (KL) are tested as detectors for their associated classes.

1.7 Contributions

This thesis makes four main contributions. These contributions correspond directly to the three RQs: the first and second contributions address RQ1, the third addresses RQ2, and the fourth addresses RQ3.

First, it frames the evaluation of SAE feature monosemanticity as a measurable problem with explicit limits. Since monosemanticity cannot be directly established from activation statistics alone, the thesis uses class alignment as an operational proxy. This clarifies what the experiments measure, avoids treating class alignment as identical to monosemanticity, and provides a principled basis for interpreting feature-level statistics under explicit label-space assumptions.

Second, it develops a scalable information-theoretic framework for measuring class alignment across full SAE dictionaries. The framework uses normalized Shannon entropy to measure concentration over classes, and KL divergence to compare a feature's class distribution with the empirical class prior. Entropy and KL are not new metrics. The contribution of this thesis is to apply them systematically to SAE feature evaluation. The thesis also clarifies their different roles and shows how prior-aware metrics can separate genuinely class-aligned features from spurious low-entropy features.

Third, it provides a controlled empirical analysis of when this proxy is informative or misleading. With the base model, SAE family, activation site, and feature width fixed, the thesis studies how class alignment changes with label granularity, SAE sparsity, and layer depth. This analysis shows that label granularity is not just an implementation detail. It is a key condition that determines whether entropy-based SAE feature evaluation is meaningful. In particular, the comparison between document-level and token-level label spaces helps identify the boundary conditions under which class alignment can support claims about monosemanticity.

Fourth, it validates the class-alignment signal in the reverse direction. Instead of only scoring features, the thesis tests whether features identified as class-aligned can recover instances of their corresponding classes. Precision and recall are standard tools, so the novelty is not the metrics themselves, but their use as an independent validation step for SAE class-alignment evaluation. This reverse validation helps position entropy and KL as scalable screening tools rather than final proof of monosemanticity, and provides a more reliable basis for selecting candidate features for further interpretability or causal analysis.

Overall, the novelty of this thesis lies in how it combines several components within a controlled evaluation design. Prior work has studied SAE reconstruction, sparsity, automated explanations, causal evaluation, and monosemanticity scores. This thesis takes a different focus. It studies class alignment as a proxy for monosemanticity with clear interpretive limits. It also applies information-theoretic metrics across full SAE dictionaries, analyzes their failure modes, and validates the resulting feature rankings from an independent class-recovery perspective. More broadly, the thesis argues that SAE feature evaluation should be label-space aware. Entropy-based metrics can be useful, but only when they are interpreted together with label granularity, class

imbalance, and reverse validation.

1.8 Thesis outline

The remainder of this thesis is organized as follows. Chapter 2 reviews the literature on mechanistic interpretability, superposition, sparse autoencoders, and current approaches to SAE feature evaluation. Chapter 3 describes the experimental setup, including the selected base model, SAE suite, datasets, layers, sparsity levels, and activation extraction procedure. Chapter 4 presents the methodology, including the class-alignment proxy, the information-theoretic H/KL framework, and the reverse P/R validation design. Chapter 5 reports the empirical results, analyzes how class alignment varies across label granularity, sparsity, and layer depth, and checks validation results. Chapter 6 concludes the thesis, discusses limitations, and outlines directions for future work.

2. Literature Review

The theoretical and empirical basis of this thesis intersects several fast-developing subfields within AI. This chapter places the method proposed in Chapter 1 in context. It reviews the main research areas that have shaped modern dictionary learning. First, it examines the foundation of mechanistic interpretability and highlights the superposition hypothesis as a central challenge to model transparency. Next, it reviews the mathematical foundations and architectural development of SAEs. Finally, the chapter analyzes current paradigms of interpretability evaluation and establishes the theoretical basis for using information-theoretic, entropy-based metrics.

2.1 Foundations of mechanistic interpretability

Efforts to understand the internal workings of neural networks have moved through several stages, shifting from high-level behavioral observations to detailed, causal investigations of intermediate activations [8]. Early and widely used interpretability work often relied on post-hoc methods, such as saliency mapping, attention rollout, and gradient-based feature attribution [1,39,40]. These approaches offered an intuitive view of which input tokens most influenced a specific output. However, they cannot reveal the specific computations, algorithms, or logical structures implemented inside the hidden layers.

Mechanistic interpretability represents an important shift in interpretability research. It is closely related to the linear representation hypothesis. This hypothesis suggests that deep neural networks learn to represent meaningful concepts as discrete, linear directions within their high-dimensional activation spaces [34]. Under this framework, an LLM is not merely a statistical "stochastic parrot" [7]. Instead, it is a system that actively constructs complex, compositional world models [27]. In an ideally interpretable network, every distinct concept would perfectly align with a single orthogonal basis vector. In the simplest neuron-level interpretation, these directions would also align with individual neuron axes [14].

However, empirical studies on both vision and language models show that this ideal one-to-one mapping is exceptionally rare [5, 10]. Instead, empirical analyses in-

dicate that many neurons are polysemantic, meaning they activate in response to a mixture of unrelated inputs. If researchers try to label a single neuron based on its strongest activations, they often find that it responds equally well to completely different concepts [33]. This overlap makes it very difficult to map clear, causal circuits. This polysemantic entanglement makes traditional neuron-level interpretation difficult. It also shows why we need to better understand how networks compress information.

2.2 Superposition and polysemanticity

A main theory for why polysemanticity is so common is the "superposition hypothesis" [14]. This hypothesis addresses a fundamental geometric bottleneck: the model may need to represent more sparse, meaningful features than it has available dimensions. To cope with this limitation, the network does not always use strictly separate directions. Instead, it encodes features as approximately orthogonal vectors in a shared latent space. In high-dimensional spaces, the number of nearly orthogonal vectors is exponentially larger than the number of strictly orthogonal ones. Superposition takes advantage of this mathematical property, enabling the network to simulate a much larger, highly sparse system.

However, superposition comes with an important trade-off [14]. While it increases representational capacity, it also introduces geometric interference. When multiple features share the same lower-dimensional space, activating one feature inevitably creates noise along the directions of others. To manage this, the network uses non-linear activation functions, like Rectified Linear Unit (ReLU), as filters. This compression works particularly well when features are sparse and only a few are active simultaneously. During forward propagation, ReLU can suppress small and noisy activations. It keeps stronger and more relevant signals. This helps the model decode superposed features with less signal loss.

Although superposition is efficient for task optimization, it makes the model's internal representations much harder for humans to interpret [8, 14]. In this setting, the standard computational basis defined by individual neurons does not align with the conceptual basis defined by meaningful features. Furthermore, the geometric structures formed by this superposition often resemble uniform polytopes, such as tetrahedrons or pentagons. These complex formations make it more difficult to analyze or isolate individual concepts using traditional methods.

Recent research has expanded on this issue by introducing the concept of incidental polysemanticity [25]. The research showed that overlapping features can occur purely by chance during random weight initialization. During the training process, gradient descent focuses on reducing loss rather than separating features, so it can

strengthen these early overlaps even when the model has enough neurons to represent features more cleanly. These findings suggest that polysemanticity is not just a capacity-saving strategy, but a natural byproduct of standard gradient descent. This highlights why post-hoc interventions, such as dictionary learning, are essential for interpreting these models.

2.3 Sparse autoencoder architectures

To unpack the superposed representations, many mechanistic interpretability studies now focus on SAEs as an unsupervised form of dictionary learning. An SAE is trained offline using the frozen internal activations extracted from a specific layer of an LLM [10, 16]. Its training objective is to reconstruct the original dense activation vector, and it achieves this by using a linear combination of a much larger set of learned, highly sparse feature directions.

Mathematically, a standard SAE maps a dense activation vector $x \in \mathbb{R}^n$ into an overcomplete latent feature space $f \in \mathbb{R}^M$, where $M > n$, often with $M \gg n$ in practice [16]. The encoder performs this projection using a weight matrix W_{enc} and a bias b_{enc}, followed by a sparsity-inducing activation function σ. Subsequently, the decoder reconstructs the activation vector $\hat{x}$ using a dictionary weight matrix W_{dec} and a bias b_{dec}:

$$f(x) = \sigma(W_{enc}x + b_{enc})$$

$$\hat{x} = W_{dec}f(x) + b_{dec}$$

2.3.1 Standard L_1 SAEs

The foundational SAE architecture uses a standard ReLU activation function and an L_1 regularization penalty applied directly to the latent vector f [16, 19]. The loss function is defined as:

$$\mathcal{L} = \|x - \hat{x}\|_2^2 + \lambda\|f\|_1$$

Although this approach successfully separates early monosemantic features, it is known to introduce two well-known limitations [16, 19, 37]. The first is activation shrinkage. Because the L_1 penalty applies a constant downward pressure on all non-zero activations, baseline SAEs tend to underestimate feature magnitudes. This shrinkage can reduce reconstruction quality, distort the geometry of the learned features, and weaken downstream causal interventions. The second limitation is the occurrence of dead latents, meaning SAE features that stop activating during training. Under strong sparsity pressure, some latents may become permanently inactive, wasting a substantial part of the dictionary's capacity.

2.3.2 Gated SAEs

To reduce activation shrinkage while maintaining differentiable regularization, Gated SAEs introduce a decoupled architecture [37]. Specifically, they separate two key processes: feature detection and magnitude estimation. In this architecture, the sparsity penalty is applied mainly to the gating mechanism, rather than directly shrinking the activation values. This strategy lowers systematic underestimation of feature magnitudes and improves the trade-off between reconstruction quality and sparsity. As a result, Gated SAEs can be viewed as an intermediate step toward later architectures with harder sparsity constraints.

2.3.3 Top-K and Batch Top-K SAEs

To avoid the shrinkage caused by L_1 regularization, Top-K SAEs use a hard sparsity constraint instead of a continuous penalty. In the forward pass, only the top-K pre-activations are retained, and the rest are set to zero [16]. Because the L_1 term is removed, the selected features can better preserve their magnitudes. Furthermore, variants like Batch Top-K SAEs extend this rule across the whole batch, which encourages the model to activate a wider range of dictionary features and helps reduce dead latents [11].

However, the Top-K operator is non-differentiable, so Top-K SAEs typically rely on surrogate gradient methods or auxiliary objectives during training, such as straight-through estimation and auxiliary losses that provide a usable gradient signal [16, 20]. Even with this added complexity, Top-K SAEs offer clear control over the active-feature budget, since the sparsity level can be fixed precisely (e.g., $K = 32$) when studying scaling laws. A remaining limitation is that better performance on the MSE-L_0 frontier does not necessarily mean more monosemantic or interpretable features. Here, L_0 denotes the number of active latent features. Some studies suggest that certain sparsity settings can produce more generic or absorption-like features [13, 32]. This makes careful tuning of K and related hyperparameters important.

2.3.4 JumpReLU and ProLU SAEs

To approximate the ideal L_0-style sparsity without using complex gating networks, another line of studies modified activation functions, especially JumpReLU and Proximal ReLU (ProLU). JumpReLU uses a discontinuous activation function that learns a separate threshold for each latent feature [38]. If the pre-activation is below the threshold, the output is set to zero; if it is above, the output passes through without the continuous shrinkage induced by an L_1 penalty. In contrast, ProLU replaces standard ReLU with

a learnable thresholded activation that aims to improve the sparsity-reconstruction trade-off, although its threshold parameter still requires approximate gradients during training [41].

Empirical results on modern LLMs show that these advanced variants can improve the Pareto trade-off between the L_0 norm and recovered cross-entropy loss [38, 41]. In some reported settings, these variants achieve better reconstruction accuracy at fixed sparsity levels than Top-K or Gated SAEs. A key limitation is that their sparsity is controlled indirectly, so L_0 cannot be set to a specific target value. As a result, fair comparisons often require large hyperparameter sweeps, which increases computational cost.

2.3.5 Advanced SAE architectures and feature absorption

Despite these architectural improvements, feature absorption can still occur in SAEs. Feature absorption occurs when a broad, high-frequency feature captures activations that should belong to more specific features. This can happen because using the broad feature reduces the overall sparsity cost [13]. For instance, when a document mentions both "biology" and "mitochondria," an SAE with strong sparsity pressure may activate only the general "biology" feature and suppress the more detailed concept.

Recent methods such as Matryoshka SAEs and KronSAE aim to mitigate this problem through different mechanisms. Matryoshka SAEs enforce nested dictionaries that preserve higher-level features while allowing finer-grained features to emerge. KronSAE uses a Kronecker-factorized design to improve efficiency while reducing absorption under fixed compute [12, 24]. Empirical results suggest that these approaches reduce feature absorption and improve measures related to disentanglement or interpretability under their respective evaluation settings.

2.4 Evaluation of sparse autoencoders

As SAE architectures have become more advanced, evaluating their learned features has become a separate problem from training them. Because dictionary learning is unsupervised, there is no single ground-truth label for each learned feature. Therefore, existing evaluation methods measure different aspects of feature quality, such as reconstruction fidelity, natural-language interpretability, causal influence, and feature-level monosemanticity scores. These methods are complementary. Therefore, this section reviews the main families of SAE evaluation methods.

2.4.1 Structural proxy metrics

In early SAE evaluation, researchers relied heavily on intrinsic structural proxy metrics, especially the trade-off between reconstruction fidelity and sparsity [19,22]. A common assumption was that if the sparse latent space is well aligned with the dense activation space, the learned features will naturally correspond to interpretable concepts. Under this view, the standard training objective for SAEs is designed to explicitly balance these two goals:

$$\mathcal{L} = \underbrace{||x - \hat{x}(x)||_2^2}_{\text{Reconstruction Error}} + \lambda \underbrace{||f(x)||_1}_{\text{Sparsity Penalty}}$$

To compare performance across different SAE architectures, researchers commonly use the Fraction of Variance Unexplained (FVU) as a main evaluation metric [16,22]. By normalizing the squared reconstruction error against the total variance of the original activations, FVU offers a scale-invariant way to measure information loss. For some time, the dominant goal was to minimize FVU while keeping the L_0 norm low. Related model-level fidelity measures also evaluate whether replacing original activations with SAE reconstructions preserves model behavior, for example, through recovered loss, cross-entropy changes, or divergence between original and reconstructed model outputs.

However, recent benchmarking shows that optimizing only structural proxy metrics does not reliably improve semantic interpretability. It may also be associated with failure modes such as feature absorption [22, 30]. When an SAE is strongly pushed to minimize the L_0 norm, it tends to compress conceptually distinct but related concepts into a single latent feature [13]. This compression minimizes the loss function mathematically but destroys the granularity needed for precise interpretation. Consistent with this, task settings with supervised or ground-truth feature labels show that features optimized primarily for sparsity do not always cleanly distinguish specific concepts from broader categories [13, 30]. Therefore, structural metrics are necessary but not sufficient. They measure whether the SAE preserves information efficiently, but they do not answer whether the learned dictionary decomposes representations into semantically clean features [16].

2.4.2 Automated interpretability methods

After researchers found that strong structural scores can still produce semantically unclear features, the focus shifted toward semantic and functional evaluation of learned features [22]. This shift led to automated interpretability, where advanced LLMs (e.g., GPT-4) act as evaluators and produce explanations for learned features based on the

text snippets that activate them most [35]. The basic idea is to inspect the highest-activating examples for a feature, generate a natural-language hypothesis for what the feature represents, and then test whether this explanation predicts further activations.

Among existing evaluation routes, this approach is relatively close to monosemanticity evaluation because it asks whether a single feature can be described by one coherent human-readable concept [35]. It also has an important advantage over label-based metrics; it can discover semantic patterns that are not visible under a predefined class partition.

However, automated interpretability has important limitations. Single-pass generation often suffers from hallucinations. LLM-generated explanations may assign plausible meanings to a feature even when the underlying representation is fragile. These explanations may also be insufficiently grounded in the actual activation structure [35]. To address this, frameworks like SAE Agentic Explainer treat interpretation as a multi-agent scientific discovery loop, where different agents collaborate to generate, challenge, and refine hypotheses against empirical data [17]. Even with these improvements, automated interpretability remains relatively expensive and is usually applied to selected features rather than to every feature in many SAE dictionaries.

In addition, text-based explanations may miss distributed coding. In a distributed setting, a human-meaningful concept can be predicted from a combination of features. However, no single feature may represent the concept by itself [43]. As a result, a high interpretability score from an LLM can be misleading. It may hide the fact that the concept is still spread across many latent features. In that case, the representation remains entangled, and the goal of dictionary learning is not fully achieved.

2.4.3 Probing-based evaluation

Another evaluation route uses supervised probes to test whether information about a concept can be recovered from model or SAE activations. In probing-based evaluation, a classifier is trained on activation vectors or SAE feature activations to predict a target label [3,6]. High probe accuracy indicates that the representation contains information about the concept and that this information is accessible to the classifier.

The strength of probing is that it provides a clear task-grounded signal. It can show whether a concept is linearly or nonlinearly separable in the representation, and it can be applied to specific downstream concepts of interest. This makes probing useful for studying whether SAEs expose information that is relevant to a labeled task [16]. However, probing does not directly evaluate single-feature monosemanticity. Probe accuracy is usually a property of a representation or a group of features, not of one latent feature alone [6]. Thus, probing is better understood as measuring concept

separability.

2.4.4 Causal disentanglement metrics

To address the ambiguity of correlation-based metrics, recent evaluation work has increasingly moved toward causal disentanglement. The primary goal is to provide stronger causal evidence about whether independent concepts are mapped to distinct, minimally interfering latent dimensions [22, 30]. A common approach in this line of research is zero-ablation, where selected latent features are artificially set to zero, and the resulting changes in model outputs are measured [35].

Two key metrics exemplify this evaluation framework: Targeted Probe Perturbation (TPP) and Spurious Correlation Removal (SCR) [22]. TPP evaluates concept isolation by training class-specific probes and identifying the latents most relevant to a target class. Next, these specific latents are set to zero. If the concept is properly isolated, this ablation will sharply reduce the accuracy of the target probe while leaving unrelated probes unaffected. SCR complements this approach by testing whether zero-ablating a small group of latents can remove a false, spurious concept from a biased probe without harming the true signal. Therefore, SCR tests whether correlated concepts are separated into distinct groups of latents.

Under this rigorous framework, a causally isolated feature should satisfy two requirements: necessity and exclusivity [22]. Necessity requires the feature to cause high divergence on the target concept, while exclusivity requires near-zero impact on unrelated concepts. Together, this process of causal verification helps test whether the SAE is not only capturing correlations, but is localizing concept-relevant causal influence into more separable latent groups.

2.4.5 Feature monosemanticity scores

Although causal and task-grounded evaluations provide useful evidence about whether features are functionally isolated, they are often benchmark-specific and do not always yield a single scalar measure of feature monosemanticity [22]. Recent work has addressed this limitation in complementary ways. SAEBench systematizes task-grounded and intervention-based evaluation across settings, while Feature Monosemanticity Score (FMS) introduces a scalar observational metric based on feature capacity together with local and global disentanglement [18, 22].

FMS operationalizes interpretability by training an interpretable classifier over latent activations, and decomposes monosemanticity into three measurable dimensions: feature capacity, local disentanglement, and global disentanglement [18]. Feature capacity measures whether a single best feature can predict the target concept on its

own. Local disentanglement measures how much performance declines when the primary feature is ablated. Global disentanglement measures whether adding auxiliary features provides little additional information beyond the primary feature. The resulting score measures how closely a learned feature approaches the "one feature, one concept" ideal. Fundamentally, FMS aims to evaluate feature-level monosemanticity in a scalable way, and its setup is concept-centric. It begins with a target concept and evaluates whether a feature cleanly carries that concept under a particular labeled dataset and classifier-based procedure.

2.4.6 Distributional and information-theoretic perspectives

Information-theoretic perspectives provide another way to think about interpretability and feature evaluation. Rather than asking only whether an SAE reconstructs activations well or whether an external model can describe a feature in natural language, these approaches examine how information is organized, compressed, or distributed across representations [4, 15, 16]. In the context of SAE features, this suggests that feature activations can be studied not only as isolated examples, but also as distributions over annotated data.

Distributional metrics are useful because they can summarize whether a feature's activations are spread broadly across many categories or concentrated under a smaller part of the label space [15]. This makes them attractive for scalable evaluation, especially when many features must be compared at once. However, their interpretation is limited by the labels used for evaluation. A feature that is concentrated under one label space is not necessarily monosemantic in a broader semantic sense, and a meaningful feature may appear diffuse if the available labels are too coarse [13].

3. Experimental Setup

This chapter describes the experimental setup used to evaluate SAE features. The aim is to build a controlled experimental environment in which feature-level class alignment can be measured across datasets, layers, and sparsity levels without changing the underlying language model or the evaluation pipeline. Section 3.1 introduces the base language model. Section 3.2 describes the SAEs used in the study. Section 3.3 presents the datasets and label spaces. Section 3.4 explains the activation extraction and preprocessing procedure. Section 3.5 summarizes the experimental design.

3.1 Base language model

All experiments use google/gemma-2-2b as the base language model. Gemma 2 is a family of open language models released by Google DeepMind, covering model sizes from approximately 2 billion to 27 billion parameters [42]. The Gemma 2 technical report describes architectural modifications such as interleaved local-global attention and grouped-query attention, and reports that the 2B and 9B models are trained with knowledge distillation. These properties make Gemma 2 a suitable target for interpretability experiments. It is large enough to contain non-trivial internal representations, while still small enough to allow systematic feature-level analysis on a single high-memory GPU.

The experiments use the pre-trained base model because the goal of this thesis is to evaluate class alignment in a controlled setting. Compared with instruction-tuned variants, the pre-trained model provides a cleaner target for studying the representational structure learned during language-model pre-training. Instruction-tuned variants may introduce additional behavioral constraints and alignment-related representations, which could complicate the interpretation of latent features.

The model is loaded in `bfloat16` precision. This precision is commonly used for efficient inference on modern accelerators and reduces memory consumption without changing the conceptual evaluation setup. The model weights are kept frozen throughout the study. No fine-tuning, probing, or additional supervised training is applied to the language model itself. All measurements are therefore observational with respect

to the base model and the already trained SAEs.

3.2 SAE suite

The SAEs evaluated in this thesis come from the Gemma Scope family. Gemma Scope is an open suite of SAEs trained on internal activations of Gemma 2 models. The technical report introduces Gemma Scope as a set of JumpReLU SAEs trained on all layers and sub-layers of Gemma 2 2B and 9B, and selected layers of Gemma 2 27B [29]. The specific SAE collection used here is `gemma-scope-2b-pt-res`, where `2b-pt` indicates that the SAEs are trained on the Gemma 2 2B pre-trained base model, and `res` indicates that the target activation site is the residual stream.

The residual stream is chosen because it acts as a central communication channel across transformer blocks. The residual stream integrates information from both attention and MLP submodules. Thus, its representations are ideal for studying high-level feature structures. Restricting the analysis to residual-stream SAEs also keeps the comparison focused; differences in the results can be attributed mainly to layer depth or sparsity level.

Each SAE has width 16k, corresponding to $F = 16,384$ latent features. The evaluated configurations cover three transformer layers and five sparsity levels per layer, giving fifteen SAE configurations in total. The hook points are `blocks.5.hook_resid_post`, `blocks.12.hook_resid_post`, and `blocks.19.hook_resid_post`.

The selected layers represent shallow, middle, and deep residual-stream positions. Layer 5 is used as an early-layer representative, layer 12 as a middle-layer representative, and layer 19 as a later-layer representative. This allows the study to examine whether class alignment changes with depth while avoiding an overly broad layer sweep.

The five sparsity levels are represented by the average L_0 values provided for each layer. The complete set is:

Table 3.1: Average L_0 values across different layers and hook points.

Layer	Hook point	Average L_0 values
5	`blocks.5.hook_resid_post`	18, 34, 68, 143, 309
12	`blocks.12.hook_resid_post`	22, 41, 82, 176, 445
19	`blocks.19.hook_resid_post`	23, 40, 73, 137, 279

Here, average L_0 denotes the average number of active SAE features per token. Lower L_0 corresponds to stronger sparsity, meaning fewer features are active for each token, so each active feature is under stronger pressure to represent a narrower direction. Higher L_0 corresponds to a denser representation, where more features can jointly participate in reconstruction. Therefore, comparing these five levels provides a controlled way to study how sparsity affects class alignment and its role as an operational proxy of monosemanticity.

For analysis, the five L_0 settings are grouped into the following tiers:

Table 3.2: Sparsity tiers and their approximate L_0 ranges and interpretations.

Tier	**Approximate range**	**Interpretation**
ultra-sparse	$L_0 < 25$	strongest sparsity
sparse	$25 \leq L_0 < 50$	low activation count
mid	$50 \leq L_0 < 100$	intermediate sparsity
dense	$100 \leq L_0 < 200$	high activation count
very-dense	$L_0 \geq 200$	weakest sparsity

This design yields a 3×5 grid of layer and sparsity settings. Since the base model, activation site, SAE width, and evaluation pipeline are fixed, the resulting comparisons isolate two main factors: layer depth and sparsity strength. It should be noted that the sparsity tiers are used as descriptive analysis groups rather than theoretically sharp categories. Their purpose is to make later comparisons easier to interpret by summarizing trends across L_0 levels while still preserving the exact average L_0 in Table 3.1.

3.3 Datasets and label spaces

This study uses three datasets to cover two levels of label granularity: document-level topic labels and token-level entity labels. This distinction is crucial to the thesis because the proposed information-theoretic metrics measure how a feature's activation mass is distributed across a predefined label space. If the label space is too coarse, a feature may be semantically coherent but not concentrated under the available labels. Conversely, a finer-grained token-level label space may expose class-aligned features more clearly.

The datasets are summarized in Table 3.3.

Table 3.3: Dataset information used in the experiments.

Dataset	Granularity	Number of classes	Split and sample use	Role in the study
`fancyzhx/ag_news`	document-level	4	all test samples	coarse topic-level baseline
`dbpedia14`	document-level	14	all test samples	finer document-level baseline
`ai4privacy/pii-masking-300k`	token-level	25 / 26	first 10,000 validation samples	main token-level experiment

3.3.1 AG News

AG News is a topic classification dataset derived from a large news corpus [45]. It contains four classes: World, Sports, Business, and Sci/Tech. Each class contains 30,000 training samples and 1,900 test samples, giving 120,000 training samples and 7,600 test samples in total. In this thesis, all test samples are used. AG News serves as a coarse document-level baseline. This makes it useful for testing whether entropy-based class-alignment metrics can separate SAE features under a low-resolution label space.

3.3.2 DBpedia14

DBpedia14 is a document classification dataset constructed from DBpedia ontology classes [45]. It contains fourteen non-overlapping classes, with 40,000 training samples and 5,000 test samples per class, giving 560,000 training samples and 70,000 test samples in total. In this thesis, all test samples are used. DBpedia14 provides a second document-level baseline, but with more classes than AG News. Thus, it allows the analysis to distinguish between two possible causes of weak class-alignment signals: coarse document-level semantics and small class count. If DBpedia14 still produces high-entropy distributions despite having fourteen classes, this would support the interpretation that document-level granularity itself limits the metric's resolution.

3.3.3 PII-masking-300k

The main dataset is PII-masking-300k, a token-level privacy-masking dataset designed for training and evaluating models that detect personally identifiable information (PII)

and other sensitive information in text [2]. The dataset card describes it as an open dataset for privacy masking, with more than 220,000 examples, 30.4 million text tokens, and 7.6 million PII tokens.

This dataset is used because its labels correspond to token-level entity types, making it directly suitable for class alignment evaluation. Compared with document-level topics, token-level PII classes provide a more direct setting for testing whether SAE features align with specific semantic or format-based categories, such as email addresses, names, phone numbers, dates, or locations.

The original labels include a non-entity class `O`. Including this background class gives 26 classes. However, most main experiments use a no-`O` variant, denoted as PII-noO, which removes the background class and leaves 25 PII entity classes. This is an evaluation choice designed to prevent the dominant non-entity class from overwhelming the class distribution. The consequences of including `O` are analyzed explicitly in Chapter 5.

For token-level experiments, the first 10,000 samples from the validation split are used with a context length of 128 tokens. This sample size is large enough to provide stable activation statistics over many token instances while keeping the computational cost manageable. In the validation analysis of experiments, the `CARDISSUER` class is excluded from macro-averaged results because it has fewer than 20 token instances in the validation subset and therefore does not support stable recall estimation. As a result, the validation results are macro-averaged over 24 classes.

3.3.4 Ethical considerations of datasets

This study uses publicly available benchmark datasets, including AG News, DBpedia14, and PII-masking-300k. The main ethical consideration comes from PII-masking-300k, because it contains labels for personally identifiable information. However, the dataset is publicly available and mainly relies on synthetic or heavily processed text, rather than real, unauthorized private records. Its use in this thesis follows the access conditions of the original dataset.

The dataset is used only for observational feature-level analysis. The base language model and SAE weights remain unchanged, and no text generation is triggered. The experiments do not attempt to identify individuals, recover private information, or redistribute sensitive data. Since the analysis is conducted at an aggregate statistical level, it does not create new personal records or expose individual-level information. Therefore, the ethical risk of using this dataset is limited.

3.4 Activation extraction and feature statistics

For each dataset and SAE configuration, the base model is run on the input text, and residual-stream activations are extracted at the SAE's corresponding hook point. The SAE encoder then maps each residual activation vector into a sparse feature activation vector of dimension $F = 16,384$. Let $a_f(t) \geq 0$ denote the activation value of SAE feature f on evaluation unit t. For document-level datasets, t refers to a document-level unit after the relevant aggregation step (exact aggregation rules are defined in Chapter 4). For token-level PII experiments, t refers to a token with a valid token-level label.

A feature is considered active on t if $a_f(t) > 0$. This criterion is appropriate for the JumpReLU SAE family used in the experiments because its activation distribution contains exact zero activations and clearly separated nonzero activations. Since this property is architecture-dependent, it should be rechecked before applying the same thresholding rule to other SAE families.

All feature-level statistics are computed in a streaming manner. Rather than storing the full token-by-feature activation matrix for the entire dataset, the implementation accumulates class-conditioned activation sums as the data is processed. This makes the computation feasible on a single A100 40GB GPU. The main memory cost arises from accumulating token-level activation statistics for 16k features over the PII validation subset.

3.5 Experimental design

The experimental design is organized around two complementary stages.

The first stage measures class alignment directly using information-theoretic metrics. For each feature, its activation mass is distributed over the label classes, producing a class-conditional distribution. Normalized Shannon entropy H measures how concentrated this distribution is, and KL divergence measures how far it deviates from the empirical class prior. This stage is observational and feature-to-class in direction: it asks which classes account for a feature's activation mass.

The second stage validates these metrics in the reverse direction using P/R. Instead of asking which class a feature belongs to, it asks whether the top-ranked features for a class can recover instances of that class. This stage is class-to-feature in direction and provides an external validation signal for the information-theoretic metrics.

The overall design therefore avoids relying on a single metric family. Entropy and KL provide scalable feature-level screening, while P/R validation tests whether the selected features can function as class-aligned detectors. This division is important because class alignment is treated in this thesis as an operational proxy for monose-

manticity, not as a complete definition of it. A low-entropy feature is not automatically assumed to be truly monosemantic; it must also be evaluated against low-frequency artifacts, high-frequency baselines, and random feature selection.

The scope of the experiments is intentionally controlled. The study does not compare different base language models, different SAE training algorithms, or different activation sites. Instead, it fixes the base model, SAE family, activation site, and feature width, varying only layer depth, sparsity level, and label granularity. This makes the empirical conclusions narrower but more interpretable.

4. Methodology and Implementation

This chapter defines the evaluation framework used to measure and validate SAE feature monosemanticity. The central methodological problem is that monosemanticity is not directly observable from activations alone. A feature may appear interpretable in a small number of examples, but fail to generalize across the dataset. Conversely, a feature may be stable, but not aligned with the label space used for evaluation. To handle this, the thesis evaluates a measurable proxy: class alignment, defined as the concentration of a feature's activation distribution over a predefined class partition.

Section 4.1 introduces the information-theoretic framework based on normalized Shannon entropy and KL divergence. Section 4.2 defines the reverse P/R validation framework. Section 4.3 specifies implementation choices and filtering rules used in the experiments.

4.1 Information-theoretic framework

4.1.1 Motivation: From monosemanticity to class alignment

The core question in evaluating whether an SAE feature is monosemantic is whether its activations are concentrated primarily on a single semantic concept, or at most a very small number of concepts. In this framework, these semantic concepts are approximated by dataset labels. For example, a feature that activates consistently only on tokens of the "email address" class can be regarded as highly class-aligned and plausibly monosemantic under this label space. By contrast, a feature that activates on email addresses, person names, cities, and unrelated tokens is closer to a polysemantic feature.

Strictly speaking, this framework does not measure monosemanticity directly. It measures how concentrated a feature's activations are under a predefined class partition. This thesis refers to that quantity as class alignment. Class alignment is useful evidence for monosemanticity, but it is not sufficient on its own. A truly monoseman-

tic feature should show stable behavior under meaningful class partitions. However, a feature that aligns with one label may still mix several subconcepts within that label.

The evaluation framework therefore treats class alignment as an operational proxy. This proxy is useful because it is scalable, quantitative, and externally checkable. However, it must be interpreted carefully. Low entropy does not by itself prove that a feature has a single human-interpretable meaning. It only shows that the feature's activation mass is concentrated under the chosen label space. For this reason, the information-theoretic metrics are later validated with a reverse P/R framework.

The basic intuition is simple: if a feature's activations are concentrated in one class, it is more class-aligned; if its activations are spread across many classes, it is less class-aligned. Shannon entropy provides a natural way to formalize this intuition because it measures the uncertainty of a probability distribution. A distribution concentrated on one class has low entropy; a distribution spread uniformly across classes has high entropy. KL divergence is then introduced as an auxiliary metric to account for class imbalance, especially when a dominant background class is present.

4.1.2 Symbols and notation

Let F denote the total number of SAE features and let C denote the number of classes in the evaluation label space. Let N be the total number of valid evaluation items.

Depending on the dataset, an item may be a document or a token. For each feature f and item t, let $a_f(t) \geq 0$ be the activation value of feature f on item t. Let $y(t) \in \{0, 1, \ldots, C-1\} \cup \{-1\}$ be the class label of item t, where $y(t) = -1$ indicates that item t is excluded from the statistics.

Based on this notation, define the accumulated activation mass of feature f on class c as

$$S_f(c) = \sum_{t:\, y(t)=c} a_f(t),$$

that is, the sum of the activation values of feature f over all evaluation items in class c.

We further define

$$T_f = \sum_{c=0}^{C-1} S_f(c) = \sum_{t:\, y(t)\geq 0} a_f(t),$$

which represents the total activation mass of feature f across all labeled evaluation items.

$S_f(c)$ measures how much activation mass feature f accumulates on class c, while T_f gives its total activation mass. All subsequent metrics are built on these two quantities. In other words, the framework is not concerned with how many evaluation items a feature is active on, but with how its total activation mass is distributed across classes.

4.1.3 Entropy and KL divergence

Based on $S_f(c)$ and T_f defined in Subsection 4.1.2, we first normalize feature activations into a class distribution, and then construct two feature-level metrics. Normalized Shannon entropy H serves as the primary metric and KL divergence serves as an auxiliary metric.

The first step is to normalize the accumulated activations in Subsection 4.1.2 into a probability distribution. We define the activation-weighted conditional class distribution of feature f as

$$P(c \mid f) = \frac{S_f(c)}{T_f}.$$

This quantity measures how much of feature f's total activation mass belongs to class c. It is not a binary measure of whether the feature is active.

Here, features with $T_f = 0$ over the evaluation set are excluded because their class-conditional activation distributions are undefined. $P(c \mid f)$ satisfies the basic properties of a probability distribution: $\sum_c P(c \mid f) = 1$ and $P(c \mid f) \geq 0$. It describes what fraction of all activations of feature f fall on class c. Thus, $P(c \mid f)$ turns feature activation into a probability distribution over the class set, which can then be characterized directly by information-theoretic metrics. The central intuition from Subsection 4.1.1 is that the more a feature's activations are concentrated on a small number of classes, the more monosemantic it is. Shannon entropy captures exactly this intuition. We define the normalized entropy of feature f as

$$H(f) = \frac{-\sum_{c=0}^{C-1} P(c \mid f) \log_2 P(c \mid f)}{\log_2 C} \in [0, 1],$$

with the convention that $0 \log 0 = 0$. The denominator $\log_2 C$ is the maximum entropy of the uniform distribution over C classes; it normalizes H into the interval $[0, 1]$, enabling cross-dataset comparison when the number of classes differs. When $P(c \mid f)$ is fully concentrated on a single class, $H(f) = 0$; when $P(c \mid f)$ is uniform over all classes, $H(f) = 1$. Therefore, the smaller $H(f)$ is, the stronger the feature's class alignment under the current class partition. Under the proxy assumption introduced in Subsection 4.1.1, this provides evidence that the feature may be closer to being monosemantic.

H is the primary metric in this work because it directly addresses the main measurement question: is a feature's activation mass concentrated on only a few classes? It depends only on the feature's own class distribution and does not require the prior distribution. However, the limitation of H follows directly from its strength; it only measures how concentrated a distribution is, but not why it is concentrated or where it is concentrated. For datasets whose class distributions are approximately balanced,

this is usually not a problem. Conversely, if the datasets have severe class imbalance, H will introduce a systematic bias.

Consider a typical named-entity setting: 90% of tokens belong to the non-entity class, while only 10% belong to all entity classes combined. If a feature places 99% of its activations on the non-entity class, its H can still be very low, making it look like a "highly monosemantic" feature. In reality, however, it may simply be tracking the most common background class in the data, rather than encoding a genuinely discriminative concept.

This shows a limitation of H. It cannot distinguish between true class alignment and simple agreement with the data prior. The latter produces a feature that appears monosemantic on the surface but in fact provides almost no additional discriminative power with respect to the classes. We refer to this phenomenon as spurious low entropy.

To identify such spurious low-entropy cases, we need a reference distribution for $P(c \mid f)$. The natural reference is not the uniform distribution, but the class prior of the data itself:

$$Q(c) = \frac{N_c}{N}, \qquad N_c = \#\{t : y(t) = c\}.$$

Here we use the empirical class distribution of the data as the prior, rather than the uniform prior $Q(c) = 1/C$. The reason is that real datasets are usually class-imbalanced. If the uniform distribution were used as the baseline, then any feature deviating from uniformity could be misinterpreted as having additional class-discriminative power, including background features that merely track the data prior. Under the empirical class prior, by contrast, $\mathrm{KL}(f) = 0$ corresponds exactly to the case where the feature's class distribution is identical to the data prior, and therefore contributes no additional discriminative information about the classes.

On this basis, we define the KL divergence of feature f relative to the class prior as

$$\mathrm{KL}(f) = \sum_{c=0}^{C-1} P(c \mid f) \log_2 \frac{P(c \mid f)}{Q(c)}.$$

Returning to the example above: if a feature merely follows the background `O` distribution, then its $P(c \mid f)$ will be very close to $Q(c)$, and hence $\mathrm{KL}(f) \approx 0$. If another low-H feature instead concentrates its activations on a rare entity class, then $P(c \mid f)$ will differ substantially from $Q(c)$, and $\mathrm{KL}(f)$ will be correspondingly larger. The former has low entropy but stays close to the empirical class prior. The latter is more likely to be genuinely class-discriminative. Under the proxy interpretation, it is also closer to monosemanticity.

When the class distribution is perfectly balanced, i.e. $Q(c) = 1/C$, we have

$$\mathrm{KL}(f) = \log_2 C \cdot (1 - H(f)).$$

In this case, H and KL carry the same information. When class distributions are approximately balanced, they remain approximately equivalent, so H alone is sufficient. Under strong class imbalance, however, the two decouple: H still measures the absolute concentration of the distribution, whereas KL measures its deviation from the data prior, and can therefore identify the spurious low entropy cases described above. In summary, we use H as the primary metric to quantify how concentrated a feature's activations are across classes, and KL as the auxiliary metric to identify spurious low entropy cases under class imbalance.

4.2 Reverse precision/recall validation framework

4.2.1 Motivation: From feature-to-class to class-to-feature

The information-theoretic metrics above operate in the feature-to-class direction. They ask how concentrated a feature's activation mass is across classes. This direction is useful for evaluating monosemanticity, but this is insufficient to prove the metrics' effectiveness. A feature may receive a very low entropy score simply because it activates only once or twice. Those rare activations may happen to fall in a single class. Such a feature is not useful as a class detector, even though its class distribution appears pure. To address this problem, the thesis introduces a reverse validation direction, class-to-feature. Instead of asking which class accounts for a feature's activation mass, the reverse framework asks how many instances of a target class are covered by the top-k features selected for that class.

This corresponds directly to recall, which is the core evidence for whether H/KL is genuinely useful. If the features selected by low H / high KL are indeed more monosemantic, then they should activate stably on instances of the corresponding class and achieve higher coverage.

Precision and recall are complementary here. Recall measures coverage, while precision measures whether those features truly concentrate their activations on the target class. Ideally, a highly class-aligned feature should achieve both high recall and high precision. By contrast, a feature whose class alignment is spuriously inflated because it activates on only a few target-class instances will usually show low recall.

Therefore, the validation framework tests two claims. First, features ranked highly by low H or high KL for a target class should outperform random features from the same class candidate pool. Second, they should not merely behave like high-frequency features. If their performance can be explained by activation frequency alone, then entropy and KL would not provide a distinct class-alignment signal.

4.2.2 P/R evaluation framework

Let $\mathcal{F}$ denote the set of features participating in evaluation, and $\mathcal{C}$ the class space. For each feature $f \in \mathcal{F}$, we first assign it to a unique main class using the existing $P(c \mid f)$ from Subsection 4.1.3:

$$c_f = \arg\max_{c \in \mathcal{C}} P(c \mid f) = \arg\max_{c \in \mathcal{C}} \frac{\sum_{t:y_t=c} a_f(t)}{\sum_{t:y_t \in \mathcal{C}} a_f(t)}.$$

Here $a_f(t)$ is the activation value of feature f on token t, and y_t is the class label of that token. This assignment is based on activation mass rather than activation frequency. For each evaluation class c, we first collect the candidate features whose main class is c. For each ranking group g, we then select the top-k features from this pool according to its score function s_g:

$$\mathcal{T}_{g,c,k} = \mathrm{TopK}_{f \in \mathcal{F}: c_f = c}(s_g(f), k).$$

where s_g is the score used by ranking group g, such as KL, H, density, or the MI-like score defined below. The detailed ranking groups are defined in the following subsection.

This defines the token-level frequency-based precision and recall. We view the k features in $\mathcal{T}_{g,c,k}$ as an OR-union detector for class c: a token t is predicted as class c if and only if at least one feature in the set activates on t. Let $\mathbb{1}_{\mathrm{hit}}(t) = \mathbb{1}[\exists f \in \mathcal{T}_{g,c,k} : a_f(t) > 0]$, and let $N_c = \sum_{t:y_t \in \mathcal{C}} \mathbb{1}[y_t = c]$ be the total number of tokens of class c. Then:

$$\begin{aligned}
\mathrm{TP}(c) &= \sum_{t:y_t \in \mathcal{C}} \mathbb{1}_{\mathrm{hit}}(t)\mathbb{1}[y_t = c], \\
\mathrm{FP}(c) &= \sum_{t:y_t \in \mathcal{C}} \mathbb{1}_{\mathrm{hit}}(t)\mathbb{1}[y_t \neq c], \\
\mathrm{FN}(c) &= N_c - \mathrm{TP}(c).
\end{aligned}$$

$$P_{\mathrm{tok}}(g,c,k) = \frac{\mathrm{TP}(c)}{\mathrm{TP}(c) + \mathrm{FP}(c)}, \quad R_{\mathrm{tok}}(g,c,k) = \frac{\mathrm{TP}(c)}{\mathrm{TP}(c) + \mathrm{FN}(c)}.$$

In addition to frequency-based precision, we also define amplitude-weighted precision. Let the summed activation over the k features on token t be $w(t) = \sum_{f \in \mathcal{T}_{g,c,k}} a_f(t)$. Then:

$$P_{\mathrm{amp}}(g,c,k) \;=\; \frac{\sum_{t:y_t \in \mathcal{C}} w(t)\, \mathbb{1}[y_t = c]}{\sum_{t:y_t \in \mathcal{C}} w(t)}.$$

For token-level entity datasets such as PII-masking-300k, span-level evaluation is also necessary because a single entity may consist of multiple contiguous tokens. We merge each contiguous sequence of tokens with the same label into a span instance

(indexed by s, with class y_s). A span is counted as a hit if any token inside it is hit by the OR-union detector. Let $\mathbb{1}^{\text{spn}}_{\text{hit}}(s) = \mathbb{1}[\exists t \in s : \mathbb{1}_{\text{hit}}(t) = 1]$, and let $N^{\text{spn}}_c = \sum_{s:y_s \in \mathcal{C}} \mathbb{1}[y_s = c]$ be the total number of spans of class c. Then:

$$\begin{aligned}
\text{TP}_{\text{spn}}(c) &= \sum_{s:y_s \in \mathcal{C}} \mathbb{1}^{\text{spn}}_{\text{hit}}(s)\mathbb{1}[y_s = c], \\
\text{FP}_{\text{spn}}(c) &= \sum_{s:y_s \in \mathcal{C}} \mathbb{1}^{\text{spn}}_{\text{hit}}(s)\mathbb{1}[y_s \neq c], \\
\text{FN}_{\text{spn}}(c) &= N^{\text{spn}}_c - \text{TP}_{\text{spn}}(c).
\end{aligned}$$

$$P_{\text{spn}}(g, c, k) = \frac{\text{TP}_{\text{spn}}(c)}{\text{TP}_{\text{spn}}(c) + \text{FP}_{\text{spn}}(c)}, \quad R_{\text{spn}}(g, c, k) = \frac{\text{TP}_{\text{spn}}(c)}{N^{\text{spn}}_c}.$$

Finally, for each ranking group g and feature budget k, we compute macro-averaged precision and recall over the evaluated classes.

4.2.3 Candidate pools and ranking groups

Subsection 4.2.2 introduced the formal P/R definitions. The next question is how the candidate pool for each class is formed and how candidate features are ranked. This subsection focuses on the construction of candidate pools, ranking groups, and baselines.

We use six ranking groups. Here, `density` denotes the activation density of a feature on the evaluation set, defined as the fraction of labeled evaluation items on which $a_f(t) > 0$. The `h / kl` groups test the effectiveness of H and KL as ranking signals; `h_f / kl_f` test their more robust versions with a density floor; `density` is a pure activation-frequency baseline; and `mi` denotes a mutual-information-like score, computed as density $\times$ KL, which serves as a frequency-adjusted control baseline. In addition, a random baseline is constructed separately within the same candidate pool in order to test whether these rankings actually outperform random sampling.

All ranking groups share the same candidate pool after dead features (defined in Subsection 4.3.1) are removed. The P/R stage does not impose an upper density bound, because `density` and `mi` themselves are control baselines defined in terms of density. Excluding high-frequency features beforehand would destroy the comparability of those baselines and would no longer answer the central question of whether H/KL outperform simple frequency-based ranking. Before ranking, only dead features are removed; high-frequency generic features remain in the pool and are scored by each ranking rule. On top of this shared candidate pool, each group selects top-k according to its own score function:

Table 4.1: Ranking groups

Group	Ranking rule	Additional density floor	Role
`kl`	KL descending	none	raw KL ranking - deviation from the empirical class prior
`h`	H ascending	none	raw H ranking - concentration of activation mass only
`density`	density descending	none	frequency ranking - tests whether performance is explained by frequent activation alone
`mi`	density × KL descending	none	MI-like ranking – a tradeoff between frequency and deviation
`kl_f`	KL descending	0.001	KL ranking + removal of extremely low-frequency noise
`h_f`	H ascending	0.001	H ranking + the same low-frequency floor

The "Additional density floor" column denotes an extra constraint applied after constructing the shared candidate pool. `kl_f / h_f` require `density` ≥ 0.001, while the raw groups `h` and `kl` impose no density constraint and rank directly over the entire candidate pool. For H, lower values correspond to more concentrated class distributions, and thus are ranked in ascending order.

The `density` and `mi` baselines are counterfactual controls for testing whether the metrics can select truly class-aligned features. If activation frequency alone explains class-recovery performance, then `density` should perform as well as or better than KL-based rankings. The density floor in `kl_f / h_f` is used to prevent ultra-rare features from entering the top-k list by accident. Features with only one or two activations are not reliable under P/R evaluation, even if they receive favorable H or KL scores. The floor is therefore best understood as a noise-control threshold rather than a core methodological constraint. The paired presence of raw `kl / h` and floored `kl_f / h_f` is precisely intended to isolate the effect of this floor.

Accompanying these ranking groups is the random baseline. For each class c and

each k, we uniformly sample k features from the main-class candidate pool $\{f : c_f = c\}$ of that class, and average over multiple repetitions. The random baseline uses the same candidate pool as $\mathcal{T}_{g,c,k}$. Thus, differences between the random baseline and the other groups mainly reflect whether score-based ranking is used. This design avoids confounds from candidate-pool size or class sample size. If the performance of a ranking group is close to the random baseline, then the ranking itself is not providing an effective signal.

In the experiments, k ranges over $\{1, 5, 10, 20\}$, covering scales from the single-feature regime to multi-feature combinations. Together, these settings define the candidate pools, ranking rules, and feature budgets used in the reverse P/R validation.

4.3 Implementation details

4.3.1 Implementation of H/KL computation

This subsection discusses how the H/KL framework in Section 4.1 is instantiated and implemented in this thesis. The relevant details fall into three aspects: the choice of label granularity, the choice of aggregation scheme, and the concrete computation and filtering procedure.

First, at the level of label granularity, the H/KL framework can be defined either at the document level or at the token level; the main difference is whether $S_f(c)$ is accumulated over documents or over tokens. In this work, the token-level setting is the primary instantiation. The direct reason is that label granularity determines the effective resolution of the metric. On document-level datasets such as AG News and DBpedia14, one document often contains multiple concepts. Therefore, a feature rarely aligns cleanly with one document class. As a result, the H distribution is compressed into the high-entropy region. By contrast, token-level labels refine the class units into PII entity types and provide a finer-grained label space, making low-H, highly class-aligned features easier to observe. For PII-masking-300k, the main results in this work use PII-noO. This does not change the definition of H/KL itself; rather, it prevents the dominant background class from obscuring the interpretation of the metrics in the main experiment.

Second, in terms of aggregation, we use activation-mass statistics rather than activation-count statistics. In other words, $S_f(c)$ and T_f accumulate activation mass rather than the number of active items. The reason is that, in JumpReLU SAEs, activation magnitude usually reflects how well the feature matches the input; strong activations are more representative of the semantic content encoded by the feature. If activation counts were used instead, all nonzero activations would be treated equally,

and weak activations would receive the same weight as strong ones, diluting the semantic signal.

Finally, in the concrete computation procedure, all metrics are computed only on alive features, meaning features whose total accumulated activation mass exceeds the dead-feature threshold. If the total accumulated activation mass T_f of a feature across all labeled evaluation items is below a threshold (set to 10^{-5} in this thesis), the feature is treated as dead and removed from the H/KL statistics, to avoid numerical instability caused by an excessively small denominator. Excluded features are marked as invalid in the output and do not participate in later aggregation. This dead-feature filtering should be distinguished from the density floor used in the ranking groups. The density floor is applied on top of the alive-feature set and belongs to the candidate-selection procedure rather than to the H/KL metric definition itself. For the H/KL aggregation statistics, we keep only features with density $\leq 10^{-2}$ to exclude syntactic or generic high-frequency features from the aggregated means; otherwise, these high-density generic features could dominate the aggregate statistics and make the mean H less informative.

In implementation, the computation only requires accumulating feature activation mass for each class to obtain $S_f(c)$ and T_f, normalizing them into $P(c \mid f)$, and then computing H, the class prior $Q(c)$, and KL. The empirical prior Q(c) is clipped with a small positive floor before computing KL to avoid numerical instability when zero-frequency classes appear. The entire procedure can be implemented in streaming form, with worst-case time complexity $O(N \cdot F)$ and no need to train any additional model.

4.3.2 Implementation of P/R validation

Subsection 4.2.2 introduced the formal P/R definitions, and Subsection 4.2.3 explained how candidate pools and ranking groups are constructed. In this subsection, we further specify three evaluation settings: feature combination, recall unit, and precision computation.

Feature combination. We combine the top-k features using an OR-union rule: if any one of them activates on an item, the item is counted as a positive prediction for the class. This allows us to directly test whether multiple candidate features for the same class improve coverage when combined, without introducing additional ensemble mechanisms such as AND rules or voting schemes. $k = 1$ corresponds to a single-feature detector, while $k = 20$ corresponds to a more permissive multi-feature detector.

Recall unit. For token-level entity datasets, since a single entity often spans multiple tokens, token-level recall underestimates features that reliably hit only part of an entity. Span-level recall is therefore closer to the question of whether an entity

instance was found, whereas token-level recall more strictly evaluates coverage of the tokens inside each instance.

Precision computation. Each top-k feature set gives rise to two versions of precision: frequency-based precision, which measures the fraction of hit items that belong to the target class, and amplitude-weighted precision, which weights hit items by their summed activation magnitude. We use amplitude-weighted precision as the primary version, because FP activations are typically systematically weaker than TP activations, making frequency precision prone to underestimating the class-aligned behavior of the feature group. At the same time, recall remains a strictly non-circular external validation signal, so we always place recall first and treat amplitude-weighted precision as auxiliary interpretation.

Under these settings, the analysis focuses on two comparisons. First, it compares the recall of `h_f / kl_f` with the random baseline. This tests whether H/KL rankings provide an effective signal. Second, it compares the amplitude-weighted precision of `h_f / kl_f` with the `density` and `mi` baselines. This tests whether their performance is simply caused by selecting high-frequency features.

5. Results and Analysis

This chapter presents the empirical results. Section 5.1 addresses RQ1 and RQ2 by analyzing the information-theoretic metrics H and KL across datasets, sparsity levels, layers, feature density, and the presence or absence of the background `O` class. Section 5.2 addresses RQ3 by evaluating whether the H/KL metrics are supported by P/R validation on the token-level PII task.

The chapter follows a two-step logic. First, for RQ1 and RQ2, H/KL results show where H has sufficient dynamic range and how different factors influence class alignment. Second, for RQ3, P/R results test whether features selected by low H and high KL can function as useful class-aligned detectors rather than artifacts of low frequency, high frequency, or random selection.

5.1 Information-theoretic results

5.1.1 Label granularity and the dynamic range of H/KL

The values of H lie in $[0, 1]$, but whether the H distribution has a meaningful dynamic range depends on label granularity. To study this, we evaluate the same set of 15 SAE configurations on AG News, DBpedia14, and PII-masking-300k, and compare how the H/KL distributions differ under different label granularities. The aggregated results are shown in Table 5.1.

Table 5.1: Aggregated statistics across the three datasets, averaged over the 15 SAE configurations after dead-feature filtering.

Dataset	**# Classes**	**Label Granularity**	**Mean H**	**Median H**	**Mean KL**	**frac($H < 0.3$)**	**frac($H < 0.5$)**
ag_news	4	document	0.8843	0.9748	0.2313	3.7%	7.3%
dbpedia14	14	document	0.8451	0.9338	0.5908	4.2%	8.4%
PII-noO	**25**	**token**	**0.4782**	**0.4996**	**2.4167**	**23.5%**	**49.0%**

Table 5.1 summarizes the central tendency and threshold fractions of the H/KL distributions, while Figure 5.1 visualizes the full distribution of H. As the figure shows, both document-level panels have most of their mass near $H = 1$. This creates a clear pile-up at the high-entropy end. In contrast, the token-level PII-noO distribution is much more spread out. It also develops a clear left tail in the low-H region.

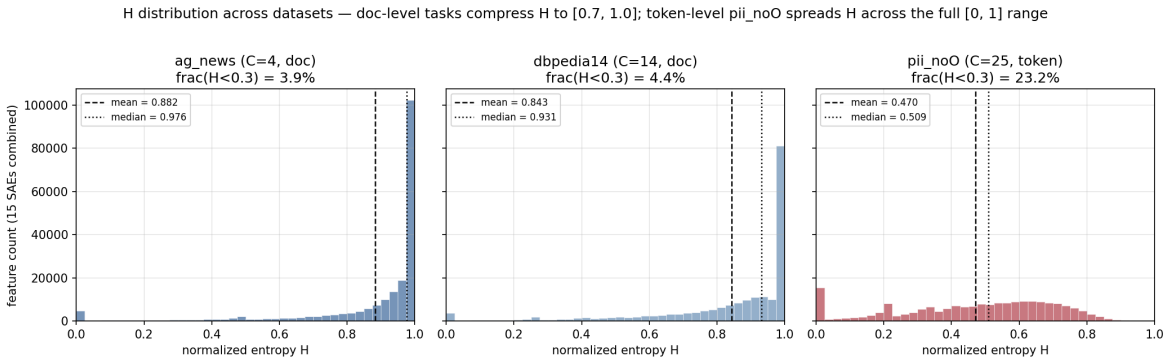


Figure 5.1: Distribution of normalized Shannon entropy H across the three datasets. Dashed lines indicate means and dotted lines indicate medians. On the document-level tasks (AG News and DBpedia14), the H distributions are strongly left-skewed and compressed into the high-H regime; under token-level PII-noO, the distribution opens up substantially, with much more mass appearing in the low-H region.

Table 5.1 and Figure 5.1 show that the median H exceeds 0.93 on both document-level datasets. More than half of the alive features have class distributions that are close to uniform on these two datasets, while the fraction of strongly class-aligned features under the $H < 0.3$ threshold is only 3.7% and 4.2%. In other words, under the document-level setting, the dynamic range of H is compressed into the narrow interval $[0.7, 1.0]$, and most features appear weakly class-aligned under the metric.

The situation changes substantially under token-level PII-noO. The mean H drops to 0.48, the median H drops to 0.50, the fraction of strongly class-aligned features rises from roughly 4% to 23.5%, and nearly half of all features satisfy H<0.5. KL changes in the opposite direction: the mean KL rises from 0.23 on AG News to 2.42 on PII-noO, an increase of about 10×. In other words, the dynamic range of H and KL opens up much more clearly on the fine-grained token-level tasks.

These differences show that the usefulness of the measurement depends on whether the label space matches the granularity of the feature patterns being measured. Document-level tasks contain only 4-14 coarse classes, and a single news article often covers multiple themes, so an individual feature is naturally more likely to spread over multiple document classes. Once the labels are refined into 25 token-level PII entity classes, the case in which one feature activates primarily for one entity class becomes much closer to being statistically observable, and class alignment signals become easier

for H to capture. Thus, the cross-dataset comparison mainly reflects the effect of label granularity on the dynamic range of H. It should not be interpreted as evidence that the same SAEs are inherently more interpretable on one dataset than on another.

For this reason, the remaining analysis focuses on PII-noO as the main evaluation setting, because this is where the dynamic range of H/KL is most clearly opened up. AG News and DBpedia14 serve as lower-resolution comparison settings, helping distinguish phenomena driven by SAE behavior from those driven by label granularity.

5.1.2 Effect of sparsity

Having established PII-noO as the main evaluation setting, this subsection turns to the effect of SAE sparsity on feature-level class alignment. We first aggregate over the 15 SAE configurations by average L_0 tier in order to isolate the main effect of sparsity.

Table 5.2: Aggregated H/KL statistics by average L_0 tier on PII-noO, with feature pools from the three layers combined within each tier.

Average L_0 tier	**Mean H**	**Mean KL**	**frac($H < 0.1$)**	**frac($H < 0.3$)**	**frac($H < 0.5$)**
ultra-sparse (< 25)	0.352	3.134	20.1%	40.1%	69.0%
sparse (25–50)	0.419	2.798	12.8%	29.4%	57.7%
mid (50–100)	0.481	2.448	7.1%	20.6%	47.0%
dense (100–200)	0.537	2.095	4.0%	14.3%	37.1%
very-dense (≥ 200)	0.556	1.890	3.4%	12.6%	33.4%

Table 5.2 shows a clear trend. As the SAE moves from ultra-sparse to very-dense, mean H increases monotonically, while mean KL decreases monotonically. This suggests that denser SAE settings tend to produce weaker class alignment overall, whereas sparser settings tend to produce stronger class alignment. This trend is monotonic across all five L_0 tiers for both mean H and mean KL.

More informative than the means themselves is the behavior of the low-H tail. As L_0 increases, the fraction of features below all three thresholds, $H < 0.1$, $H < 0.3$, and $H < 0.5$, falls systematically. The strictest bucket, $H < 0.1$, drops from 20.1% to 3.4%. This suggests that dense SAE settings lose the strongest class alignment tail

first, rather than merely shifting the entire distribution slightly to the right.

Looking more closely along the L_0 axis, the monotonic trend is not uniform in speed, but shows clear saturation. From ultra-sparse to dense, both the mean H and the low-H fractions change substantially. From dense to very-dense, however, the marginal effect of further increasing L_0 becomes much smaller. In other words, the main effect of sparsity on class alignment is concentrated in the low-to-mid L_0 regime; beyond that, the marginal impact of larger L_0 diminishes.

If we inspect the absolute number of low-H features in representative SAE configurations, the size of the low-H tail becomes even clearer. On PII-noO, the three ultra-sparse SAEs ($L_0 \approx 18 - 23$) each independently contribute 2,100 – 2,650 features with $H < 0.1$ and 4,300 – 5,300 features with $H < 0.3$. By contrast, the most extreme very-dense model (layer 19, $L_0 = 279$) has only 41 features with $H < 0.1$, a difference of about 65×. That is, a single sparse SAE already contains thousands of candidate class-aligned features, more than enough to support the P/R validation of low-H features carried out in Section 5.2.

This result is consistent with the sparsity-capacity tradeoff in SAE training. A low L_0 constraint limits the number of features that can be active simultaneously for each token, which may encourage each active feature to correspond to a narrower semantic direction. Conversely, a high L_0 allows more features to participate jointly in reconstruction, which may lead to broader activation distributions over classes.

In summary, SAE sparsity is a major factor shaping feature-level class alignment in these experiments. Lower L_0 values are associated with lower H, higher KL, and a much larger low-H tail, while the marginal benefit diminishes in the higher-L_0 regime. These findings suggest that sparse SAE settings provide more candidate class-aligned features for downstream validation, and are consistent with the sparsity-capacity tradeoff.

5.1.3 Effect of layer depth

Subsection 5.1.2 showed that L_0 determines the overall direction of variation in class alignment. This subsection addresses a further question. Beyond the overall trend, do different layers also alter the structure of feature-level class alignment? If we only examine the marginal means aggregated by layer on the PII-noO dataset, the layer effect appears weak. Specifically, the mean H for layers 5, 12, and 19 all fall within a narrow range of 0.474 to 0.485. This difference of less than 3% seems to suggest that layer depth matters little. However, this observation is based only on marginal averages. Once we analyze all three datasets, the three layers, and the five L_0 levels together, the interaction between layer, L_0, and task becomes clear:

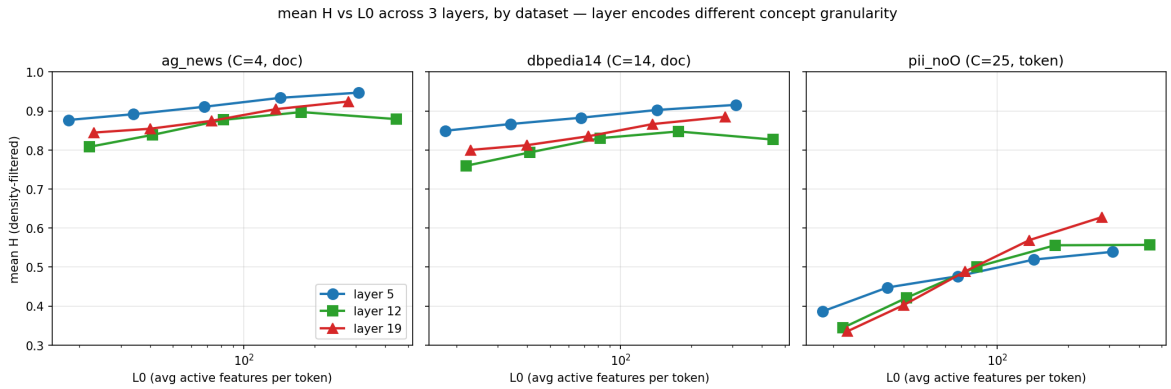


Figure 5.2: Mean H as a function of L_0 on the three datasets, color-coded by layer (blue: layer 5, green: layer 12, red: layer 19). The three panels share the same y-axis range 0.3 – 1.0 for direct comparison, corresponding from left to right to AG News / DBpedia14 / PII-noO.

The patterns in Figure 5.2 naturally split into two types: document-level and token-level.

Document-level (AG News and DBpedia14): Stable advantage of layer 12. In both document-level panels, layer 5 consistently has the highest H, layer 12 is almost always the lowest, and layer 19 stays in the middle but closer to layer 12. More importantly, this ordering is essentially unchanged from low L_0 to high L_0, so the layer effect here can be summarized by three traits: a stable ranking, a consistent direction, and little modulation by L_0.

One possible interpretation of this pattern is that concepts like news topics or encyclopedia categories represent an intermediate level of abstraction. Therefore, they are more likely to form stable directions in the middle residual stream. Shallow layer representations retain basic details, such as characters, morphology, and local collocations. However, this is rarely enough to summarize an entire document. In contrast, deep layer representations are heavily shaped by next-token prediction. They mix in more local context and output-oriented information. Middle layer representations occupy an intermediate position between these two extremes. They have moved beyond shallow, form-based features, but they are not yet dominated by deep predictive tasks. This may explain why layer 12 shows stronger alignment with document-level topic classes.

Token-level (PII-noO): The layer effect flips with L_0. Under PII-noO, the pattern changes completely. At low L_0, layer 19 has the lowest H, while layer 5 has the highest. As L_0 increases, this order gradually reverses. At high L_0, layer 5 reaches the lowest H, indicating the strongest class alignment, and layer 19 has the highest H, reflecting the weakest class alignment. In other words, PII-noO does not lack a layer effect; however,

the layer effect itself changes direction with L_0. This dynamic also helps explain why the marginal means across layers are so close; the layer ordering in different L_0 regimes cancels out after aggregation.

One possible explanation is that PII classes can be identified through two types of cues. The first type includes character or format cues, such as the "@" symbol for emails, digit strings for phone numbers, or date formats for birthdays. The second type involves contextual semantic cues, like a name appearing after the phrase "my name is." These two types of information are likely represented more strongly in shallow and deep layers, respectively. When L_0 is low, each active feature is under stronger pressure to represent a smaller number of cues. Under these conditions, deeper layers have already undergone stronger semantic organization. Therefore, they are more likely to align directly with a specific PII class, resulting in a lower H. Conversely, when L_0 is high, the SAE allows more features to be active per token. This allows shallow-layer SAEs to represent distinct format-level cues with separate features. Together, this creates a more concentrated distribution across PII classes, enabling shallow layers to outperform deeper ones. However, the current data directly supports only an interaction effect: the impact of the layer depends on the value of L_0. To verify exactly which semantic cues are processed by each layer, further evidence at the feature level is still required.

In conclusion, the impact of layer depth on class alignment is governed by task granularity and L_0. The current experiments illustrate this pattern. For document-level topic tasks, the middle layer examined here consistently shows lower H. In contrast, token-level PII tasks show a strong interaction between the specific layer and L_0. Therefore, these findings indicate that layer effects cannot be evaluated in isolation, but must be contextualized within the specific task granularity and sparsity constraints to fully understand SAE performance.

5.1.4 Joint distribution of H and density

Subsections 5.1.2 and 5.1.3 compared the average behavior across different L_0 tiers and layers at the SAE level. However, they did not explain how strongly class-aligned features are distributed within the feature space. This subsection therefore shifts the analysis to the feature level, aggregating all alive features from the 15 SAE settings on PII-noO by density interval. Using the density definition introduced in Subsection 4.2.3, this subsection groups features by activation density and examines how H varies across density intervals.

Table 5.3: Variation of H with density on PII-noO, aggregated over 15 SAE configurations.

density interval	**Median H**	**Mean H**	**frac($H < 0.3$)**	**frac($H > 0.7$)**
$[0, 10^{-4})$	0.333	0.312	43.5%	0.7%
$[10^{-4}, 10^{-3})$	0.588	0.549	11.1%	19.6%
$[10^{-3}, 10^{-2})$	0.645	0.595	9.9%	36.5%
$[10^{-2}, 1]$	0.730	0.672	5.9%	57.2%

Table 5.3 reveals a clear three-part structure, which in turn provides the empirical basis for the density range used in the main analysis and for the lower and upper density thresholds.

The ultra-low-density range $[0, 10^{-4})$ is a small-sample artifact band. Its median H is only 0.33 and frac($H < 0.3$) is as high as 43.5%, which at first glance appears to form a band of strongly class-aligned features. But features in this interval typically activate only 1 – 2 times over 10,000 evaluation samples, so $P(c \mid f)$ almost inevitably collapses to a single class under finite sampling. Therefore, the low H here is more likely to be a finite-sample artifact than reliable evidence of class alignment; this is also why the later validation stage imposes a lower density bound at 10^{-3}.

The high-density range $[10^{-2}, 1]$ is dominated by high-frequency features with diffuse class distributions. Its median H increases to 0.73, frac($H > 0.7$) reaches 57.2%, and frac($H < 0.3$) drops to only 5.9%. These features are likely to include syntactic structures or other common, generic patterns. Including them in the aggregation would systematically inflate the mean H and obscure the low-H tail that sparse SAEs naturally display. To prevent this issue, this section imposes an upper density limit of 10^{-2} for the aggregation process.

The middle range $[10^{-4}, 10^{-2})$ is the main study region of this section. Within this interval, H increases only mildly with density (median H rises from 0.59 to 0.65), but the low-H tail remains stable, frac($H < 0.3$) stays consistently around 10% – 11%. This indicates that reliable low-H features are not limited to the extreme low-frequency end, but are distributed across the entire middle band.

In summary, this three-part density structure provides the empirical basis for the filtering strategy. The low H values observed in the $[0, 10^{-4})$ range are not reliable because they are strongly affected by finite-sample effects. Meanwhile, the high-density $[10^{-2}, 1]$ region is dominated by features with diffuse class distributions and likely includes many generic linguistic patterns. Therefore, the middle band is the most reliable

region for identifying candidate class-aligned features. These observations help explain why the later candidate-selection procedure uses both H and density constraints.

5.1.5 The spurious low-entropy trap

Beyond the main results discussed above, there is an additional methodological trap. This issue directly affects how we interpret the metrics. This problem occurs when the `O` non-entity class is kept in the dataset. In that case, the PII-withO results may appear to show strong class alignment. However, this should not be treated as strong evidence for the monosemanticity proxy. The numbers we obtained may look extremely strong but are in fact misleading:

Table 5.4: Effect of including the `O` class on H/KL statistics for PII.

Dataset	Includes O?	# Classes	Mean H	Median H	Mean KL	$H < 0.3$ fraction	$H < 0.5$ fraction
PII-noO	No	25	0.4782	0.4996	2.4167	0.235	0.490
PII-withO	Yes	26	0.1962	0.1087	0.8205	0.743	0.892

Table 5.4 shows that on PII-withO, the mean H drops from 0.48 to 0.20, the median H from 0.50 to 0.11, and the fraction of features with $H < 0.3$ rises from 24% to 74%. Looking only at H, one might conclude that nearly 75% of the SAE features are strongly class-aligned. However, this is a canonical case of spurious low entropy.

The source of this trap is the class-imbalance issue in Section 4.1. When `O` accounts for the majority of tokens (80%+), a generic feature that merely follows the empirical class prior will also have $P(c \mid f)$ concentrated on the `O` class, and therefore receive a very low H. However, the H metric only measures how concentrated a distribution is. It cannot distinguish whether a feature is focused on a meaningful entity class or merely on a dominant background class.

KL divergence provides an important correction signal here. Specifically, the mean KL drops from 2.42 on PII-noO to 0.82 on PII-withO, less than one third of the PII-noO value. This decrease indicates that the low-H features in the PII-withO setting have class distributions that are much closer to the empirical class prior Q. Because they show so little deviation from this baseline, these features are likely just tracking the background distribution. They provide little evidence of discriminative class-specific structure. In short, under such strongly imbalanced class priors, low H must be interpreted together with KL.

For this reason, the main experiments in this thesis use PII-noO, which removes the `O` class from evaluation, allowing H to serve as the primary concentration metric

in a setting without a dominant background class, while retaining KL as an auxiliary correction signal for background effects.

5.1.6 Summary of H/KL results

The H/KL results support four main conclusions.

Label granularity is the primary factor determining whether the H/KL metrics have discriminative power. In fine-grained token-level settings such as PII-noO, the dynamic range of H/KL can be effectively opened up, the mean H is around 0.48, and the fraction of features with $H < 0.3$ reaches 23.5%.

SAE sparsity is the dominant hyperparameter affecting feature-level class alignment in these experiments. The sparser the SAE setting, the stronger the observed class alignment. This trend appears not only in the means of H/KL, but also in the fraction and absolute number of features in the low-H tail.

Layer depth affects class alignment dynamically, heavily conditioned by task granularity and L_0. While middle layers consistently provide the strongest alignment for document-level topics, the optimal layer for token-level tasks reverses depending on the sparsity setting. Therefore, layer effects must always be contextualized within specific tasks and sparsity constraints.

At the feature level, H exhibits a three-part structure as a function of density. The interval $[0, 10^{-4})$ is a small-sample artifact band, $[10^{-2}, 1]$ is dominated by high-frequency generic features, while the middle range $[10^{-4}, 10^{-2})$ is the most reliable region for analyzing the low-H tail.

These findings motivate the P/R validation in Section 5.2, which tests whether the low-H, high-KL features selected under these constraints also function as useful class-level detectors.

5.2 Reverse P/R validation

This section uses the P/R validation framework defined in Section 4.2 to test whether H/KL rankings select SAE features with stronger class alignment. It first presents the core results, and then examines whether the H/KL rankings provide a useful class alignment signal, the rationale for using amplitude-weighted precision and span recall as the primary validation metrics, and the extent to which the sparsity and layer conclusions from Section 5.1 are supported by the validation results. All results in this section are macro-averaged over the 15 SAE configurations on PII-noO, corresponding to three layers and five L_0 levels.

5.2.1 Main validation results

To keep the narrative clear and focused, the main text reports only the primary metric pair: `ampP` and `spnR`. In this context, `ampP` denotes amplitude-weighted precision and `spnR` denotes span recall. For completeness, `tokP` and `tokR` denote token-level precision and recall, while `spnP` denotes span-level precision. The analysis focuses on five key comparison groups: `h_f`, `kl_f`, `density`, `mi`, and `random`. For the complete table of results, please refer to Appendix A.

Table 5.5: Results of the five key comparison groups under the primary metric pair `(ampP, spnR)` at different values of `k`. Each cell reports `(ampP, spnR)`.

k	h_f	kl_f	density	mi	random
1	0.858 / 0.404	0.836 / 0.386	0.251 / 0.830	0.438 / 0.887	0.448 / 0.105
5	0.772 / 0.799	0.763 / 0.782	0.295 / 0.973	0.406 / 0.974	0.389 / 0.391
10	0.709 / 0.902	0.705 / 0.892	0.315 / 0.985	0.398 / 0.983	0.379 / 0.573
20	0.631 / 0.955	0.631 / 0.949	0.332 / 0.989	0.388 / 0.988	0.372 / 0.748

Table 5.5 reveals two key observations. First, the `h_f` and `kl_f` groups maintain relatively high coverage while achieving substantially higher amplitude-weighted precision than the `density` and `mi` groups. This suggests that H/KL rankings are not merely selecting high-frequency features. Second, the overall performance of `h_f` and `kl_f` is broadly similar, although `h_f` shows a slight advantage. The difference between the raw `h / kl` groups and the floored `h_f / kl_f` groups, as well as the complete token/span results, are deferred to the later discussion and Appendix A.

5.2.2 Validating H/KL against low-frequency, high-frequency, and random baselines

The first validation question is whether H/KL rankings avoid low-frequency degeneracy. The raw `h` and `kl` rankings often select features that activate only once or twice. These features can have nearly perfect class concentration, but they cover very few instances of the target class.

The density floor solves this problem. At $k = 1$, raw `h` has span recall 0.026, while `h_f` reaches 0.404. Raw `kl` has span recall 0.028, while `kl_f` reaches 0.386. This difference of more than an order of magnitude shows that the density floor is not a minor implementation detail. It is necessary for turning entropy and KL into useful retrieval signals.

The second validation question is whether H/KL rankings outperform random selection from the same candidate pool. The results indicate that they do. At $k = 5$, `h_f` span recall is 0.799, compared with 0.391 for random. At $k = 10$, `h_f` span recall is 0.902, compared with 0.573 for random. The ranking signal therefore adds substantial information beyond simply assigning each feature to its highest-mass class.

The third validation question is whether H/KL rankings merely select high-frequency features. The results suggest that they do not. At $k = 5$, `density` and `mi` reach span recall 0.973 and 0.974, respectively, but their amplitude-weighted precision is only 0.295 and 0.406. By contrast, `h_f` has lower but still strong span recall, and much higher amplitude-weighted precision at 0.772.

This pattern shows that high recall alone is not evidence of strong class alignment. A high-frequency feature can cover many spans simply by activating broadly. What distinguishes `h_f` is the combination of coverage and purity. It covers target-class spans while concentrating its activation mass on the target class.

5.2.3 Choice of primary P/R metrics

This subsection explains why the main text uses `ampP` and `spnR` as the primary precision and recall metrics, respectively.

Table 5.6: Comparison between `tokP` and `ampP` for `h_f` under different values of `k`.

k	tokP	ampP	Δ
1	0.821	0.858	+0.037
5	0.632	0.772	+0.140
10	0.492	0.709	+0.217
20	0.356	0.631	+0.275

As k increases, the gap becomes larger. At $k = 20$, `tokP` is only 0.356, while `ampP` still reaches 0.631. This is not a contradiction, but directly reflects the point emphasized in Subsection 4.3.2: FP activations are systematically weaker in magnitude than TP activations.

Frequency precision treats every FP activation as an equally weighted error. However, in SAE systems, many FP activations have much smaller amplitudes than TP activations. As a result, these low-amplitude FP activations are closer to background noise. In contrast, amplitude-weighted precision weights activations by their magnitude. This approach better answers a critical question: what fraction of the feature group's total activation mass is assigned to the target class? This advantage becomes

especially clear against the random baseline. For instance, at $k = 20$, the `h_f` method achieves an `ampP` of 0.631, while the random baseline reaches only 0.372. The gap between `h_f` and the random baseline is much smaller when measured by `tokP`. Therefore, `ampP` serves as the primary precision metric in this work.

Turning to the recall metrics, the `h_f` group achieves a `tokR` of 0.497 and a span-level recall (`spnR`) of 0.799 at $k = 5$. When $k = 10$, these values increase to 0.666 for `tokR` and 0.902 for `spnR`. In practice, after tokenization, multi-token entities typically span 3 – 8 subwords. The `spnR` metric addresses this by grouping all tokens within a single entity instance into one evaluation unit. Under `spnR`, an entity instance is counted as found as long as at least one top-k feature activates on at least one token in the span. The fact that `spnR` is substantially higher than `tokR` is informative. It indicates that the selected features usually activate somewhere within a PII instance, even if they fail to cover every single subword comprising that instance.

The substantially lower `tokR` does not always indicate a failure to identify the entity instance. Instead, the performance gap between `tokR` and `spnR` originates from two distinct sources. The first source is a "false miss." Some subword tokens, such as punctuation marks, separators, or generic digit strings, appear across multiple PII classes. Features that activate on these ambiguous tokens tend to have their probability distribution $P(c \mid f)$ spread across several classes. This spread increases their H value and pushes them down the ranking. Consequently, these features rarely make it into the top-k selection. Because of this ranking mechanism, the selected features tend to be more class-aligned under the chosen label space, and they are less likely to activate on these ambiguous tokens. Unfortunately, the `tokR` metric penalizes this class-specificity, incorrectly treating it as a failure to cover those tokens. The second source is a "true miss." This occurs when subwords within a target-class span are genuinely not activated by any top-k feature. This specific shortfall represents the actual gap in recall.

The `spnR` metric groups all tokens within an entity instance into a single evaluation unit. Because of this design, `spnR` is closer to measuring entity-level misses and is less affected by the token-level false misses discussed above. From a downstream perspective, the primary goal is to determine whether an entity instance can be identified at all. Since `spnR` directly addresses this question, it is more appropriate for the entity-level PII setting used in this thesis. Therefore, it serves as the primary recall metric in this study.

5.2.4 Sparsity and layer effects in P/R validation

We begin by examining sparsity. As demonstrated in Section 5.1, there is a clear pattern: lower L_0 settings generally correspond to lower H and stronger class alignment.

To simplify our comparisons, we define within-layer L_0 ranks. For every layer, we sort its five SAEs by their L_0 values from smallest to largest. We then group SAE configurations with the same within-layer L_0 rank across the three layers into a single tier. Therefore, tier 0 consists of the sparsest SAE from each layer, while tier 4 contains the densest. Based on this tier structure, we then aggregate the h_f metrics at $k = 5$:

Table 5.7: `h_f ampP` and `h_f spnR` across within-layer L_0 tiers on PII-noO at $k = 5$.

L_0 tier	`h_f ampP`	`h_f spnR`
0 (sparse)	**0.816**	0.787
1	0.802	0.791
2	0.778	0.813
3	0.753	0.803
4 (dense)	0.710	0.802

The table shows that `ampP` decreases monotonically from 0.82 at tier 0 to 0.71 at tier 4, indicating that the top-5 features selected from sparse SAE settings concentrate a larger share of their activation mass on the target class. This is a direct class-alignment signal and, under the proxy assumption introduced earlier, provides stronger evidence for monosemanticity.

By contrast, `spnR` is roughly flat or even slightly increasing (from 0.787 in tier 0 to 0.802 in tier 4), but this does not mean that higher-L_0 SAE settings have stronger class alignment. A more plausible explanation is that dense SAE settings have more alive features (Section 5.1 showed about 72% alive at $L_0 = 22$ versus about 99% alive at $L_0 = 445$). As a result, the same-class candidate pools are larger. This makes it easier for the $k = 5$ top-k selection to form a feature set that covers most spans of the class. In other words, dense SAEs exchange "multi-feature redundancy" for a small gain in OR-union coverage, rather than making each individual feature more class-aligned.

The `ampP` metric is the more direct class-alignment measure here, because it measures what fraction of the feature set's total activation mass is assigned to the target class. Importantly, this metric is not affected by the size of the candidate pool. As a result, the H trends presented in Section 5.1 and the `ampP` trends discussed here point to the same conclusion. They suggest that stronger sparsity constraints are associated with stronger class alignment. Under the proxy interpretation, this improvement provides stronger evidence for monosemanticity.

Next, we consider layer effects. Relative to L_0 tiers, the differences across layers are much smaller when results are averaged over the five L_0 settings for each layer. Specifically, for the ampP metric, layer 19 exceeds layer 5 by only 5.3 percentage

points (0.741 versus 0.688). Similarly, for the `spnR` metric, layer 12 exceeds layer 5 by just 3 percentage points (0.919 versus 0.889). Although these numerical differences are relatively small, the overall trends remain clear.

Table 5.8: `h_f ampP` and `h_f spnR` across layers on PII-noO at $k = 10$.

layer	`h_f` ampP (k=10)	`h_f` spnR (k=10)
5	0.688	0.889
12	0.699	**0.919**
19	**0.741**	0.897

Layer 19 achieves the highest `ampP`, while layer 12 achieves the highest `spnR`. This suggests a possible division of roles across layers. Candidate features in deeper layers show higher amplitude-weighted precision, whereas the combined feature sets from the middle layer examined here provide more complete span coverage for a given class. Previously, Subsection 5.1.3 demonstrated that the ordering of H across layers on the PII-noO dataset flips depending on L_0. However, when we analyze the data by L_0 tier, this flipping behavior does not transfer to the `ampP` and `spnR` metrics. Instead, the `ampP` of layer 19 and the `spnR` of layer 12 consistently rank first across all five L_0 tiers. One possible interpretation is that the H flip is a property of the distribution center of the features. This does not contradict the interaction observed in Subsection 5.1.3. The two analyses summarize different parts of the feature distribution. The H curves describe the central tendency across all alive features, while the P/R validation evaluates only the selected top-k tail.

In contrast, the `h_f` top-k depends only on the low-H, density-filtered tail of each layer's feature distribution. This tail remains consistently stronger for layers 19 and 12 compared to layer 5, so the interaction observed in the full H distribution becomes less visible in the top-k P/R validation results. Finally, the performance gap between layers is clearly smaller than the gap between L_0 tiers. For this reason, the layer effect is treated as a secondary observation in this section.

5.2.5 Summary of P/R validation results

The core conclusions are as follows. First, the P/R results support H and KL as effective ranking signals for class alignment. Furthermore, based on the proxy assumption introduced in Subsection 4.1.1, these metrics can serve as practical proxy indicators related to monosemanticity. This conclusion is jointly supported by three key pieces of evidence:

Non-degenerate: once the density floor is added, the `spnR` of `h_f` rises from the raw `h` group's 0.026 to 0.404 at $k = 1$, from 0.110 to 0.799 at $k = 5$, and from 0.181 to 0.902 at $k = 10$. This shows that H-ranked features with a density floor cover target-class spans rather than forming a degenerate solution that almost never activates.

Non-random: `h_f` exceeds the random baseline by about 41 percentage points in `spnR` at $k = 5$ (0.799 versus 0.391), and by about 26 percentage points in `ampP` at $k = 20$ (0.631 versus 0.372). Therefore, the floored H ranking provides a class-alignment signal; if the H-based ranking did not distinguish more class-aligned features from less class-aligned ones, these numbers would be much closer.

Not merely high-frequency: using `density` and `mi` as counterfactual baselines tests the claim that high activation frequency alone is sufficient. Their `ampP` fluctuates only between 0.25 and 0.44, while `h_f` reaches 0.63 – 0.86, about $2 - 3\times$ larger than density. The H-based ranking is not merely selecting high-frequency features; it selects features with stronger class-aligned activation concentration.

Thus, the results support H and KL as class-alignment ranking signals for SAE features. Under the proxy interpretation introduced in Subsection 4.1.1, this provides practical evidence related to monosemanticity. However, it still keeps class alignment distinct from monosemanticity itself. P/R provides a complementary validation signal and helps rule out alternative explanations based on random selection, high-frequency baselines, and low-frequency degeneracy. Therefore, the metric pair (`ampP, spnR`) provides a suitable validation summary for the class-alignment results in this thesis.

6. Conclusion

6.1 Summary of the study

This thesis examined how SAE features can be evaluated for monosemanticity under a scalable and empirically grounded framework. Instead of treating reconstruction quality or sparsity as direct evidence of interpretability, the thesis focused on class alignment as a measurable proxy. The main idea was that a more monosemantic candidate feature should concentrate its activation mass on a smaller and more coherent set of classes.

The study used the information-theoretic framework defined in Section 4.1 to address RQ1 and RQ2. Specifically, normalized Shannon entropy and KL divergence measured feature-level class alignment, while the empirical analyses in Subsections 5.1.1 – 5.1.3 examined how class alignment changed with label granularity, sparsity level, and layer depth. The study then used the reverse P/R validation framework defined in Section 4.2 to address RQ3, with the main validation results reported in Subsections 5.2.1 – 5.2.4. This design allowed the thesis to evaluate not only whether a feature looked class-specific, but also whether class-aligned features were useful in practice.

In this way, the conclusion returns to the research gap identified in Section 1.4. The results address the first gap by evaluating SAE feature monosemanticity more directly, using class alignment as a measurable proxy. They also address the second gap by clarifying the interpretation and boundary conditions of selectivity-based evaluation. In particular, the findings show that label granularity, sparsity, and layer depth all affect how H/KL should be interpreted.

Overall, the thesis shows that class alignment can be a useful way to screen SAE features, but only under clear methodological limits. It is most informative when the label space is fine-grained and when the SAE is sparse enough to produce concentrated features. It is less informative when labels are too coarse, highly imbalanced, or not well matched to the concepts represented by the model.

6.2 Main conclusions

Building on the summary in Section 6.1, this section draws the main conclusions from the RQ-specific results reported in Chapter 5. The main conclusions can be organized around the three RQs, moving from measurement to conditioning factors and then to validation.

The first conclusion is that H and KL provide a practical but conditional way to evaluate feature-level class alignment as a proxy for SAE feature monosemanticity. They make it possible to rank features across a full SAE dictionary without relying on manual inspection. This is useful because SAE dictionaries are too large to evaluate feature by feature. However, the results also show that these metrics should not be interpreted as direct proof of monosemanticity. A low-entropy feature is not automatically monosemantic. It only shows concentration with respect to the given label set.

The second conclusion is that label granularity is critical. The experiments show that coarse document-level labels are often not suitable for this evaluation. A document can contain many local concepts, so a feature may be meaningful without aligning with the document's global category. Token-level labels provide a more appropriate test case because they are closer to the level at which many SAE features appear to activate. This suggests that future evaluations using label-based proxies for SAE monosemanticity should pay close attention to whether the labels match the scale of the features being studied.

The third conclusion is that SAE sparsity has a strong influence on class alignment. Sparser SAE settings produced more concentrated and more useful class-aligned features in the experiments. This supports the view that sparsity is not only a training constraint, but also an important factor for interpretability. However, sparsity alone is not sufficient. A sparse feature may still be uninformative if it activates too rarely or fails to recover meaningful examples from its target class.

The fourth conclusion is that layer depth matters, but its effect is less direct than the effect of sparsity. The experiments suggest that different layers may support different interpretability properties. Some layers may produce features with higher class-specific precision, while others may provide better coverage when several features are combined. This means that there may not be a single best layer for all interpretability goals. The choice of layer should depend on whether the priority is class-specific precision, coverage, or downstream class detection.

The fifth conclusion is that reverse validation is an important complement to information-theoretic scoring. H/KL rankings can identify candidate features, but they cannot show whether these features recover instances of the corresponding class. P/R

validation provides an important additional check. In particular, amplitude-weighted precision and span recall are suited to this setting, because they reflect how SAE activations behave and how entity spans are represented in tokenized text.

6.3 Contributions

These contributions follow from the RQs discussed in Section 1.4 and the evaluation literature reviewed in Chapter 2. This thesis contributes a simple evaluation framework for studying SAE feature-level class alignment as a scalable proxy for monosemanticity. The framework is useful because it separates two questions that are often combined: whether a feature's activation mass is concentrated on a class, and whether that feature can recover examples of that class. By combining H/KL scoring with reverse P/R, the thesis provides a more robust evaluation than using either the feature-to-class or class-to-feature direction alone. It also positions entropy and KL as scalable screening tools, not as final proof of monosemanticity.

The thesis also clarifies the conditions under which class-alignment metrics are meaningful. It shows that these metrics depend strongly on the label space, the sparsity level of the SAE, and the layer under evaluation. In particular, label granularity is a key factor for whether entropy-based SAE feature evaluation is informative. This is important because it prevents overinterpretation of low-entropy features. The results do not support a general claim that low entropy always means monosemanticity. Instead, they support a more careful claim: class alignment is a useful proxy when the labels are fine-grained and the selected features are validated through reverse P/R analysis.

Finally, the thesis shows that class-aligned SAE features may have practical value as lightweight signals for detection tasks, especially in settings such as PII identification. However, they should be used as support tools rather than complete detection systems. More broadly, this thesis argues that SAE feature evaluation should take the label space into account.

6.4 Implications for SAE evaluation

This section connects the implications of the thesis to the evaluation literature reviewed in Section 2.4. The findings support the view discussed in Subsection 2.4.1 that structural proxy metrics are necessary but insufficient for interpretability. The proposed H/KL and reverse P/R framework also complements the automated interpretability and causal evaluation approaches discussed in Sections 2.4.2 and 2.4.4. It is not intended to replace manual explanations or causal interventions. Instead, it helps

narrow down candidate features that can later be examined with those more detailed methods.

The findings of this thesis have several implications for SAE feature evaluation. First, class-alignment metrics should be interpreted as measures that depend on the chosen label space, rather than universal measures of monosemanticity. Low entropy only shows that a feature is concentrated under the chosen label space; it does not by itself prove that the feature has a single human-interpretable meaning.

Second, the results suggest that fine-grained token-level or span-level labels are more informative than coarse document-level labels for evaluating SAE features' class alignment. Document-level labels may hide local feature structure because many SAE features correspond to entity-level, lexical, syntactic, or format-based patterns, not whole-document topics.

Third, entropy should be combined with prior-aware and reverse-validation checks. KL divergence helps identify whether a feature differs from the empirical class prior, while reverse P/R tests whether selected features can recover examples of their assigned classes. Together, these checks make H/KL more suitable as scalable screening tools.

Overall, the thesis suggests that SAE feature evaluation should be label-space-aware and multi-stage. The proposed H/KL and reverse P/R framework is best understood as an intermediate evaluation layer for identifying candidate class-aligned features. These candidates require further evidence before being treated as fully monosemantic.

6.5 Limitations

The main limitation of this thesis is that class alignment depends on the available evaluation labels. H and KL only measure the concentration of activation mass over predefined classes. If the label set does not match the concepts represented by the model, the metrics may underestimate or misrepresent class-aligned interpretability. This is especially important for coarse document-level labels, where many meaningful local features may be spread across several document categories.

A second limitation is that the method favors features whose meanings align with the predefined class labels. Some features may be stable and interpretable, but not tied to a single class. For example, a feature could represent a syntactic pattern, a common formatting structure, or a broad lexical property. Such a feature may receive a weak H/KL score even if it is meaningful. Therefore, the framework evaluates one label-aligned form of interpretability rather than all possible forms of feature meaning.

A third limitation is that the reverse validation results remain imperfect. The

best features still produce false positives and false negatives. Some errors may be caused by overlap between related classes, such as different types of names. It may also reflect limits of the specific base model, SAE family, and activation-site setting used in this thesis. The current experiments cannot fully separate these causes.

A fourth limitation is that many classes appear to be represented by groups of features rather than by a single feature. This weakens a strict "one feature, one concept" interpretation of the results. It suggests that, in practice, feature groups may be a better unit of analysis than individual features.

Finally, the thesis does not provide direct causal evidence. The results show that some features are class-aligned and can recover target examples, but they do not prove how these features affect model behavior. Causal intervention experiments would be needed to test that question more directly.

6.6 Future work

Future work should first test whether the findings generalize across other models, SAE families, activation sites, and feature widths. This would show whether the observed effects of label granularity, sparsity, and layer are specific to this experimental setup or reflect a broader pattern in SAE interpretability.

A second direction is to combine this framework with more direct interpretability methods. H/KL can be used to select candidate features at scale, and then manual inspection, automated interpretability, or causal interventions can be applied to the most promising cases. This would make the evaluation both scalable and more grounded in feature-level semantic evidence.

A third direction is to improve the evaluation label space. Hierarchical labels, multi-label annotations, or learned concept taxonomies may provide a better match for the concepts or patterns captured by SAE features. This would be especially useful for document-level tasks, where global labels may be too coarse.

A fourth direction is to evaluate feature groups. Since the experiments suggest that several related features may jointly cover one class, future work could study whether small feature sets provide more reliable and interpretable units of analysis.

Finally, future work could explore practical applications in privacy and safety analysis. The PII results suggest that class-aligned SAE features may help identify tokens or spans related to sensitive information through model activations. However, such applications require caution. Incomplete detection could still create privacy risks.

6.7 Final remarks

This thesis shows that SAE feature monosemanticity can be studied through class alignment as a proxy, but only with careful interpretation. H and KL are useful for finding candidate class-aligned features, especially when the label space is fine-grained and the SAE is sufficiently sparse. Reverse P/R validation further shows whether these candidates are practically meaningful.

The broader conclusion is that scalable interpretability evaluation should be label-space aware. It should not rely on a single metric or a single evaluation direction. A feature that looks clean under one measure may still fail under another. The value of the proposed framework is that it makes this tension explicit. It offers a practical way to identify promising SAE features for further interpretability or causal analysis. At the same time, it makes clear where the current evidence stops.

Bibliography

[1] S. Abnar and W. Zuidema. Quantifying attention flow in transformers. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 4190–4197. Association for Computational Linguistics, 2020.

[2] Ai4Privacy. pii-masking-300k, 2024. https://huggingface.co/datasets/ai4privacy/pii-masking-300k, Accessed on 20 May, 2026.

[3] G. Alain and Y. Bengio. Understanding intermediate layers using linear classifier probes. In *The fifth International Conference on Learning Representations*, 2017.

[4] K. Ayonrinde, M. T. Pearce, and L. Sharkey. Interpretability as compression: Reconsidering sae explanations of neural activations with mdl-saes. *arXiv preprint arXiv:2410.11179*, 2024.

[5] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6541–6549. IEEE, 2017.

[6] Y. Belinkov. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219, 2022.

[7] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pages 610–623. Association for Computing Machinery, 2021.

[8] L. Bereska and E. Gavves. Mechanistic interpretability for ai safety – a review. *arXiv preprint arXiv:2404.14082*, 2024.

[9] R. Bommasani, D. A. Hudson, E. Adeli, R. Altman, S. Arora, et al. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2022.

[10] T. Bricken, A. Templeton, J. Batson, B. Chen, A. Jermyn, et al. Towards monosemanticity: Decomposing language models with dictionary learning, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html, Accessed on 20 May 2026.

[11] B. Bussmann, P. Leask, and N. Nanda. Batchtopk sparse autoencoders. *arXiv preprint arXiv:2412.06410*, 2024.

[12] B. Bussmann, N. Nabeshima, A. Karvonen, and N. Nanda. Learning multi-level features with matryoshka sparse autoencoders. *arXiv preprint arXiv:2503.17547*, 2025.

[13] D. Chanin, J. Wilken-Smith, T. Dulka, H. Bhatnagar, S. Golechha, and J. Bloom. A is for absorption: Studying feature splitting and absorption in sparse autoencoders. *arXiv preprint*, 2025.

[14] N. Elhage, T. Hume, C. Olsson, N. Schiefer, T. Henighan, et al. Toy models of superposition. *arXiv preprint arXiv:2209.10652*, 2022.

[15] M. Fereidouni, M. U. Haider, P. Ju, and A. Siddique. Evaluating sparse autoencoders for monosemantic representation. In *Findings of the Association for Computational Linguistics: EACL 2026*, pages 5969–5984, 2026.

[16] L. Gao, T. Dupré la Tour, H. Tillman, G. Goh, R. Troll, A. Radford, I. Sutskever, J. Leike, and J. Wu. Scaling and evaluating sparse autoencoders. In *The Thirteenth International Conference on Learning Representations*, 2024.

[17] J. Han, W. Xu, M. Jin, and M. Du. Sage: An agentic explainer framework for interpreting sae features in language models. *arXiv preprint arXiv:2511.20820*, 2025.

[18] R. Härle, F. Friedrich, M. Brack, S. Wäldchen, B. Deiseroth, P. Schramowski, and K. Kersting. Measuring and guiding monosemanticity. *arXiv preprint arXiv:2506.19382*, 2025.

[19] R. Huben, H. Cunningham, L. Riggs Smith, A. Ewart, and L. Sharkey. Sparse autoencoders find highly interpretable features in language models. In *The Twelfth International Conference on Learning Representations*, 2024.

[20] P. Joly, A. Kumar, A. Dziedzic, and F. Boenisch. TRACE: Transcoder-based concept editing, 2026. https://openreview.net/forum?id=yZirk1CoZQ, Submitted to ICLR 2026, Accessed on 20 May 2026.

[21] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, et al. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[22] A. Karvonen, C. Rager, J. Lin, C. Tigges, J. Bloom, et al. Saebench: A comprehensive benchmark for sparse autoencoders in language model interpretability. *arXiv preprint arXiv:2503.09532*, 2025.

[23] B. Kim, M. Wattenberg, J. Gilmer, C. Cai, J. Wexler, F. Viegas, et al. Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR, 2018.

[24] V. Kurochkin, Y. Aksenov, D. Laptev, D. Gavrilov, and N. Balagansky. Kronecker factorization improves efficiency and interpretability of sparse autoencoders, 2025. https://openreview.net/forum?id=CVXpkc3bXc, Submitted to ICLR 2026, Accessed on 20 May 2026.

[25] V. Lecomte, K. Thaman, R. Schaeffer, N. Bashkansky, T. Chow, and S. Koyejo. What causes polysemanticity? an alternative origin story of mixed selectivity from incidental causes. *arXiv preprint arXiv:2312.03096*, 2023.

[26] A. J. Li, S. Srinivas, U. Bhalla, and H. Lakkaraju. Interpretability illusions with sparse autoencoders: Evaluating robustness of concept representations. *arXiv preprint arXiv:2505.16004*, 2025.

[27] K. Li, A. K. Hopkins, D. Bau, F. B. Viégas, H. Pfister, and M. Wattenberg. Emergent world representations: Exploring a sequence model trained on a synthetic task. In *The Eleventh International Conference on Learning Representations*. OpenReview.net, 2023.

[28] P. Liang, R. Bommasani, T. Lee, D. Tsipras, D. Soylu, et al. Holistic evaluation of language models. *arXiv preprint arXiv:2211.09110*, 2023.

[29] T. Lieberum, S. Rajamanoharan, A. Conmy, L. Smith, N. Sonnerat, V. Varma, J. Kramár, A. Dragan, R. Shah, and N. Nanda. Gemma scope: Open sparse autoencoders everywhere all at once on gemma 2. In *Proceedings of the 7th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pages 278–300, 2024.

[30] A. Makelov, G. Lange, and N. Nanda. Towards principled evaluations of sparse autoencoders for interpretability and control. *arXiv preprint arXiv:2405.08366*, 2024.

[31] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013.

[32] G. Minegishi, H. Furuta, Y. Iwasawa, and Y. Matsuo. Rethinking evaluation of sparse autoencoders through the representation of polysemous words. In *The Thirteenth International Conference on Learning Representations*, 2025.

[33] T. Oikarinen and T.-W. Weng. Linear explanations for individual neurons. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 38639–38662. PMLR, 2024.

[34] K. Park, Y. J. Choe, and V. Veitch. The linear representation hypothesis and the geometry of large language models. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235, pages 39643–39666. PMLR, 2024.

[35] G. S. Paulo, A. T. Mallen, C. Juang, and N. Belrose. Automatically interpreting millions of features in large language models. In *International Conference on Machine Learning*, 2025.

[36] D. Rai, Y. Zhou, S. Feng, A. Saparov, and Z. Yao. A practical review of mechanistic interpretability for transformer-based language models. *arXiv preprint arXiv:2407.02646*, 2025.

[37] S. Rajamanoharan, A. Conmy, L. Smith, T. Lieberum, V. Varma, J. Kramár, R. Shah, and N. Nanda. Improving sparse decomposition of language model activations with gated sparse autoencoders. In *Advances in Neural Information Processing Systems*, 2024.

[38] S. Rajamanoharan, T. Lieberum, N. Sonnerat, A. Conmy, V. Varma, J. Kramár, and N. Nanda. Jumping ahead: Improving reconstruction fidelity with jumprelu sparse autoencoders. *arXiv preprint arXiv:2407.14435*, 2024.

[39] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[40] M. Sundararajan, A. Taly, and Q. Yan. Axiomatic attribution for deep networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 3319–3328. PMLR, 2017.

[41] G. M. Taggart. Prolu: A nonlinearity for sparse autoencoders. Alignment Forum, 2024. https://www.alignmentforum.org/posts/HEpufTdakGTTKgoYF/prolu-a-nonlinearity-for-sparse-autoencoders, Accessed on 20 May 2026.

[42] G. Team, M. Riviere, S. Pathak, P. G. Sessa, C. Hardin, S. Bhupatiraju, L. Hussenot, T. Mesnard, B. Shahriari, A. Ramé, et al. Gemma 2: Improving open language models at a practical size. *arXiv preprint arXiv:2408.00118*, 2024.

[43] H. Thasarathan, J. Forsyth, T. Fel, M. Kowal, and K. G. Derpanis. Universal sparse autoencoders: Interpretable cross-model concept alignment. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267, pages 59304–59325. PMLR, 2025.

[44] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[45] X. Zhang, J. Zhao, and Y. LeCun. Character-level convolutional networks for text classification. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, pages 649–657, 2015.

[46] Y. Zhang, P. Tiňo, A. Leonardis, and K. Tang. A survey on neural network interpretability. *IEEE transactions on emerging topics in computational intelligence*, 5(5):726–742, 2021.

Appendix A. Full P/R Results Table

A.1 Full P/R Results for 15 SAEs on PII-noO

This appendix provides the complete result table corresponding to Section 5.2, retaining all ranking groups (h / kl / h_f / kl_f / mi / density / random) and all evaluation combinations (tokP / tokR, spnP / spnR, and ampP / spnR). The main text keeps only the primary metric pair (ampP, spnR) in order to preserve the main line of argument.

k	metric	h	kl	h_f	kl_f	mi	density	random
1	tokP / tokR	0.978 / 0.013	0.802 / 0.015	0.821 / 0.168	0.807 / 0.159	0.312 / 0.687	0.179 / 0.630	0.419 / 0.043
1	spnP / spnR	0.974 / 0.026	0.798 / 0.028	0.785 / 0.404	0.771 / 0.386	0.228 / 0.887	0.133 / 0.830	0.402 / 0.105
1	ampP / spnR	0.985 / 0.026	0.812 / 0.028	0.858 / 0.404	0.836 / 0.386	0.438 / 0.887	0.251 / 0.830	0.448 / 0.105
5	tokP / tokR	0.942 / 0.073	0.823 / 0.070	0.632 / 0.497	0.632 / 0.473	0.184 / 0.896	0.137 / 0.897	0.337 / 0.185
5	spnP / spnR	0.926 / 0.110	0.807 / 0.109	0.517 / 0.799	0.514 / 0.782	0.117 / 0.974	0.089 / 0.973	0.296 / 0.391
5	ampP / spnR	0.966 / 0.110	0.846 / 0.109	0.772 / 0.799	0.763 / 0.782	0.406 / 0.974	0.295 / 0.973	0.389 / 0.391
10	tokP / tokR	0.897 / 0.124	0.813 / 0.126	0.492 / 0.666	0.498 / 0.642	0.148 / 0.933	0.121 / 0.938	0.304 / 0.305
10	spnP / spnR	0.874 / 0.181	0.790 / 0.190	0.357 / 0.902	0.362 / 0.892	0.091 / 0.983	0.077 / 0.985	0.246 / 0.573
10	ampP / spnR	0.940 / 0.181	0.853 / 0.190	0.709 / 0.902	0.705 / 0.892	0.398 / 0.983	0.315 / 0.985	0.379 / 0.573
20	tokP / tokR	0.821 / 0.210	0.764 / 0.212	0.356 / 0.801	0.365 / 0.774	0.122 / 0.953	0.109 / 0.959	0.269 / 0.459
20	spnP / spnR	0.784 / 0.316	0.726 / 0.314	0.231 / 0.955	0.233 / 0.949	0.076 / 0.988	0.070 / 0.989	0.196 / 0.748
20	ampP / spnR	0.900 / 0.316	0.839 / 0.314	0.631 / 0.955	0.631 / 0.949	0.388 / 0.988	0.332 / 0.989	0.372 / 0.748

Appendix B. Disclaimer on the Use of AI Tools

AI tools were used during different stages of this thesis. Their use can be divided into three main phases.

In the early stage of the thesis, my knowledge of LLM and SAEs was still limited. I used ChatGPT to support my learning process and to help me understand background concepts related to LLMs, mechanistic interpretability, and SAEs. This helped me improve the efficiency of my literature learning and build the basic knowledge needed for the thesis.

In the middle stage of the thesis, I began the experimental work. During this stage, AI was mainly used in two ways. First, I used ChatGPT to help me become more familiar with CSC tools and workflows. Before this thesis, I had only limited experience with CSC tools from one course. In addition to reading the official CSC documentation, I used ChatGPT to understand basic environment setup, command-line operations, and file management practices. This was also useful because different CSC directories have different storage limits and capacity rules. I managed the experiment files based on both the official documentation and suggestions from ChatGPT.

Second, I used ChatGPT to help debug experimental code. For example, the initial experimental workflow was run on Puhti nodes, but the code later produced repeated errors. Through discussion with ChatGPT, I identified that the V100 GPU architecture on Puhti was not fully compatible with the large model used in the experiments. With the help of ChatGPT, I revised the code and migrated the experiments from Puhti to Mahti.

In the final stage of the thesis, AI tools were mainly used for writing support. When editing the thesis in Overleaf, I used ChatGPT to ask about LaTeX commands and layout adjustments, especially for tables, figures, and pages with large blank spaces. This improved the efficiency of formatting the thesis. After completing the first draft based on the experimental reports, I also used ChatGPT and Grammarly for language polishing, spelling checks, grammar correction, and sentence structure improvement.