# LVLM-Aided Alignment of Task-Specific Vision Models

## Supplementary Material

## A. Additional Experimental Results

In this section, we present ablations for the use of open-source LVLMs, the selection of the number of clusters $J$ in PPEPS-WGM, and the effect of LVLM-aided alignment on the robustness of vision models to global perturbations. Further, we provide additional insights on the group accuracy gains on the medical datasets and on the effect of the alignment on the embedding space for DecoyMNIST.

### A.1. Open-Source LVLMs

In Table 1, we compare the verdict accuracy of the OpenAI GPT models used in the main paper with recent open-source alternatives. Verdict accuracy is the proportion of clusters that overlap spurious features and are then actually correctly classified as spurious by the Critic & Judge pair. As open-source models, we evaluate Llama-4-Scout (*Llama-4-Scout-17B-16E-Instruct*) [7] as both Critic and Judge, and Qwen3 models [9] using *Qwen3-VL-235B-A22B-Thinking* as the Critic and *Qwen3-235B-A22B-Instruct-2507* as the Judge. While the Qwen3 models under-perform the larger OpenAI models, Llama-4 achieves a verdict accuracy comparable to GPT-4o. This demonstrates that our method does not rely on any specific proprietary LVLM and can also be effectively applied with open-source models. Together with the other measures discussed in the main paper, this further enhances the accessibility of LVLM-VA.

Table 1. Comparison of the verdict accuracy for different Critic & Judge pairs including open source models on the knee alignment samples.

| **GPT-5** | **Llama4** | **GPT-4o** | **Qwen3** | **GPT-4o-mini** |
|---|---|---|---|---|
| 1.00 | 0.88 | 0.87 | 0.46 | 0.42 |

### A.2. Number of Clusters

When choosing the number of clusters for our Positive Predictive Effect Probabilistic Segmentation via Weighted Gaussian Mixtures (PPEPS-WGM), it is essential to ensure that there are theoretically enough clusters to separate spurious from core features, thereby avoiding ambiguity in the critic's responses. While in principle there is no strict upper bound on the number of clusters, excessively increasing this number leads to the fragmentation of very small regions of core or spurious features, which in turn makes the task more difficult for the LVLM Critic & Judge pair. The segmentation maps $C$ for varying numbers of clusters in Figure 1 illustrate that, for the skin lesion dataset and its spurious features, increasing the number of clusters beyond seven does not yield additional benefits. In addition to the visual guidance, we

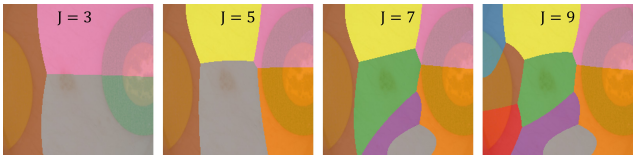


Figure 1. Illustration of clustering on the skin lesion dataset for four different choices of the number of clusters $J$. With $J = 3$, the pink and grey clusters span large regions of both the spurious bandage and the lesion. Increasing to $J = 7$ yields a more focused cluster on the true lesion area, whereas a further increase to $J = 9$ offers no substantial additional improvement.

conducted a short experiment to show that the verdict accuracy of the Critic & Judge pair is quite robust to the number of clusters $J$ making it a non-critical hyperparameter. For the knee dataset, the verdict accuracy for unambiguously classifying clusters changes only marginally between five and nine clusters. The verdict accuracy classifying the relevancy of a certain cluster is directly reflected in the binary correction masks (relevant yes/no) used for the alignment step and suggests in practice to pick a reasonably large number for $J$. Following insights from previous experiments, we fixed the number of clusters to seven across all medical datasets.

Table 2. Comparison of the verdict accuracy for different cluster sizes $J$.

| $J$ | **3** | **5** | **7** | **9** |
|---|---|---|---|---|
| $\text{Acc}_{\text{Verdict}}$ | 0.68 | 0.84 | 0.86 | 0.86 |

### A.3. Absolute Performance on Medical Datasets

Table 3 shows the absolute Average Group Accuracy (AGA) and Worst Group Accuracy (WGA) for the original model $f$ and for all shortcut mitigation approaches corresponding to the improvements visualized in Figures 9 and 10 (main paper). These results further show that the apparently larger improvement in WGA achieved by SUBG is mainly driven by a substantial reduction in overall accuracy.

### A.4. Effect of Alignment on Embedding Space

When observing the embeddings for the original and the aligned model on the test set (Fig. 2) for DecoyMNIST, LVLM-VA reduces the model's reliance on simple shortcuts leading to better class discrimination.


Figure 2. Test set embeddings of the DecoyMNIST MLP model before (left) and after (right) the LVLM-VA alignment step. The clusters are more separated, as the model is less affected by spurious shortcuts.

### A.5. Increased Robustness to Global Perturbation

The main scope of LVLM-VA is to reduce reliance on spurious features present during training by aligning the task model's attribution with human expectations. Even though for global shortcuts, attribution regions may be less spatially separable and segmentation may not well isolate the spurious features, discouraging attribution to generally irrelevant regions can still improve robustness in practice. We explicitly evaluate the effect of LVLM-VA based alignment beyond strictly localized confounders. We apply MedMNIST-C corruptions [1] to the skin data at test time and measure the corruption-induced accuracy drop, $\Delta Acc = Acc_{\text{clean}} - Acc_{\text{corr}}$. We report how the alignment via LVLM-VA reduces this drop by steering the model to focus on relevant features during training, $\Delta Acc_f - \Delta Acc_{f,\text{aligned}}$. Across 7 seeds, LVLM-VA significantly reduces the drop ($p < 0.05$, Wilcoxon signed-rank) for representative global corruptions that may occur in practice: GammaCorrection (+0.025), FocusBlur (+0.022), and Contrast (+0.055).

## B. Details on Experiments

In the following, we provide additional information about the conducted experiments.

### B.1. Training Details

Below, we describe the training settings used in our experiments, including the chosen hyperparameters and the computational environment.

For all experiments, we generated explanation maps $\Phi(x, y, f)$ using DeepLiftShap [6], as implemented by Kokhlikyan et al. [4], with default parameters. The background dataset for DeepLiftShap consists of 25 samples randomly drawn from the training set. Random seeds were set to $\{0, \dots, 6\}$, and the $\gamma$ parameter within the RRR loss was fixed to 0 across all experiments. Furthermore, for all datasets, one batch comprises of $I_{x_a} = 8$ alignment samples and of $I_{x_s} = 64$ training samples, resulting in a total batch size of $I = 72$.

**Computational Environment.** All experiments were implemented in Python 3.11.7 and used PyTorch 2.1.1 to train the models. We used the official OpenAI Python package (version 1.13.3) to access the GPT models. For the open-source models, we relied on inference providers available through HuggingFace. Training and fine-tuning were carried out on a machine equipped with an NVIDIA RTX A6000 GPU and an Intel Xeon Gold 5418Y CPU with eight cores and 48GB of RAM.

**DecoyMNIST.** For the DecoyMNIST dataset, we utilize the implementation provided by Ross et al. [8], which adapts the well-known MNIST dataset [5]. We train the model during both the initial training phase and the alignment step for 1000 epochs each, using a learning rate of $10^{-5}$ and the Adam optimizer [3]. The classifier is a multilayer perceptron (MLP) with a single hidden layer of size 256. The training set comprises 48 000 samples, and the test set contains 10 000 samples.

**Medical Datasets.** For both medical datasets, we train a ResNet50 model [2] with two output units. The original training phase and the subsequent alignment step are each run for 100 epochs with a learning rate of $10^{-5}$ using the Adam optimizer. To determine the optimal value of $\lambda$, we perform a hyperparameter search, analogous to that used for DecoyMNIST, over a logarithmic range from 1 to $10^5$. The statistical significance of improvements over the original model (before the alignment step) is assessed using a Wilcoxon signed-rank test.

### B.2. Used Prompts

Below, we list all prompts used for the Critic & Judge pair. For all three datasets, we provide the Critic and Judge prompts, as well as the human specifications for the respective classes. The expressions `<label>` and `<class_description>` are instance-wise placeholders referring to the ground-truth class of the alignment sample, whereas `<cluster_colors>` and `<num_clusters>` denote the chosen colour set and the number of clusters, which remain fixed throughout the task.

## References

[1] Francesco Di Salvo, Sebastian Doerrich, and Christian Ledig. Medmnist-c: Comprehensive benchmark and improved classifier robustness by simulating realistic image corruptions. *arXiv preprint arXiv:2406.17536*, 2024. 2

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 2

[3] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[4] Narine Kokhlikyan, Vivek Miglani, Miguel Martin, Edward Wang, Bilal Alsallakh, Jonathan Reynolds, Alexander Melnikov, Natalia Kliushkina, Carlos Araya, Siqi Yan, and Orion Reblitz-Richardson. Captum: A unified and generic model interpretability library for pytorch, 2020. 2

[5] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 2

[6] Scott M. Lundberg and Su-In Lee. A unified approach to interpreting model predictions. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, page 4768–4777, Red Hook, NY, USA, 2017. Curran Associates Inc. 2

[7] Meta AI. Introducing LLaMA 4: Advancing multimodal intelligence. https://ai.meta.com/blog/llama-4-multimodal-intelligence/, 2024. 1

[8] Andrew Slavin Ross, Michael C. Hughes, and Finale Doshi-Velez. Right for the right reasons: training differentiable models by constraining their explanations. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, page 2662–2670. AAAI Press, 2017. 2

[9] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025. 1

| Method | Knee Radiographs | | Skin Lesions | |
|---|---|---|---|---|
| | AGA | WGA | AGA | WGA |
| DFR | $0.5714 \pm 0.0113$ | $0.0257 \pm 0.0270$ | $0.6871 \pm 0.0039$ | $0.1771 \pm 0.0482$ |
| JTT | $0.5693 \pm 0.0271$ | $0.0471 \pm 0.0423$ | $0.7239 \pm 0.0112$ | $0.3100 \pm 0.0898$ |
| SUBG | $0.4868 \pm 0.0148$ | $0.3457 \pm 0.0565$ | $0.6375 \pm 0.0242$ | $0.5086 \pm 0.0398$ |
| LVLM-VA | $0.5954 \pm 0.0147$ | $0.1914 \pm 0.0631$ | $0.7493 \pm 0.0199$ | $0.4457 \pm 0.0894$ |
| original | 0.5875 | 0.0300 | 0.7400 | 0.2300 |

Table 3. Average Group Accuracy (AGA) and Worst Group Accuracy (WGA) for the original model $f$ and after the shortcut mitigation steps, for both medical datasets. The results are averaged over seven random seeds (mean ± std).

The first image is the original input image of class `<label>`, identifiable as `<class_description>`.
The second image shows `<num_clusters>` distinct clusters `<cluster_colors>` derived from the vision model's classification process.
The third image overlays these clusters on the original image to help you locate each cluster's position.

**Analysis Instructions:**
1. First examine the original image to identify the key features of class `<label>`.
2. For each cluster `<cluster_colors>`:
   - Describe which area of the original image it covers.
   - Determine if this area contains relevant features for class `<label>`.
   - Note if the cluster covers adhesive bandages (which are spurious features).

**Important Notes:** Adhesive bandages appear as colorful patches in the original image.
These bandages are typically larger than the relevant skin lesions.
Bandages are spurious features and should not be considered relevant for classification.

**For each cluster, clearly state:** What it covers in the original image; whether it supports correct classification (by covering relevant features); and whether it focuses on spurious features (like bandages).

Figure 3. Skin Lesions - Critic.

Table 4. Skin Lesions - Human Specifications per Class

| Class | Description |
|---|---|
| `benign` | Non-cancerous skin growths that typically present with symmetrical shapes, clear borders, and uniform brown or black coloring with a small diameter, i.e., covering only a small part of the image. |
| `malignant` | Cancerous skin lesions characterized by asymmetry, irregular borders, brown or black color variations with multiple distinct colors, and diameters typically covering a large area of the image. |

Table 5. Knee Radiographs - Human Specifications per Class

| Class | Description |
|---|---|
| `no osteoarthritis` | Healthy knee joints without osteoarthritis, typically characterized by normal joint spacing, absence of bone deformities, smooth joint surfaces, and clear cartilage without visible deterioration. |
| `osteoarthritis` | Knee joints affected by osteoarthritis, characterized by narrowed joint spaces, bone spurs (osteophytes), noticeable cartilage degradation, sclerosis (increased bone density), and visible joint irregularities or deformities. |

You have the task to translate text descriptions into readable JSON format for further processing.
For each of the <num_clusters> clusters <cluster_colors>, you need to determine from the description whether it focuses on a skin lesion or not.

Create a JSON with ”color” and ”verdict” as keys where:
• ”color” must be one of <cluster_colors>
• ”verdict” must be either ”yes” (focuses on lesion) or ”no” (does not focus on lesion)

**IMPORTANT:** Return only valid JSON format with an ”output” key containing a list of exactly <num_clusters> elements, one for each color in <cluster_colors>.

For clarification, here are examples:

Example 1 Input:
Analysis of Clusters:

Grey:
- **Coverage**: Bottom center of the image.
- **Relevance**: Does not cover the skin lesion.
- **Spurious Features**: Covers part of a colored patch (bandage).
- **Conclusion**: Not relevant.
...
Example 1 JSON:
{{
”output”: [
{{
”color”: ”grey”,
”verdict”: ”no”
}}
...]
}}

Figure 4. Skin Lesions - Judge

The first image is the original input image of class <label>, identifiable as <class_description>.
The second image shows <num_clusters> distinct clusters <cluster_colors> derived from the vision model’s classification process.
The third image overlays these clusters on the original image to help you locate each cluster’s position.

**Analysis Instructions:**
1. First examine the original image to identify the key features of class <label>.
2. For each cluster <cluster_colors>:
   - Describe which area of the original image it covers.
   - Determine if this area contains relevant features for class <label>.
   - Note if the cluster covers radiographic markers (letters R or L), which are spurious features.

**Important Notes:** Radiographic markers appear as letters R or L in the original image, indicating right or left leg.
These markers are typically located at the edges of the image, outside the joint area.
Markers are spurious features and should not be considered relevant for classification.

**For each cluster, clearly state:** What it covers in the original image; whether it supports correct classification (by covering relevant features); and whether it focuses on spurious features (like radiographic markers).

Figure 5. Knee Radiographs - Critic

You have the task to translate text descriptions into readable JSON format for further processing.
For each of the <num_clusters> clusters <cluster_colors>, you need to determine from the description whether it is relevant and does not focus on spurious features.
If cluster includes both a part of relevant features and a spurious feature, it should be assigned ”no” (not relevant), unless the covered part of the relevant features is large, then it should be assigned ”yes”.

Create a JSON with ”color” and ”verdict” as keys where:
• ”color” must be one of <cluster_colors>
• ”verdict” must be either ”yes” (might focus on relevant areas of the knee) or ”no” (focuses on spurious features)

**IMPORTANT:** Return only valid JSON format with an ”output” key containing a list of exactly <num_clusters> elements, one for each color in <cluster_colors>.

For clarification, here are examples:

Example 1 Input:
Analysis of Clusters

Grey Cluster
- **Coverage**: Small part of the knee.
- **Relevance**: Might be relevant as it does not cover a spurious feature.
- **Spurious Features**: Does not cover any spurious features.
...
Example 1 JSON:
{{
”output”: [
{{
”color”: ”grey”,
”verdict”: ”yes”
}}
...]
}}

Figure 6. Knee Radiographs - Judge

The following images include hand written digits.
The first image is the original input image of class <label>, which can be recognized as <class_description>.
The second image is a visualization map indicating different clusters considered important for classifying class <label>.
The third image is a visualization map from class <label> overlaid in the original image to support you in relating locations between both images.
In some of the images spurious decoys in the corner are introduced to confuse the model generating the clustered visualization maps and the squares in the corner do not represent one of the classes and should be considered spurious and not be considered within the visualization maps.
The visualization map consist of <num_clusters> clusters with the colors <cluster_colors>, where each cluster describes an area of focus from the original image.
First, examine the original image to identify which parts belong to class <label>.
Then, look at the second image to see the <num_clusters> clusters for class <label>.
For each cluster <cluster_colors>, describe the area where the model focuses to predict class <label>.
Determine whether each cluster is within the boundaries of the class <label> using the third image.
A cluster supports the correct prediction only if it fully or partially focuses on an area within the class <label>.
If a cluster is outside the class structure, clearly state that this cluster does not support the correct prediction.
Do not provide introductory sentences.
Consider the following three examples: ...

Figure 7. DecoyMNIST - Critic

You have the task to translate the responses of a large vision language model (LVLM) into readable JSON format for further processing.
The task of the LVLM was, for each of the <num_clusters> clusters in the second and third image, to identify whether the focus aligns with any part of the digit depicted in a first image or not.
First, you read through the LVLM response. Then you identify for each of the clusters <cluster_colors> whether the focus was on the digit or not.
Then you construct a valid JSON with "color" and "verdict" as keys. The "color" key should strictly be one of <cluster_colors>. The "verdict" key should strictly be either "yes" or "no".
For example, "color": "red" and "verdict": "yes" means that the red cluster did focus on the digit. "color": "blue" and "verdict": "no" means that the blue cluster did not focus on the digit.
**IMPORTANT:** Please make sure to only return in valid JSON format, with the "output" key as a list of JSON. The list should strictly contain <num_clusters> elements, one for every cluster in <cluster_colors>.
For clarification, here are a few examples:
**Examples**
Example 1 Input: Pink: The cluster covers the vertical line of the digit 7 and no decoy square in the corner. It is relevant. Brown: The cluster fully covers a decoy square in the lower right corner. It is not relevant. Yellow: The cluster covers the horizontal line of the digit 7 and no decoy square in the corner. It is relevant. Example 1 JSON: ”output”: [ ”color”: ”pink”, ”verdict”: ”yes” , ”color”: ”brown”, ”verdict”: ”no” , ”color”: ”yellow”, ”verdict”: ”yes” ] . . . ===== END OF EXAMPLES =====

Figure 8. DecoyMNIST - Judge

Table 6. DecoyMNIST - Human Specifications per Class

| Class | Description |
|---|---|
| 0 | A closed, continuous loop with no starting or ending point, representing a circle or oval shape. |
| 1 | A single, straight vertical line, typically with a small base or serif at the bottom. |
| 2 | A curved line starting from the top, forming an open loop to the right, and then descending in a diagonal line toward the left. |
| 3 | Two small, open, curved loops stacked vertically, each curving to the right, connected in the middle. |
| 4 | A vertical line with an angled horizontal line starting from its midpoint, and a diagonal line connecting the top of the vertical line to the bottom of the horizontal line. |
| 5 | A horizontal line at the top connected to a vertical line descending downward, which then curves sharply to the left and forms an open loop. |
| 6 | A vertical line starting from the top, curving downward to the left, and forming a closed loop at the bottom. |
| 7 | A horizontal line at the top connected to a diagonal line that descends toward the left, with no curves or loops. |
| 8 | Two distinct loops one on the top and one on the bottom connected in the middle. |
| 9 | A small loop at the top with a vertical line descending downward from the loop’s right side. |