IMPROVING VISION LANGUAGE MODELS ON DISCRIMINTIVE TASKS

Mohamed Jad Aouad
École Polytechnique
IP Paris
Palaiseau, France

Bryan Chen
École Polytechnique
IP Paris
Palaiseau, France

Marceau Leclerc
École Polytechnique
IP Paris
Palaiseau, France

Théo Paquier
École Polytechnique
IP Paris
Palaiseau, France

ABSTRACT

Sparse Attention Vectors (SAVs) [1] enable large multimodal models (LMMs) to perform classification tasks without finetuning and in a highly data-efficient way. In this work, we explore modifications to the SAVs approach to improve both **class discrimination** and **hallucination detection**, including penalty terms for non-target classes, non-linear transformations based on the artanh function, and kernel-based similarity alternatives. We also propose a per-class selection strategy and integrate compare these methods against classical machine learning (ML) approaches. Although our experiments show *comparable* performance to the original SAVs method rather than consistent improvements, these explorations provide valuable insights into the design of discriminative head-selection mechanisms. We evaluate on BLINK [2], NaturalBench [3], a LMM-Hallucination dataset, VLGuard safety [4], EuroSAT [5], Oxford-IIIT Pets [6], and discuss potential applications in low-data domains (e.g. medical, satellite imagery). Our code is available Here.

Figure 1: Examples from **NaturalBench** and **LMM-Hallucination** used for evaluating SAVs performance, against a Zero-shot straight-forward approach. Extracted from [1].

1 Introduction

With the emergence of large Vision-Language Models (VLMs), such as LLaVA-OneVision [7], InternVL [8] and Qwen2-VL [9], researchers and practitioners have been able to address a wide range of open-ended tasks including image captioning, visual question answering, and multi-image reasoning. However, using the same models for *discriminative* tasks—like classification, multiple-choice reasoning, or safety checks—often requires parameter updates (finetuning) and large amounts of labeled data, which may be prohibitive in real-world domains like healthcare, where data can be expensive and limited. Thus, LMMs are not directly suited for foundational discriminative vision-language tasks.

Expanding on pure image classification, Convolutional Neural Networks (CNNs) [10, 11] and Vision Transformers [12] have proven to be state of the art methods, but also do require a large amount of labeled data, even in the context of transfer-learning.

Sparse Attention Vectors (SAVs) [1] offers a compelling alternative: by selecting a small subset of attention heads from a frozen LMM and leveraging these as feature extractors, one can achieve competitive classification performance *without finetuning* and only requiring a number of examples of the order of *20 per label*. This is particularly attractive when the dataset is small or specialized (e.g., medical images, satellite data), and when computational resources or time are limited.

In this work, we revisit the SAVs framework and explore multiple directions to extend or improve it:

1. A **penalty-based similarity score** that favors heads that strongly discriminate one class over others, possibly with a further non-linear addition to bias head selection.
2. **Kernel-based similarity measures** (e.g., RBF, Laplacian) to go beyond the cosine distance.
3. **Per-class head selection strategies** with particular voting mechanisms.
4. A **Classic ML approaches** (e.g. Lasso) to further ground SAVs' relevance.

Although these methods did not consistently beat the original SAVs baseline in our tests, they helped us better understand the SAVs mechanism and its caveats. We describe these modifications in detail, provide quantitative and qualitative observations, and reflect on the scenarios (e.g. few-shot settings, specialized tasks) in which such scoring may be beneficial.

2 Sparse Attention Vectors (SAVs): Recap and Motivation

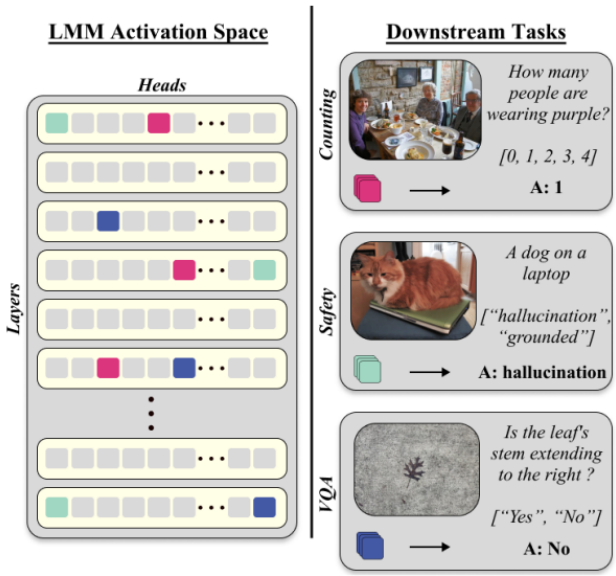


Figure 2: **SAVs Overview.** We illustrate how a large multimodal model's attention heads can be tapped as discriminative features for classification. (Extracted from the original paper [1].)

Original SAVs Pipeline. Given a small sample (e.g., ≈ 20 labeled examples per class) of image-text pairs (x_i, y_i), where $y_i \in \mathcal{C}$ is a class label, the *SAVs* procedure is as follow:

1. **Extract Attention Vectors.** For each labeled (x_i, y_i), store the attention vectors $\mathbf{h}_l^m(x_i)$ from all heads m in all layers l of a frozen LMM, where

$$\mathbf{h}_l^m(x_i) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_m}}\right)V$$

where Q, K, and V are the query, key, and value matrices respectively, and the dimensionality of each head d_m which is given by $\frac{d}{H}$ (the embedding dimension divided by the number of heads). We denote $\mathbf{h}_l^m(x_i)$ as an attention vector for head m in layer l.

This yields a set of attention vectors $\{\mathbf{h}_l^m(x_i) \mid i = 1, \ldots, N\}$ for each head m and layer l.

2. **Compute Class Centroids.** We evaluate each vector's discriminative ability by computing its performance under a nearest class centroid classifier. For each class c, compute the average attention vector for that class and head such that
$$\mu_c^{l,m} = \frac{1}{|N_c|} \sum_{j:y_j=c} \mathbf{h}_l^m(x_j)^T$$
where $N_c = \{j : y_j = c\}$ is the set of indices of examples with label c.
3. **Compute Similarities.** For each input x_i, we compute its cosine similarity to each class centroid head:
$$s_{l,m}(x_i, c) \;=\; \frac{\mathbf{h}_l^m(x_i)^T \, \mu_c^{l,m}}{\|\mathbf{h}_l^m(x_i)\| \, \|\mu_c^{l,m}\|}, \quad \forall\, c \in \mathcal{C}.$$
4. **Assign Discrimination Score.** For each head (l, m), measure how often its predicted class $\hat{y}_{l,m} = \arg\max_c s_{l,m}(x_i, c)$ matches y_i in the labeled set. This score(l, m) is defined as follows:
$$\text{score}(l, m) = \sum_{i=1}^{N} \mathbb{1}\left[\arg\max_{c \in \mathcal{C}} s_{l,m}(x_i, c) = y_i\right]$$
where $\mathbb{1}[.]$ is the indicator function that evaluates to 1 when the condition is true (and 0 otherwise).
5. **Select Top-k Heads.** Keep the few best heads
$$\mathcal{H}_{\text{SAVs}} = \{(l, m) \mid s(l, m) \text{ is among the } k \text{ highest}\}.$$

For inference on a new sample x', each chosen head (l, m) predicts a class label
$$\hat{y}_{l,m} \;=\; \arg\max_{c \in \mathcal{C}} s_{l,m}(x'^T, c),$$
and the final classification is a simple majority vote among those heads:
$$\hat{y} \;=\; \arg\max_{y \in \mathcal{C}} \sum_{(l,m) \in \mathcal{H}_{\text{SAVs}}} \mathbb{1}\big[\hat{y}_{l,m} = y\big].$$

Despite its simplicity, SAVs achieves strong results on tasks ranging from visual question answering to hallucination detection [1], all without finetuning the LMM. This is of special interest when data is scarce or one cannot afford large-scale training.

3 Proposed Extensions

3.1 Penalty-based Score: Enforcing Class Confidence and Inter-Class Separation

In the vanilla similarity $s_{l,m}(x_i, c)$ above, a head can be "sure" about class c yet still produce moderate or high similarity for some other classes c'. Our intuition is that a *truly* discriminative head should simultaneously:

- Have high similarity (close to 1) for the correct class,
- Have low similarity (close to -1) for *all other* classes.

Hence, we propose:
$$\widetilde{s}_{l,m}(x_i, c) \;=\; s_{l,m}(x_i, c) \;-\; \frac{1}{|\mathcal{C}| - 1} \sum_{c' \neq c} s_{l,m}(x_i, c').$$

Denoting $x = s_{l,m}(x_i, c)$ and
$$y = \frac{1}{|\mathcal{C}| - 1} \sum_{c' \neq c} s_{l,m}(x_i, c'),$$
we can think of $\widetilde{s}_{l,m}(x_i, c) = x - y$. If $x \approx 1$, the head is confident about class c, and if $y \approx -1$, it is confident that c' are *not* correct. Subtracting these terms puts heads that are "sure" about one class but ambiguous about others at a disadvantage compared to heads that are "sure" about *exactly one* class.

3.2 Non-linear Mapping: Emphasizing Near-Extreme Values

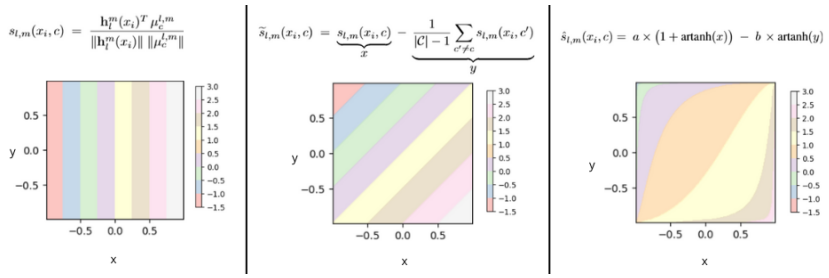


Figure 3: Comparison of non-linear mapping functions emphasizing near-extreme values. From left to right : vanilla SAVs, wrong class penalization, non-linear artanh mapping.

A further idea is to amplify heads that are near ± 1 in the above scoring. We use an *inverse hyperbolic tangent* transform (artanh) to spread out values near ± 1. Specifically, define:

$$\hat{s}_{l,m}(x_i, c) = \; a \times \big(1 + \text{artanh}(x)\big) \; - \; b \times \text{artanh}(y),$$

where $x = s_{l,m}(x_i, c)$ is the similarity for the correct class, and $y = \frac{1}{|\mathcal{C}|-1} \sum_{c' \neq c} s_{l,m}(x_i, c')$ is the average similarity to the other classes. Here, a, b are hyperparameters controlling how strongly we reward (or penalize) near-extreme values on either the correct x or incorrect y class(es) prediction. We will explicit how we performed their selection in Section 4.

Intuition: When a head is "almost certain" ($x \approx 1$ and $y \approx -1$), artanh(± 1) grows very large in magnitude. This pushes that head's score up so that it is more likely to be selected. Heads that produce moderate scores (e.g. $x = 0.3, y = -0.2$) get a less dramatic boost. Thus, we can direct the method to prefer heads that are *very decisive* about their classification — either highly discriminative (similarity close to 1) or strongly penalizing (similarity close to -1), as both cases provide confident information.

To better illustrate this behavior, we visualize in Figure 4 the effect of varying a and b on the curvature of the scoring function. This highlights how different settings modulate the emphasis on near-extreme values, which is crucial for promoting the selection of such decisive heads.

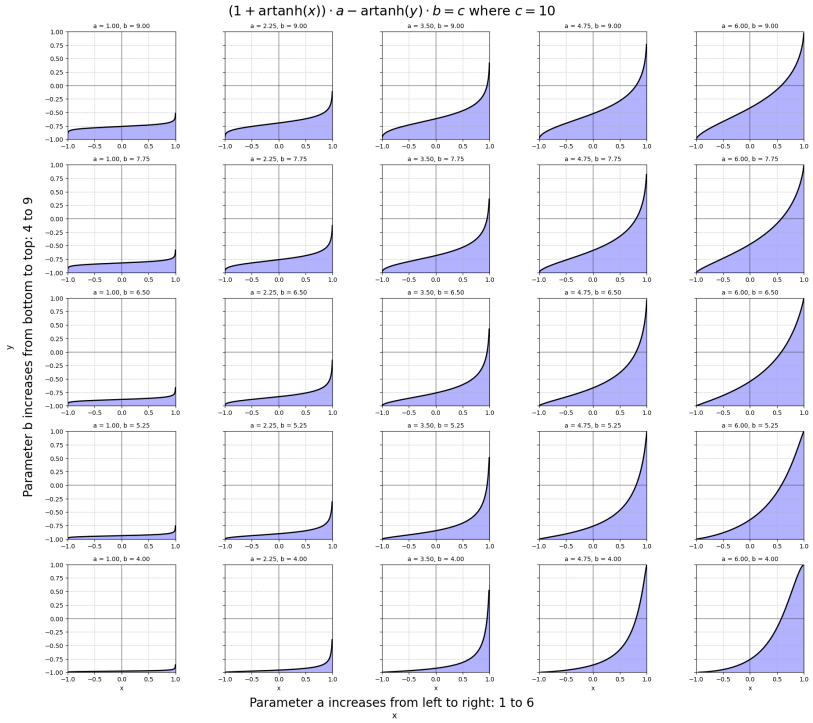


Figure 4: Effect of a and b on the shape of the scoring function using artanh.

3.3 Kernel-Based Similarity Functions

Instead of relying purely on cosine similarity, we explore alternative non-linear similarity metrics, aiming to capture more nuanced distances between a head vector $\mathbf{h}_l^m(x)$ and class centroid $\mu_c^{l,m}$. This modification to [1] was also motivated by the simplicity of its implementation. Specifically, we benchmark:

- **RBF/Gaussian Kernel**: $k(\mathbf{v}, \mathbf{u}) = \exp(-\gamma\|\mathbf{v} - \mathbf{u}\|_2)$
- **Laplacian Kernel**: $k(\mathbf{v}, \mathbf{u}) = \exp(-\gamma\|\mathbf{v} - \mathbf{u}\|_1)$
- **Sigmoid Kernel**: $k(\mathbf{v}, \mathbf{u}) = \tanh(\gamma\mathbf{v}^\top\mathbf{u} + \beta)$

In these cases, instead of cosine similarity, we replace $s_{l,m}(x, c)$ with $k(\mathbf{h}_l^m(x), \mu_c^{l,m})$. Trying out these kernels allow for gaining insight as to how we can navigate the spaces generated by the attention heads of specific models.

Because we are using these kernels to simply "sort" attention heads by importance, we don't need to carefully select the hyperparameters above. We will go on and lead our experiments using $\gamma = 1$ and $\beta = 0$.

3.4 Per-class Head Selection

In the original SAVs framework (see Section 2), head selection is performed by ranking all attention heads based on their overall discriminative ability across all classes. In our per-class head selection approach, we extend this idea by identifying, for each class $c \in \mathcal{C}$, the attention heads that are most effective at detecting features specific to that class. This not only provides a more tailored selection of heads but also offers enhanced interpretability by revealing which heads specialize in recognizing particular classes.

Given a set of labeled examples $\{(x_i, y_i)\}_{i=1}^N$ with $y_i \in \mathcal{C}$, we first compute the cosine similarity for each head (l, m) and each class c as in the original SAVs method:

$$s_{l,m}(x_i, c) \;=\; \frac{\mathbf{h}_l^m(x_i)^T\,\mu_c^{l,m}}{\|\mathbf{h}_l^m(x_i)\|\,\|\mu_c^{l,m}\|},$$

where the class centroid $\mu_c^{l,m}$ is defined as

$$\mu_c^{l,m} \;=\; \frac{1}{|N_c|}\sum_{j:\,y_j=c}\mathbf{h}_l^m(x_j),$$

with $N_c = \{j \mid y_j = c\}$ denoting the set of indices corresponding to class c.

For per-class evaluation, we define a class-specific head score that counts the number of correctly classified samples for class c using head (l, m):

$$\text{score}_{l,m}(c) \;=\; \sum_{i:\,y_i=c}\mathbb{1}\Big[\arg\max_{c'\in\mathcal{C}} s_{l,m}(x_i, c') = c\Big],$$

where $\mathbb{1}[\cdot]$ is the indicator function. For each class c, we then select the top-k_c heads with the highest $\text{score}_{l,m}(c)$ values:

$$\mathcal{H}_{\text{SAVs}}(c) \;=\; \Big\{(l, m)\;\Big|\;\text{score}_{l,m}(c) \text{ is among the top } k_c \text{ for class } c\Big\}.$$

When classifying a new sample x', we propose two alternative voting schemes based on the per-class head selection:

1. Class-favored Voting. For each class c, we compute an aggregated vote count:

$$\mathcal{C}(c) \;=\; \sum_{(l,m)\in\mathcal{H}_{\text{SAVs}}(c)}\mathbb{1}\Big[\arg\max_{c'\in\mathcal{C}} s_{l,m}(x', c') = c\Big],$$

and then assign the label by choosing the class with the highest vote:

$$\hat{y} \;=\; \arg\max_{c\in\mathcal{C}}\mathcal{C}(c).$$

2. Max-Similarity Voting. Alternatively, we can sum the cosine similarities directly for each class:

$$\mathcal{S}(c) \;=\; \sum_{(l,m)\in\mathcal{H}_{\text{SAVs}}(c)} s_{l,m}(x', c),$$

and predict the class as

$$\hat{y} \;=\; \arg\max_{c\in\mathcal{C}} \mathcal{S}(c).$$

Both voting mechanisms leverage the fact that different heads may be more specialized for different classes, allowing for a more nuanced decision process. The class-favored voting aggregates discrete head decisions, while max-similarity voting directly utilizes the strength of the cosine similarities, providing a continuous measure of confidence.

3.5 Logistic Regression and Group Lasso

Motivation. The previous strategies were based on the initial score and the idea that the aggregation procedures could help to identify discriminative heads on which one could base a decision.

As proven in [1], this approach does perform better than alternative and more established methods, such as linear probing the last VLM layer or performing a nearest neighbour vote.

A method from traditional machine learning that is relevant here is Lasso-penalized logistic regression. In this approach, an l_1 penalty is imposed on the logistic loss, which not only aids in fitting a classifier (extensible to multiclass problems through multinomial logistic regression or one-vs-all strategies) but also intrinsically performs feature selection by driving non-informative coefficients to zero [13].

Expanding on this concept, the sparse group lasso (SGL) [14] introduces a composite regularization that combines group-level and individual-level sparsity. This dual penalty is particularly advantageous when predictors have a natural grouping (e.g., different heads in a neural network), as it enables the selective removal of entire groups while also refining the choice of features within each group. Such hierarchical selection enhances interpretability and helps mitigate overfitting in high-dimensional settings.

For our purposes, we chose to try both methods on the the last L layers, and used various coefficients C for the respective penalties of both Lasso and SGL. We will come back to the choice of those hyperparameters in Section 4. For SGL in particular, groups are defined to be attention heads, all of dimension 128.

4 Experiments

We tested our penalty-based and non-linear scores on the same benchmarks used in the original SAVs paper (plus a few additional tasks), comparing to the baseline SAVs approach. We used two foundation models for extracting attention heads:

- **LLaVA-OneVision** [15], a 7B-parameter open-source VLM specialized for multi-image and single-image tasks.
- **Qwen2-VL** [16], another advanced VLM capable of dynamic resolution handling.

These models allow for a direct comparison to [1]. In order to perform our experiments in a resource-efficient way, we ran both models on all train and test data from our benchmarks (see following Section 4.1) and saved all attention heads' activations to then easily and locally run our experiments. Harvesting all of these activations for both models took 8 hours on a single Nvidia L40 GPU.

4.1 Benchmarks and Setup

Following [1], all training procedures are lead with exactly 20 examples per class for each benchmark. The dataset sizes provided below reflect the sizes of the respective testing sets. All of these are common vision-language benchmarks that were created to evaluate models on various tasks, each reformulated as a discriminative task (e.g., multiple-choice or classification) for the purpose of our experiments. All the formatting that took place was done precisely following what is described in [1]. We provide up to 4 choices (A, B, C or D), with one being the true answer while the other 3 being possibly taken at random in the remaining wrong classes if there are more than 3 such classes (see *image classification* below).

Safety. **LMM-Hallucination** [17] holds pairs of images and statements to classify in one of two of the following : *hallucinating* or *grounded*, and holds $1,817$ examples. **VLGuard** [4] is another binary classification task (safe vs. unsafe) focusing on identifying harmful or disallowed content in the image and/or text prompt. For instance, unsafe examples can be related to sexual, hateful, or intrusive content. The test set holds $1,116$ examples.

Visual Question Answering. **BLINK** [2] contains 14 sub-datasets visual tasks (e.g. multi-view reasoning, similarity comparison, localization) in multiple-choice format. These tasks are designed to be easily tackled by any human child, while being challenging for VLMs. For each question, we treat the possible choices as distinct classes. Each of the sub-tasks is somewhat particular, involving possibly several images, classification on 2 to 4 labels, and with varying example sizes (typically around 100). As a result, we chose to take the average of accuracies on all tasks to define our so-called BLINK "accuracy". Because of this set-up, this dataset wasn't used in our validation procedure, and only served as a testing benchmark. **NaturalBench** [3] holds question answering with two images and two questions for each "group" of examples. These are typically highly complex images and proposed descriptions, that often would require a human subject to take some time to answer. Following [1] we measure question accuracy (correctly answered a question for both images), image accuracy (correctly answered both questions for an image) and group accuracy (all pairs correct). Our test size accounts for $9,500$ groups of 4 image-text pairs.

Image classification. As SAVs is designed to tackle discriminative tasks, we can evaluate it on "native" classification benchmarks. **EuroSAT** [5] is a 10-class satellite land-use classification dataset of small and blurry images. Our test size comprises $20,250$ examples. Finally, **Oxford-IIIT Pets** [6] provides a classification task onto a large label set of 37 cat/dog breeds. As a result, its training set in our set up is the largest, with 740 examples, leaving $6,650$ test samples.

4.2 Hyper-parameter tuning and evaluation approach

Sections 3.2 and 3.5 involve hyperparameters that we need to find without biasing our results. To that end, we chose to use 25% of the sizable benchmarks (all but BLINK) as a validation set. We then took the average accuracy on this validation set to select the "best" configuration in each case, attributing equal weight to all validation benchmarks. As a result, we compare all of our final results on the remaining test data.

To compare the method from [1] to our additions in a fair way, all modified versions of the similarity computations will be assessed on the 20 attention heads they select.

Precisely, we tried the following hyper parameters for each method:

- **Artanh non-linearity** (3.2) on top of the penalized SAVs. We chose to take 6 values of a and b evenly spread out between 0.3 and 1.5, as it allowed for a diverse set of level contour lines.
- **Linear ML approaches** (3.5).
  - We tried fitting both Lasso and SGL using the $L = 1$ to 4 last layers of attention heads. This choice is motivated by the fact that we still want to find a somewhat "light" approach to compare to SAVs, and because both models we are benchmarking on have 28 heads of dimension 128 per layer. This means that for 4 layers, we still are fitting such linear models on $14,336$ features.
  - For both the Lasso and SGL penalties, we chose to try (inverse) weighting coefficients that went from $C = 10^{-5}$ to 10^{2} in order to have a wide window of penalization strengths.

4.3 Results and Observations

We present our main results in Table 1.

The first thing that should be addressed is that no single method stands out across all benchmarks, for any of the two models. However some insightful observations can be made.

Penalty-based scoring matches the vanilla SAVs approach (the "baseline" cosine similarity) at best, and sometimes degrades it. In fact for LLaVA-Onevision, scores are identical with and without this added layer of complexity. On Qwen2-VL, applying the artanh non linearity tweak presented in Section 3.2 does slightly impact the results, but not in any significant way. The main interesting result that stands out from these observations is that the base SAVs method already allows to select highly discriminative heads, and doesn't require further tweaking in the ranking of attention heads.

Model	Safety		VQA				Image Cls.	
	MHalu	VLGuard	BLINK	Natural Bench			EuroSAT	Pets
				Text	Image	Group		
LLaVA-OV-7B-ZS								
Baseline - Cosine Similarity								
SAVs (from [1])	80.8	94.3	51.8	60.3	62.3	35.1	86.7	97.0
SAVs (our reproduction)	82.0	96.4	46.3	72.6	72.6	54.0	**83.4**	**97.9**
Cosine similarity								
+penalty (3.1)	82.0	96.4	46.3	72.5	72.6	53.9	**83.4**	97.8
+penalty, artanh (3.2)	82.0	96.4	46.3	72.5	72.6	53.9	**83.4**	97.8
Alternative similarities (3.3)								
Gaussian	81.8	95.9	46.0	**73.6**	**73.5**	**55.5**	81.2	97.7
Laplacian	**82.5**	96.6	47.8	71.7	72.1	53.2	81.2	**97.9**
Sigmoid	79.8	**97.0**	44.2	72.5	72.7	55.3	81.1	95.5
Linear Methods (3.5)								
Lasso	80.9	86.4	**50.0**	68.4	68.6	46.7	65.1	97.2
SGL	80.6	86.9	45.0	69.3	69.4	48.1	15.6	05.3
Qwen2-VL-7B-ZS								
Baseline - Cosine Similarity								
SAVs (from [1])	85.1	96.0	47.2	57.6	60.9	32.3	79.9	98.1
SAVs (our reproduction)	82.6	96.3	47.5	74.3	75.2	56.1	74.3	98.5
Cosine similarity								
+penalty (3.1)	82.6	96.2	47.4	74.3	75.2	56.1	74.2	98.5
+penalty, artanh (3.2)	83.2	96.2	49.3	74.0	74.9	55.2	74.2	98.5
Alternative similarities (3.3)								
Gaussian	83.5	**96.7**	47.8	**75.3**	**76.5**	58.4	72.5	98.6
Laplacian	83.3	96.5	48.5	74.3	75.5	56.9	71.1	**98.7**
Sigmoid	82.0	94.0	41.1	75.1	76.4	**59.1**	70.8	98.4
Linear Methods (3.5)								
Lasso	**84.9**	91.8	**49.9**	73.9	74.7	56.5	**81.2**	98.3
SGL	84.8	92.1	44.0	73.7	74.4	55.3	19.6	05.6

Table 1: Results evaluation on Safety, Visual Question Answering (VQA), and Image Classification benchmarks. The best result for each generative model is shown in **bold** and the second best is underlined. We grayed-out the results taken from [1], that are just here for reference, as they were obtained from another test set.

Kernel-based similarities reliably produce satisfying results, that gravitate around the cosine similarity's. It is true that depending on the model and the benchmark these alternative sometimes have the edge over the baseline, but it has to be underlined that there is no clear pattern of improvement. Has a result, selecting the "best" similarity would require an added validation step on top of the SAVs approach, which in terms require more labeled examples, thus reduces the power of this data-efficient method. All in all, any similarity that was attempted is a valid candidate for use in practice.

Linear methods did slightly outperform the SAVs-based approaches in some particular settings, but overall proved to be at best comparable - for Lasso - and most often inferior to it. This is particularly true for the SGL on classification benchmarks, that respectively hold 10 and 37 different classes, showing this method is unfit for tasks with sizeable amounts of possible labels.

The hyperparameters selected in the validation procedure (when relevant) were as follows. For the artanh non linearity applied to our "penalized" SAVs (see Section 3.2), we observed very similar and oftentimes identical results with most values of a and b. This further shows that SAVs selects the relevant attention heads off-the-shelf and doesn't need such a trick.

For the linear ML methods, the hyperparameters selected are shown in Table 2. As a reminder, C is the (inverse) penalization weight and L is the number of last layers taken as features.

Method	Model	C	L
Lasso	*LLaVA-OV-7B-ZS*	100	4
SGL	*LLaVA-OV-7B-ZS*	100	4
Lasso	*Qwen2-VL-7B-ZS*	0.001	2
SGL	*Qwen2-VL-7B-ZS*	0.0001	4

Table 2: Hyperparameters selected for both methods and models.

The most striking observation is that both models are tremendously different in the regularization they required. *Qwen2-VL* required a much higher penalization, which indicates that its heads hold a more noisy information than *LLaVA-OV*'s. Most importantly, it has to be underlined that the maximum number of layers $L = 4$ was picked most often. We still chose to limit our parameter search to this bound, as it already stands an "informative" advantage against SAVs-based methods, which limit themselves to 20 heads, i.e. almost 6 times less information.

4.3.1 Per-class head selection

In our experimental setup, we chose to select 25 attention heads per class because preliminary testing suggested that this was the optimal number for the task. Importantly, we used the original scoring function from the paper without incorporating any of our proposed modifications. The experiments with per-class head selection yielded very disappointing results, consistently under-performing the baseline approach proposed in the original paper. For example, while binary classification tasks achieve accuracies exceeding 80%, the performance on benchmarks with many classes drops dramatically (e.g., only around 4% accuracy on Oxford-IIIT Pets with more than 30 classes, reminding of the results with SGL).

A particularly interesting observation is that altering the voting mechanism has a significant impact on performance. We evaluated two distinct voting strategies : **Class-favored Voting** and **Max-Similarity Voting**.

Our results indicate that the max-similarity voting mechanism is far superior to the class-favored voting strategy.

Figure 5 illustrates the outcomes for LLaVA-OneVision, with the left panel showing results obtained via class-favored voting (stock) and the right panel demonstrating the gains achieved with max-similarity voting. Similarly, Figure 6 shows comparable results for Qwen2-VL.

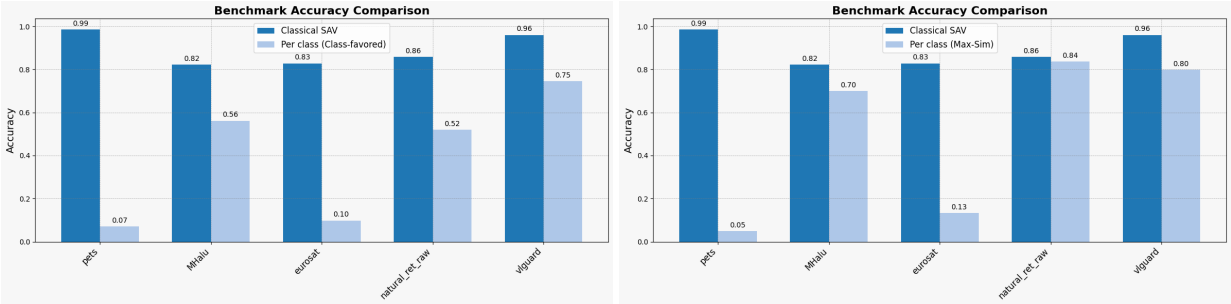


Figure 5: LLaVA-OneVision results: **Left** - Class-favored Voting (stock), **Right** - Max-Similarity Voting.

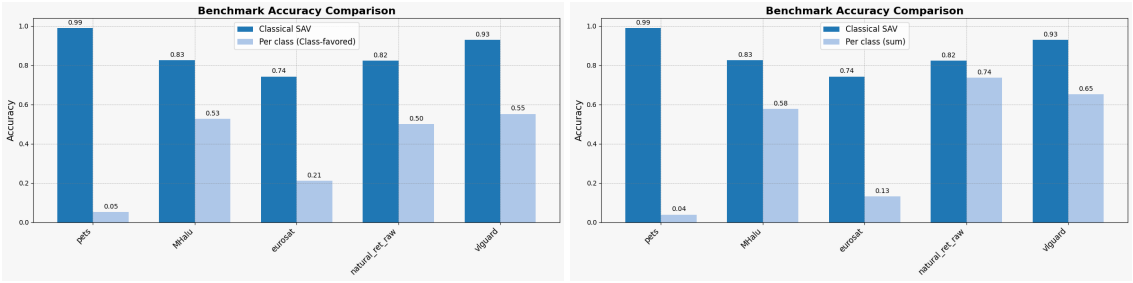


Figure 6: Qwen2-VL results: **Left** - Class-favored Voting (stock), **Right** - Max-Similarity Voting.

Beyond these accuracy comparisons, we also examined the distribution of selected attention heads across classes. Figure 9 shows two visualizations of the selected heads: one for the EuroSAT benchmark and one for the LMM-Halucination benchmark. These plots reveal that the chosen heads vary across classes, indicating that no common subset of heads emerges for all classes. Additional visualizations are available in Appendix A.

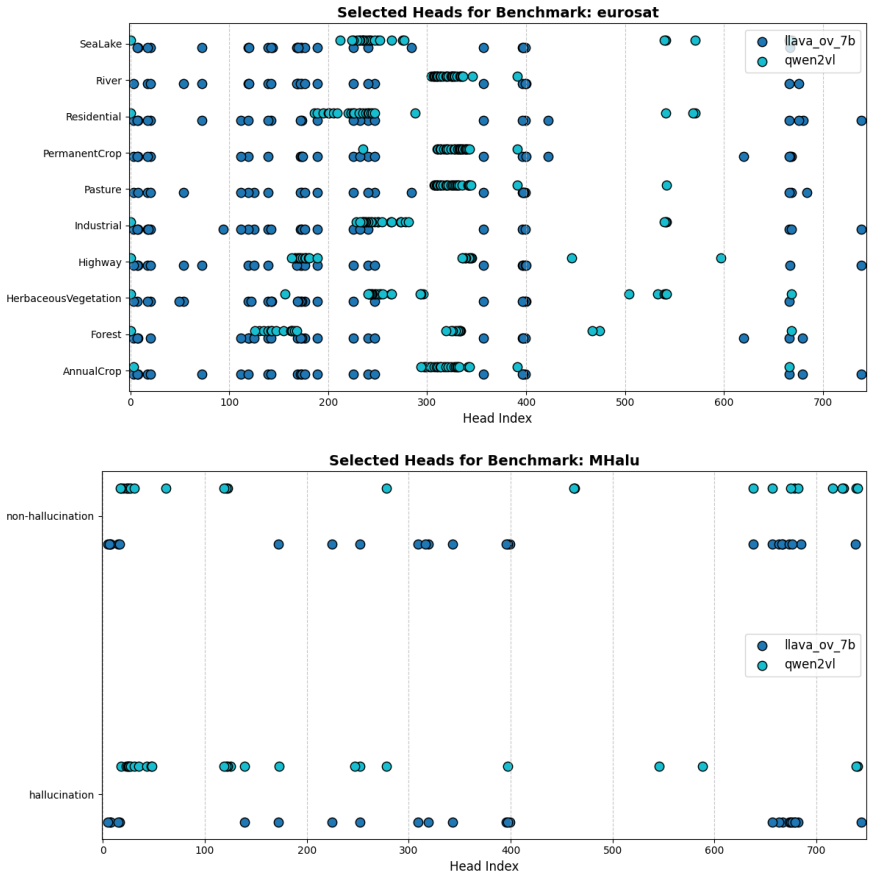


Figure 7: Visualization of selected heads: **Top** - EuroSAT, **Bottom** - LMM-Hallucination. Note that the selected heads differ across classes.

In summary, although it appeared natural to choose heads on a per-class basis, our findings suggest that heads with general discriminative power yield better performance overall. It is plausible that a modified scoring function, one that more strongly favors the discrepancy between classes, could lead to improved results in this framework. Another area of improvment would be to choose the number of heads per class with a more informed approach.

5 Discussion and Conclusion

Although we did not observe consistent performance improvements, our study provides insight into various ways of modifying or extending SAVs. Exploring various similarities can be relevant, but the core SAVs strategy is simple yet robust, thus doesn't need complexifying.

All in all, we further underlined the power of Sparse Attention Vectors, a great strategy that allows for powerful discriminative capacities across various domains, making this method a serious contender in fields such as medical and satellite imaging, where labeled data suffer from scarcity.

Acknowledgments

We would like to express our sincere gratitude to **Prof. Éric Moulines** for his mentoring and his support during this project. Lastly, we extend our thanks to **Prof. El Mahdi El Mhamdi** who supervised the Research Seminar class.

References

[1] C. M. et al., "Sparse Attention Vectors: Generative Multimodal Model Features Are Discriminative Vision-Language Classifiers," 2025.

[2] X. F. et al., "BLINK: Multimodal Large Language Models Can See but Not Perceive," 2024.

[3] B. L. et al., "NaturalBench: A Compositional Benchmark for Vision-Language Models," 2024.

[4] Y. Z. et al., "Safety Fine-Tuning at (Almost) No Cost: A Baseline for Vision Large Language Models," 2024.

[5] P. H. et al., "EuroSAT: A Novel Dataset and Deep Learning Benchmark for Land Use and Land Cover Classification," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 12, no. 7, pp. 2217–2226, 2019.

[6] O. P. et al., "Cats and Dogs," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3498–3505, 2012.

[7] B. Li, Y. Zhang, D. Guo, R. Zhang, F. Li, H. Zhang, K. Zhang, P. Zhang, Y. Li, Z. Liu, and C. Li, "Llava-onevision: Easy visual task transfer," 2024.

[8] Z. C. et al., "How Far Are We to GPT-4V? Closing the Gap to Commercial Multimodal Models with Open-Source Suites," 2024.

[9] P. Wang, S. Bai, S. Tan, S. Wang, Z. Fan, J. Bai, K. Chen, X. Liu, J. Wang, W. Ge, Y. Fan, K. Dang, M. Du, X. Ren, R. Men, D. Liu, C. Zhou, J. Zhou, and J. Lin, "Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution," 2024.

[10] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, pp. 1097–1105, 2012.

[11] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[12] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021.

[13] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, pp. 267–288, 12 2018.

[14] N. Simon, J. Friedman, T. Hastie, and R. Tibshirani, "A sparse-group lasso," *Journal of Computational and Graphical Statistics*, vol. 22, 04 2013.

[15] B. L. et al., "LLaVA-OneVision: Easy Visual Task Transfer," 2024.

[16] P. W. et al., "Qwen2-VL: Enhancing Vision-Language Model's Perception of the World at Any Resolution," 2024.

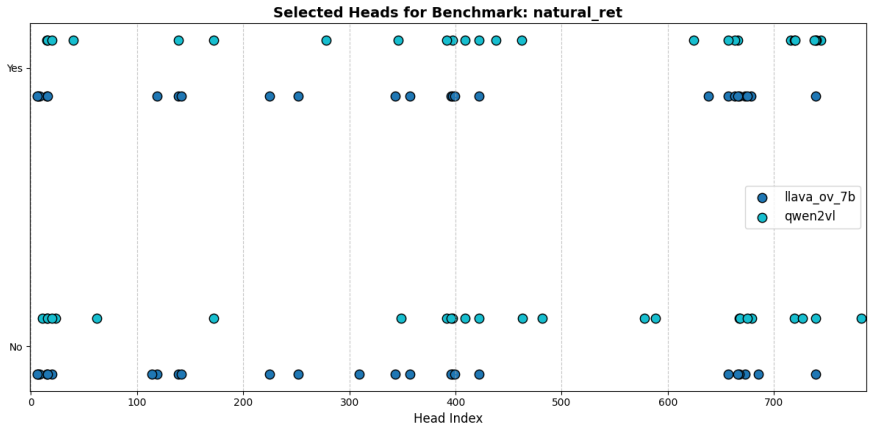
[17] X. Chen, C. Wang, Y. Xue, N. Zhang, X. Yang, Q. Li, Y. Shen, L. Liang, J. Gu, and H. Chen, "Unified hallucination detection for multimodal large language models," 2024.

A Appendix - additional visualizations for per-class voting strategies

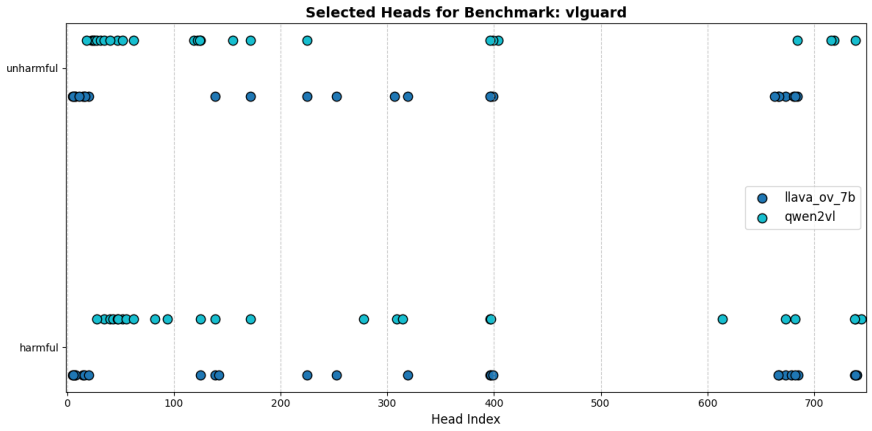
In this appendix, we provide additional visualizations of the selected attention heads for each class. While we have already discussed this aspect earlier in the paper, we include these figures to offer more context and clarity.

Unlike the original paper, which presents a different approach, our method selects attention heads on a per-class basis, meaning that each class has its own set of preferred heads. From the three visualizations provided, we observe that certain attention heads are consistently selected across multiple classes. However, each class also retains a unique set of heads, supporting our initial intuition about class-specific attention patterns.

Despite these findings aligning with our expectations, it is important to note that the overall performance of the method remains suboptimal. This suggests that while the selection of attention heads captures some meaningful structure, it does not necessarily translate into better model effectiveness.



(a) Selected heads: NaturalBench



(b) Selected heads: VLGuard Safety

Figure 8: Visualization of selected heads across VLGuard and NaturalBench benchmarks

Figure 9: Visualization of selected heads: Oxford-IIIT Pets