Explainable Representations in Self-Supervised Learning for Image Understanding

Darpan Gaur, Abhinav Vsk Kompella , Aditya Bacharwar, Naik Avaneesh Radhakrishnan
Indian Institute of Technology Hyderabad
{co21btech11004, es21btech11002, es21btech11003, es21btech11022}@iith.ac.in

Abstract

Self-supervised learning (SSL) has emerged as a powerful alternative to supervised learning, enabling models to learn meaningful representations from unlabeled data. However, these learned representations often lack interpretability, making it difficult to understand their decision-making process. In this project, we aim to investigate the explainability of SSL-based feature representations by integrating explainable artificial intelligence (XAI) techniques. Specifically, we will leverage contrastive learning methods such as SimCLR, MoCO, and SwAV, and employ Grad-CAM, importance-based techniques, and perceptual components-based techniques to interpret learned representations. Our objective is to evaluate the trade-off between explainability and performance, providing insights into how SSL embeddings capture meaningful features. By improving the interpretability of self-supervised representations, we aim to enhance their applicability in critical domains such as healthcare, autonomous systems, and scientific discovery.

1. Introduction

Supervised learning has traditionally been the dominant paradigm in computer vision, requiring large-scale labeled datasets for training deep neural networks. However, acquiring labeled data is costly and time-consuming. Self-supervised learning (SSL) addresses this issue by enabling models to learn feature representations from unlabeled data using pretext tasks. Contrastive learning, particularly SimCLR, has shown remarkable success in this domain by maximizing the similarity between augmented views of the same image while pushing apart different images.

Despite the success of SSL in learning powerful feature representations, a critical limitation remains: lack of interpretability. Unlike supervised learning, where class labels provide a semantic understanding of features, SSL representations are often opaque, making it challenging to understand what the model has learned. Explainability is crucial for:

- Trust and Transparency: Understanding SSL feature representations can improve trust in AI systems, particularly in safety-critical applications.
- Bias Detection: XAI techniques can help reveal biases in SSL models, preventing unintended discriminatory behavior.
- Downstream Task Adaptability: Interpretable SSL representations can improve transferability to various tasks by ensuring that learned features align with meaningful concepts.

2. Problem Statement

Our project aims to bridge the gap between self-supervised learning and explainable AI by investigating how to interpret SSL-learned representations. Specifically, we will:

- Train an SSL model using contrastive learning (e.g., SimCLR[2], MoCo[3]) and SwAV [1].
- Apply Grad-CAM[5] and RELAX[6], explainability techniques to visualize the important regions in SSL feature maps.
- Explore perceptual component-based [7] explainability techniques to align SSL representations with human-interpretable concepts.
- Investigate the impact of different SSL architectures on explainability.

3. Literature Review

3.1. Self-supervised Representation Learning

3.1.1. SimCLR: Simple Framework for Contrastive Learning of Visual Representations

SimCLR[2] is a contrastive self-supervised learning framework that learns visual representations of images without requiring labeled data. The paper explores various components of the model and showcases the importance of each component via numbers from experiments.

The first key component of the SimCLR model is the augmentations it applies, which lead to the positive and

negative pairs needed to perform contrastive learning. The mentioned model stochastically chooses two augmentations from the chosen three augmentations to get the training pairs. The paper also mentions various other augmentations but reinforces the choice of the three augmentations (random crop, random colorization, and random Gaussian noise).

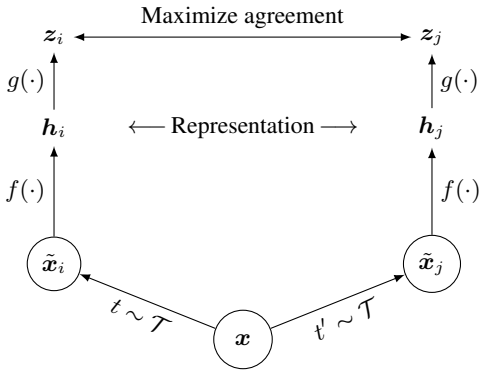


Figure 1. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder network $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss. After training is completed, we throw away the projection head $g(\cdot)$ and use encoder $f(\cdot)$ and representation $\boldsymbol{h}$ for downstream tasks.

Batches of such augmented images are fed to the encoder component where a ResNet architecture was used. The paper then experimented with different projection layers and showed via experimentation that a non-linear projection component worked the best. The contrastive loss is then applied to these projected outputs. SimCLR shows various key components that can be used to extract representations in a self-supervised manner which can be made use of to come up with an explainable self-supervised learning model.

3.1.2. Learning features by Swapping Assignments between multiple Views (SwAV) of an Image

SwAV (Swapping Assignments Between Views) [1] is a self-supervised learning method that introduces a clustering-based approach to learning visual representations without explicit contrastive loss. SwAV directly learns cluster assignments for different augmentations of the same image. This is achieved by using a swapped prediction mechanism, where the model predicts the cluster assignment of one view using the features of another as seen in the loss function:

$$L(\mathbf{z}_t, \mathbf{z}_s) \quad = \quad \ell(\mathbf{z}_t, \mathbf{q}_s) + \ell(\mathbf{z}_s, \mathbf{q}_t), \tag{1}$$

The terms in the loss function representing the swapped

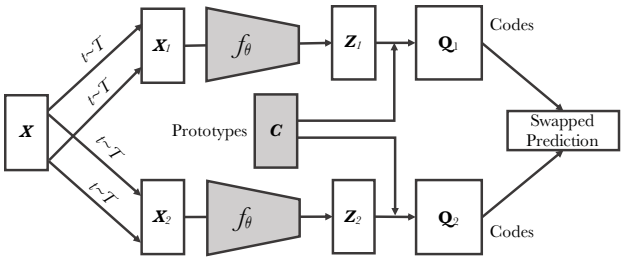


Figure 2. We first obtain "codes" by assigning features to prototype vectors. We then solve a "swapped" prediction problem wherein the codes obtained from one data augmented view are predicted using the other view. Prototype vectors are learned along with the ConvNet parameters by backpropagation.

prediction terms are defined as the cross-entropy loss between the code and the probability obtained by taking a softmax of the dot products of z_i and all prototypes in C (set of learned prototypes). Some improvements that the SwAV paper proposes firstly is computing the codes in an online fashion and secondly is the choice of another image augmentation, namely Multi-crop. In the multi-crop strategy we use two standard resolution crops and sample additional low resolution crops that cover only small parts of the image because comparing random crops of an image plays a central role by capturing information in terms of relations between parts of a scene or an object.

3.1.3. MoCo: Momentum Contrast for Unsupervised Visual Representation Learning

Momentum Contrast (MoCo) [3] is a self-supervised learning framework built on the idea of contrastive learning. MoCo maintains a dynamic dictionary of negative samples using a momentum-based encoder. This dictionary is implemented as a queue, where old representations are progressively replaced by new ones. The novelty that MoCo introduces in its momentum encoder, which helps maintain consistency in feature representations over time and reduces memory constraints by allowing the model to use a relatively smaller batch size while still accessing a large pool of negative samples.

The MoCo architecture consists of two encoders: a query encoder and a key encoder. The query encoder is updated via backpropagation, while the key encoder is updated using an exponential moving average (momentum update) to ensure stability as follows:

$$\theta_{\mathrm{k}} \leftarrow m\theta_{\mathrm{k}} + (1-m)\theta_{\mathrm{q}}. \tag{2}$$

Given an image, two different augmentations are applied, producing a query image and a key image. The query image is encoded using the query encoder, and the key image is encoded using the momentum encoder. The contrastive

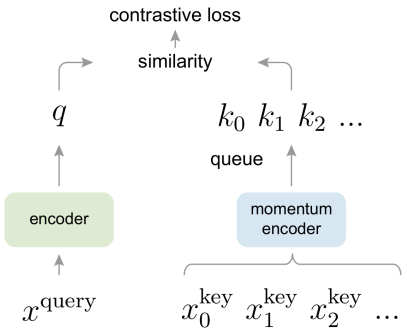


Figure 3. Momentum Contrast (MoCo) trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys $\{k_0, k_1, k_2, ...\}$ are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the current mini-batch enqueued and the oldest mini-batch dequeued, decoupling it from the mini-batch size. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder. This method enables a large and consistent dictionary for learning visual representations.

loss, typically InfoNCE defined as follows:

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+/\tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i/\tau)} \tag{3}$$

This loss is then applied to maximize similarity between the query and key embeddings while minimizing similarity with a queue of negative samples. This framework has been widely adopted in self-supervised learning due to its efficiency in handling large-scale datasets with limited computational resources.

3.2. Explainability Techniques

3.2.1. Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization

Gradient-weighted Class Activation Mapping (Grad-CAM)[5] is a widely adopted method that provides visual explanations for model predictions by leveraging gradient information to highlight the most important regions in an input image. Unlike earlier Class Activation Mapping (CAM) approaches, which required model modifications, Grad-CAM is architecture-agnostic and can be applied to various types of model architectures.

Grad-CAM generates class-specific saliency maps by computing the importance of feature maps in a CNN for a given target class. The method consists of the following steps:

1. **Forward Pass:** The input image is processed through the CNN, and feature maps A^k from a selected convolutional layer are extracted.

2. **Gradient Computation:** The gradients of the target class score y^c are computed with respect to each feature map:

$$\alpha_k^c = \frac{1}{Z} \sum_i \sum_j \frac{\partial y^c}{\partial A_{ij}^k}$$

where Z represents the spatial dimensions of the feature map.

3. **Weighted Feature Map Combination:** A weighted sum of the feature maps is computed using the importance weights α_k^c:

$$L_c = \text{ReLU}\left(\sum_k \alpha_k^c A^k\right)$$

The ReLU function ensures that only positive influences contribute to the visualization, preventing the inclusion of irrelevant or misleading features.

4. **Heatmap Generation and Overlay:** The resulting saliency map is re-sampled to the original image resolution and overlayed on the input image to highlight the key regions that contribute to the model decision.

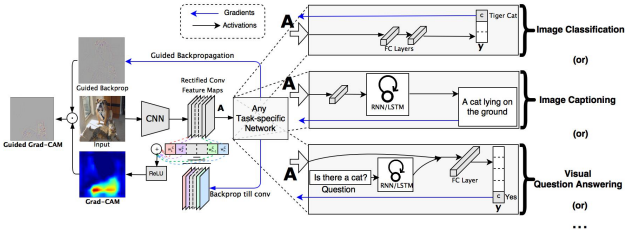


Figure 4. Grad-CAM visualization of a ResNet-50 model for the class "zebra." The heatmap highlights the regions in the input image that are most relevant for the model's prediction.

3.2.2. RELAX: Representation Learning Explainability

- Representation Learning Explainability (RELAX)[2] is a novel framework for explaining representations that also quantify their uncertainty.
- It measures the change in the representation of an image when compared with the masked version of the image.
- The central idea is that when informative parts are masked out, the representation should change significantly, i.e., the similarity between masked and unmasked representations should be low when informative parts are masked out and high when uninformative parts are masked out.

Let $\mathbf{X} \in \mathbb{R}^{H \times W}$ represents image of dimensions $H \times W$ and f be the feature extraction model that extracts representation $\mathbf{h} = f(\mathbf{X}) \in \mathbb{R}^D$.
To create masked images, we apply a stochastic mask $\mathbf{M} \in [0,1]^{H \times W}$, where M_{ij} is drawn from some distribution. The masked representation is given by $\bar{\mathbf{h}} = f(\mathbf{X} \odot \mathbf{M})$, where $\odot$ denotes element-wise multiplication.

The similarity between masked and unmasked representations is measured using cosine similarity,

$$s(\mathbf{h}, \bar{\mathbf{h}}) = \frac{\langle \mathbf{h}, \bar{\mathbf{h}} \rangle}{\|\mathbf{h}\| \|\bar{\mathbf{h}}\|},$$

, where $\|\cdot\|$ denotes the Euclidean norm of a vector. Importance R_{ij} of pixel (i, j) is defined as:

$$R_{ij} = \mathrm{E}_{\mathbf{M}}\left[s(\mathbf{h}, \bar{\mathbf{h}}) M_{ij}\right].$$

$$\bar{R}_{ij} = \frac{1}{N} \sum_{n=1}^{N} s(\mathbf{h}, \bar{\mathbf{h}}_n) M_{ij}(n).$$

The uncertainty U_{ij} of pixel (i, j) is defined as:

$$U_{ij} = \mathrm{Var}_{\mathbf{M}}[s(\mathbf{h}, \bar{\mathbf{h}}) M_{ij}].$$

$$\bar{U}_{ij} = \frac{1}{N} \sum_{n=1}^{N} (s(\mathbf{h}, \bar{\mathbf{h}}_n) - \bar{R}_{ij})^2 M_{ij}(n).$$

The figure 5, shows an example where RELAX is used to investigate the explanations and uncertainities for a selection of widely used feature extraction models. Red indicates high values, and blue indicates low values. In the figure, two birds are present, one prominently displayed in the foreground and another in the background. The plot shows all models emphasize the bird in the foreground with low uncertainty. However, the emphasis on the bird in the background varies with different degrees of uncertainty.

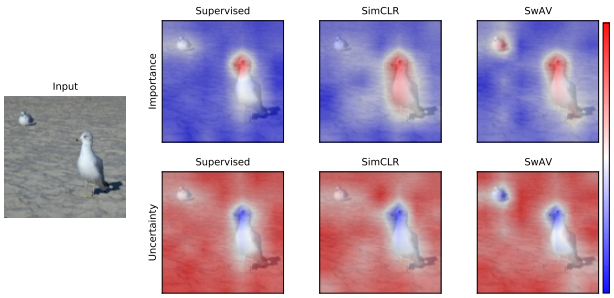


Figure 5. RELAX explations and uncertainty estimates for a VOC image.

3.2.3. Explaining Representation Learning with Perceptual Components [7]

The paper introduces a novel method to analyze representation spaces using three key perceptual components: color, shape, and texture. We employ selective masking of these components to observe changes in representations, resulting in distinct importance maps for each. Here, importance and uncertainty are found using the RELAX method.

Making strategies for Perceptual Components:

- **Color:** Original image is masked with the grayscale version of the image. Masking operation is denoted as:

  $$\mathbf{X_{MC}} = (\mathbf{X} \odot \mathbf{M}) + (\mathbf{X_{grayscale}} \odot (1 - \mathbf{M}))$$

  where $\mathbf{X_{grayscale}}$ is a grayscale transformed input image and $\mathbf{X_{MC}}$ is color masked image.

- **Shape:** Information about shape is extracted from the image by using edge detection. Canny edge detection is used to extract the edges of the image. The masked image is denoted as:

  $$\mathbf{X_{MS}} = \mathbf{X_{EdgeImage}} \odot \mathbf{M}$$

  where $\mathbf{X_{EdgeImage}}$ is the output of edge detection and $\mathbf{X_{MS}}$ is edge masked image.

- **Texture:** First input image is transformed to grayscale and then Gaussian blur is used to mask the image. The masked image is denoted as:

  $$\mathbf{X_{MT}} = (\mathbf{X_{grayscale}} \odot \mathbf{M}_t + (\mathbf{X_{blur}} \odot (1 - \mathbf{M}_t)))$$

  where $\mathbf{X}_{grayscale}$ is the grayscale transformation to the input image, $\mathbf{X_{blur}}$ is gaussian blurred grayscale image and $\mathbf{X}_{MT}$ is texture masked image. Here we use the mask with the addition of edge image and normal mask $\mathbf{M}_t = \mathbf{M} \vee \mathbf{X_{EdgeImage}}$ for masking where $\vee$ is logic element wise OR operator between two binary inputs. This masking ensures that the edges are not affected by blur operation.

4. Dataset

Since our objective is to obtain explainable representations using self-supervised learning, we are not restricted to a specific dataset. Instead, we aim to experiment with datasets that provide diverse visual concepts, enabling a better evaluation of both representation learning and explainability. For this purpose, currently we have considered the CIFAR-10 dataset for simplicity. As the size of images in CIFAR-10 is small, we also used the ImageNet dataset for our experiments.

5. Experiments

Experiments Setup

- Dataset: CIFAR-10 Test Set
- Models Compared:
- – SimCLR-trained on CIFAR-10 (domain-matched, fully trained)
  – SwAV-pretrained on ImageNet
  – MoCo-pretrained on ImageNet
  – Pretrained SimCLR ResNet-50 on ImageNet

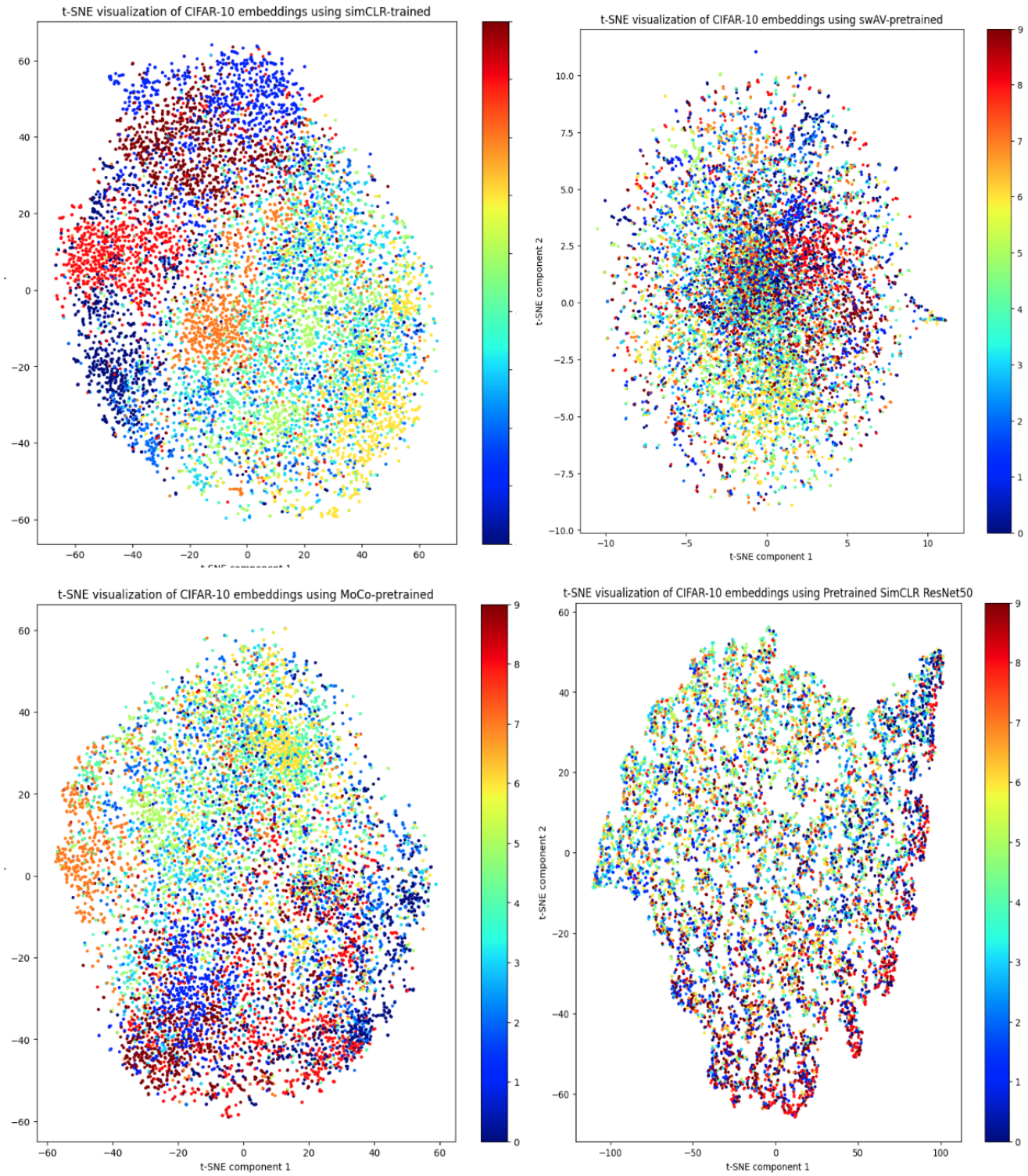


Figure 6. t-SNE visualizations of CIFAR-10 test set embeddings from various contrastive models. SimCLR was trained on CIFAR-10, while SwAV, MoCo, and SimCLR ResNet-50 were pretrained on ImageNet. Each point is colored by the ground-truth label. Clearer clustering in the SimCLR-CIFAR10 plot highlights the importance of domain-specific training for representation quality.

5.1. Visualizing Embedding Spaces of Contrastive Learning Models via t-SNE [6]

To understand and compare the semantic quality of embeddings learned by different contrastive learning models, we performed a t-SNE visualization experiment using the CIFAR-10 test set. The goal was to evaluate how well these self-supervised models encode class-discriminative features, even in the absence of labels during training.

The methodology for the experiment is as follows:

- Embedding Extraction: For each model, embeddings were extracted from the penultimate layer (or projection head output).
- Dimensionality Reduction: 2D t-SNE applied on the extracted embeddings.
- Color Encoding: Each point is colored by its ground-truth class label to assess class-wise separation.

The SimCLR model trained on CIFAR-10 produced well-separated clusters, indicating strong alignment between learned representations and ground-truth classes. This suggests that task-specific self-supervised training leads to highly disentangled embeddings. In contrast, the SwAV and MoCo models, both pretrained on ImageNet, showed overlapping and less distinct clusters, with SwAV appearing especially entangled near the center. MoCo demonstrated slightly better class separation, but the embeddings remained noisy and diffuse. The pretrained SimCLR ResNet-50 model showed a distinct spatial structure, but without clear clustering, highlighting that even with the same learning objective, domain mismatch and architecture depth can reduce embedding utility.

5.2. Evaluating Invariance of Saliency in Self-Supervised Models Using RELAX

To probe the invariance and stability of self-supervised contrastive models in identifying semantically important regions of an image, we conducted an experiment using RELAX, a recently proposed explainability method for SSL. RELAX generates pixel-level importance masks that highlight the regions most influential to the learned embeddings. Our goal was to test whether these models focus on consistent regions in both original and augmented versions of an image, a core assumption in contrastive learning. This experiment provides insight into whether self-supervised models maintain semantic focus under augmentations. A high degree of alignment would support the claim that these models learn transformation-invariant features, while discrepancies would point to limits in the robustness of their learned representations.

Methodology for the experiment is as follows:

- For an image I, compute the RELAX importance mask M_orig.
- Apply an augmentation A to obtain I_aug = A(I).
- compute RELAX mask M_aug on I_aug.
- Apply the inverse transformation A^{-1} to M_aug $\rightarrow$ M_aug_inv, aligning it back to the original image space.
- Compare M_orig and M_aug_inv both:
  - Visually (overlay masks).
  - Quantitatively using Intersection-over-Union (IoU): Threshold each mask using its mean importance value. Convert to binary mask (1 for pixels ¿ threshold, else 0).

$$\text{IoU} = \frac{|M_{\text{orig}} \cap M_{\text{aug\_inv}}|}{|M_{\text{orig}} \cup M_{\text{aug\_inv}}|}$$

Table 1. Intersection over Union (IoU) between original and augmented image importance masks for different self-supervised models across four augmentations.

Model	HFlip	VFlip	Rotate (15°)	Grayscale
SwAV	0.5721	0.4508	0.4860	0.5229
MoCo	0.6053	0.4519	0.6990	0.8944
SimCLR	0.6228	0.6415	0.6183	0.7654

Across the three self-supervised learning models—SwAV [fig 14], MoCo [fig 15], and SimCLR [fig 16]—the RELAX visualizations reveal distinct levels of spatial invariance to common image augmentations. SwAV consistently highlights the key regions (e.g., the subject's body and head) across all augmentations, demonstrating strong robustness and low uncertainty, particularly after reversing the transformations. In contrast, MoCo exhibits more scattered and augmentation-sensitive attention, with noticeable shifts in importance maps and higher uncertainty, especially under vertical flips and rotations, suggesting weaker localization stability. SimCLR, trained from scratch on CIFAR-10, falls in between: it maintains relatively coherent attention on primary features like the horse bodies, but displays minor instability under more disruptive transformations. Overall, these results reflect how model architecture, training data, and contrastive strategy influence the consistency of learned representations, with SwAV showing the highest invariance and MoCo the least in these visual assessments.

5.3. GradCAM Experimentation

5.3.1. Approaches to implement Grad-CAM

We have implemented Grad-CAM from scratch to visualize discriminative regions in CNNs, using intermediate feature maps and gradients from various layers of ResNet architectures. We have also experimented with and taken the results of layer choice on spatial resolution and interpretability.

Since Grad-CAM is traditionally used in supervised settings with class-specific gradients, we adapted it for self-supervised models (e.g. simCLR, MoCo v2, SwAV) by introducing proxy tasks such as similarity-based ranking.

We have computed gradients of cosine similarity between embeddings of two images with respect to feature maps of ResNet layers, allowing visualization of what features drive representation closeness.

We have selected a pair of images using the two methods below:

- The pair of images is an augmented image of a single image.
- The pair of images is different but taken from the same class.

With the above pair of images, we calculated the cosine similarity and used Grad-CAM to interpret which regions contributed most to high- or low-similarity scores. We have also applied Grad-CAM to a wide range of models, including ResNet50 and its variants.

An experiment has also been conducted to verify and visualize the features that the model has learned, as shown in Figure 7.

5.3.2. Interpretability of Learned Features:

- For high similarity pairs, Grad-CAM typically highlights overlapping regions (e.g., face patterns and body of the cat), suggesting the model has learned to focus on semantically meaningful features.
- For low similarity pairs, Grad-CAM may either highlight different regions (e.g., one image shows the head of a cat, another its tail) or non-object areas—indicating divergence in learned attention.

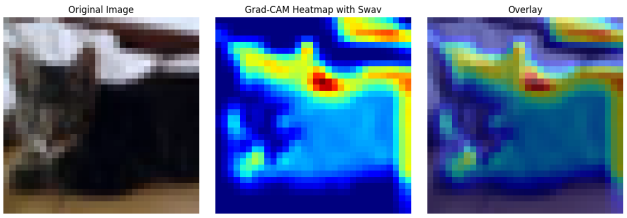
5.4. Perceptual Component Explainability

We began by replicating the original RELAX and modifications mentioned in [7] for the experimental setup, where the importance of three core perceptual components: color, shape, and texture, was evaluated using attribution-based explanations and frequency-filtered image variants. This served as a baseline for assessing how different visual properties contribute to the model's prediction.

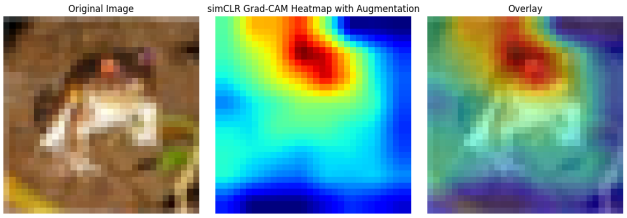
Following the methodology outlined in [7], we extended this analysis to incorporate additional perceptual factors: brightness, depth, and sharpness. Also, as each component uses a sort of filter to get the importance relating to that component, we also tried using a low-pass gaussian filter and a similar high-pass filter.

For each new perceptual component, we implemented the corresponding transformation pipelines:

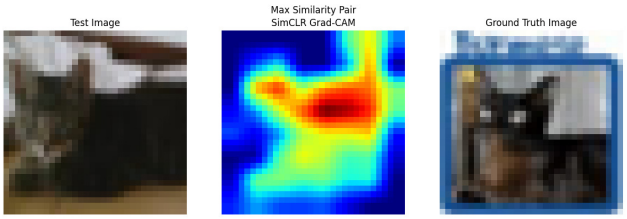
- **Brightness**: To evaluate the importance of brightness in the input image, we use the same technique used for color i.e. we replace the masked part by a less brightened version of that part. To get this change of brightness, we first convert the RGB image to LAB [lightness, A (red-green), and B (yellow-blue)], decrement the lightness channel, and then convert the modified image back to RGB.
- **Depth**: We used a MiDas [4] depth estimation model that computes relative inverse depth from a single image. It has been trained on 10 distinct datasets using multiobjective optimization to ensure high quality on a wide range of inputs. Using this depth map we calculate the importance in a similar manner to the shape and texture.
- **Sharpness**: Sharpness was enhanced by using a Laplacian sharpening kernel, and the importance of sharp re-



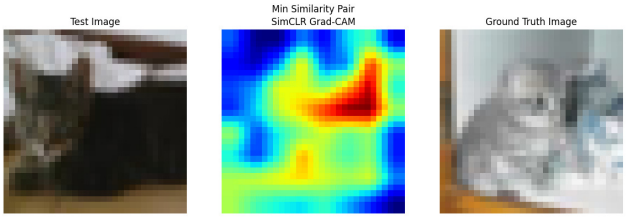
(a) Heatmap augmentation with SwAW



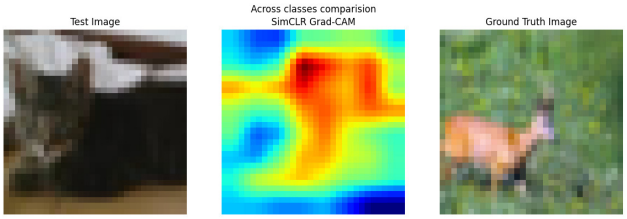
(b) Heatmap augmentation with simCLR



(c) Max similarity pair



(d) Min simlarity pair



(e) Across class comparison

Figure 7. Interpretability experiments using GradCAM

Figure 8. Input image alongside its low-pass and high-pass filtered versions to isolate frequency components. These versions are used to evaluate the model's reliance on low- and high-frequency visual information.

gions was determined by the difference between the sharpened image and the input. This approach again parallels the methods used for shape and texture.

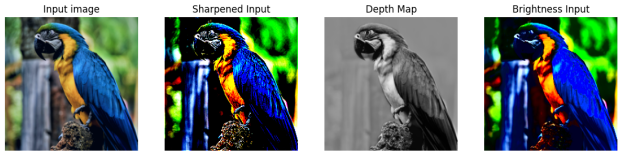


Figure 9. Input image, sharpened version, estimated depth map, and brightened image. These perceptual transformations are used to assess sensitivity of models to brightness, depth cues, and sharpness.

In case of perceptual concepts like color, we get the importance maps by replacing the masked area with the same area deprived of that concept. Thus, for perceptual concepts like sharpness and brightness, we replace the mask with a less brightened and sharpened part. In case of perceptual concepts like shapes and textures, we directly use the masks on edge maps,etc. So, similarly in the case of depth, we use the masks directly on the depth map obtained from the model. These images can be seen in Figure 9.

We can see the results for this experiment in figures 17 and 18. In general, considering models, the MoCo model (momentum contrast) performs visually better by mainly marking some part of the parrot as important in different importance maps. MoCo also remained the most invariant to these important maps, only showing a spread-out important region for the high frequency filtered map.

Considering the importance maps, the depth map visually looks the best as the background is masked in its doing. The sharpness maps cover most of the parrot body for all the models.

5.5. SimClr

We used SimClr as a feature extractor, used same architecture as in [2] with a ResNet-18 backbone, and a linear layer on top of the ResNet-18 features. We train the model on CIFAR-10 dataset.

For data augmentation, following techniques are used:

- `Color Jitter`: [0.8, 0.8, 0.8, 0.2] for brightness, contrast, saturation, and hue respectively, is applied to the input image with a probability of 0.8.
- `Random Resized Crop`: Randomly crop the input image and resize it to a given size of 32x32.
- `Random Horizontal Flip`: Randomly flip the input image horizontally with a probability of 0.5.
- `Gaussian Blur`: Apply Gaussian blur to the input image with a kernel size of 0.1*size of the input image, i.e., 3x3 kernel for 32x32 image.

Two views are created for each image using the above data augmentation techniques. They are passed through the feature extractor, and loss is minimized. Below given are the parameters used for training the model:

- Epochs: 1000
- Batch Size: 256
- Learning Rate: 3e-4
- Weight Decay: 1e-4
- Optimizer: Adam with $\beta_1 = 0.9$, $\beta_2 = 0.999$
- Scheduler: Cosine Annealing LR Scheduler

Figure 10 shows the training loss and accuracy of the model, in which we achieved an accuracy of 82.8125%.

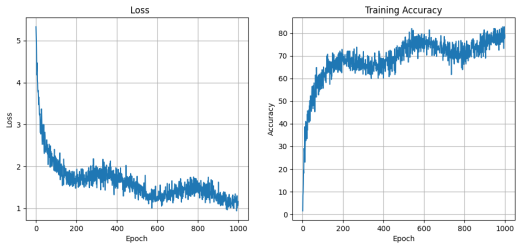


Figure 10. Loss and Accuracy plot for SimClr

5.6. Relax

Our SimCLR trained model is used as a feature extractor for the RELAX[6] model. Masks are generated for the input image and simlarity is found between the masked image and the original image. Importance and uncertainty are calculated for each pixel in the image using simlarity scores.

Figure 11, 12, and 13 show the RELAX explanation and uncertainty on CIFAR-10 dataset. In the figure 12, model is able to give importance to the aeroplane in the image and uncertainty is high in the background. In the figure 13, model is able to give importance to the bird in the image and uncertainty is high in the background. This shows that RELAX model is able to give importance to the object in the image and uncertainty in the background.

6. Observations

We have shown some few examples of the output from our SimCLR plus RELAX pipeline showing the most important

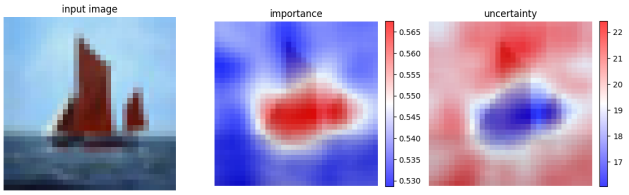


Figure 11. RELAX explanation and uncertanity on CIFAR-10 dataset

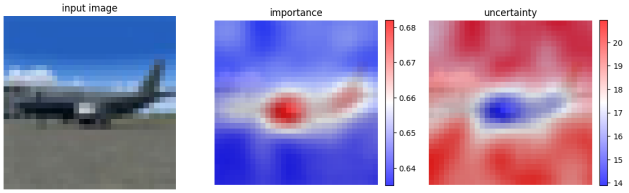


Figure 12. RELAX explanation and uncertanity on CIFAR-10 dataset

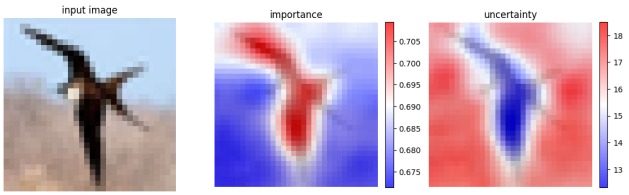


Figure 13. RELAX explanation and uncertanity on CIFAR-10 dataset

parts of the image and also the certainty with which this importance is proposed. The high importance areas are highlighted as masking those key areas of the images changes the representation of the image by a large margin. We can see the model noting the main object from the images being the ship, airplane and the bird with high certainty. For future works we'd like to compare the explainability of this pipeline with other explainable self supervised models.

7. Code Availability

The code for the experiments conducted in this project is available at github: https://github.com/darpan-gaur/cvProject.

References

[1] Mathilde Caron, Ishan Misra, Julien Mairal, Priya Goyal, Piotr Bojanowski, and Armand Joulin. Unsupervised learning of visual features by contrasting cluster assignments. *Advances in neural information processing systems*, 33:9912–9924, 2020. 1, 2

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020. 1, 3, 8

[3] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 1, 2

[4] René Ranftl, Katrin Lasinger, David Hafner, Konrad Schindler, and Vladlen Koltun. Towards robust monocular depth estimation: Mixing datasets for zero-shot cross-dataset transfer. *IEEE transactions on pattern analysis and machine intelligence*, 44(3):1623–1637, 2020. 7

[5] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2017. 1, 3

[6] Kristoffer K Wickstrøm, Daniel J Trosten, Sigurd Løkse, Ahcène Boubekki, Karl Øyvind Mikalsen, Michael C Kampffmeyer, and Robert Jenssen. Relax: Representation learning explainability. *International Journal of Computer Vision*, 131(6):1584–1610, 2023. 1, 8

[7] Yavuz Yarici, Kiran Kokilepersaud, Mohit Prabhushankar, and Ghassan AlRegib. Explaining representation learning with perceptual components. In *2024 IEEE International Conference on Image Processing (ICIP)*, pages 228–234. IEEE, 2024. 1, 4, 7

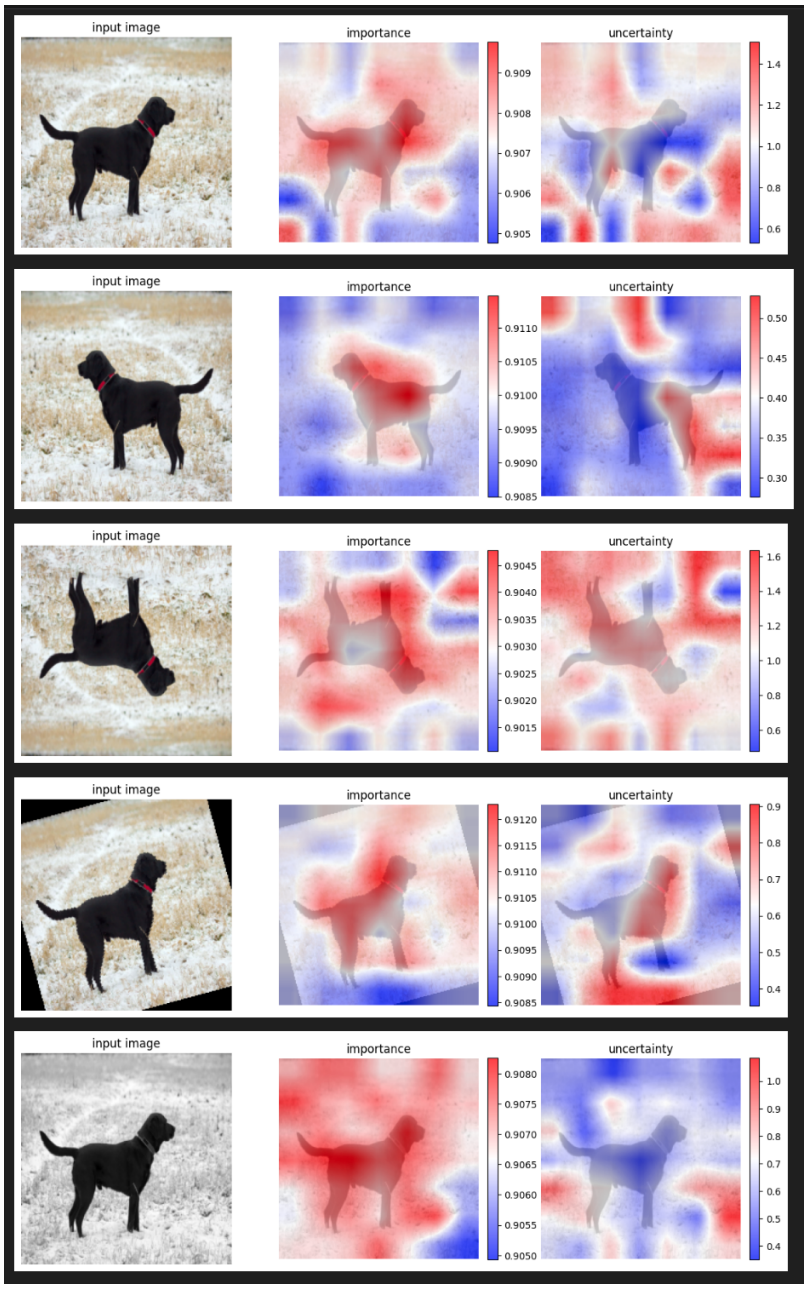


Figure 14. RELAX visualizations for the SwAV model across various augmentations: horizontal flip, vertical flip, 15° rotation, and grayscale. Importance maps remain focused on the object (dog), and uncertainty is relatively low, showcasing SwAV's strong spatial invariance and robust attention.

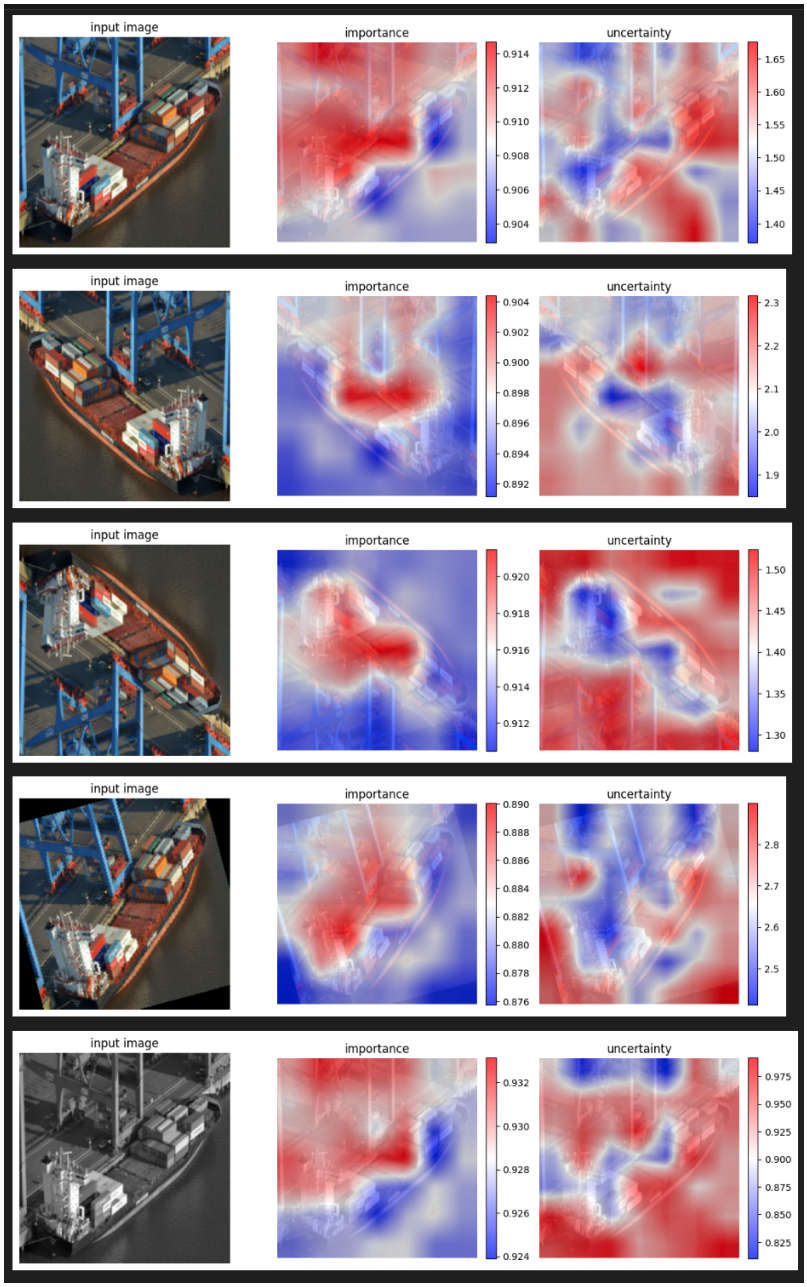


Figure 15. RELAX visualizations for the MoCo model under similar augmentations. The model exhibits higher uncertainty and shifting importance regions, particularly under strong geometric changes, suggesting comparatively lower invariance and stability.

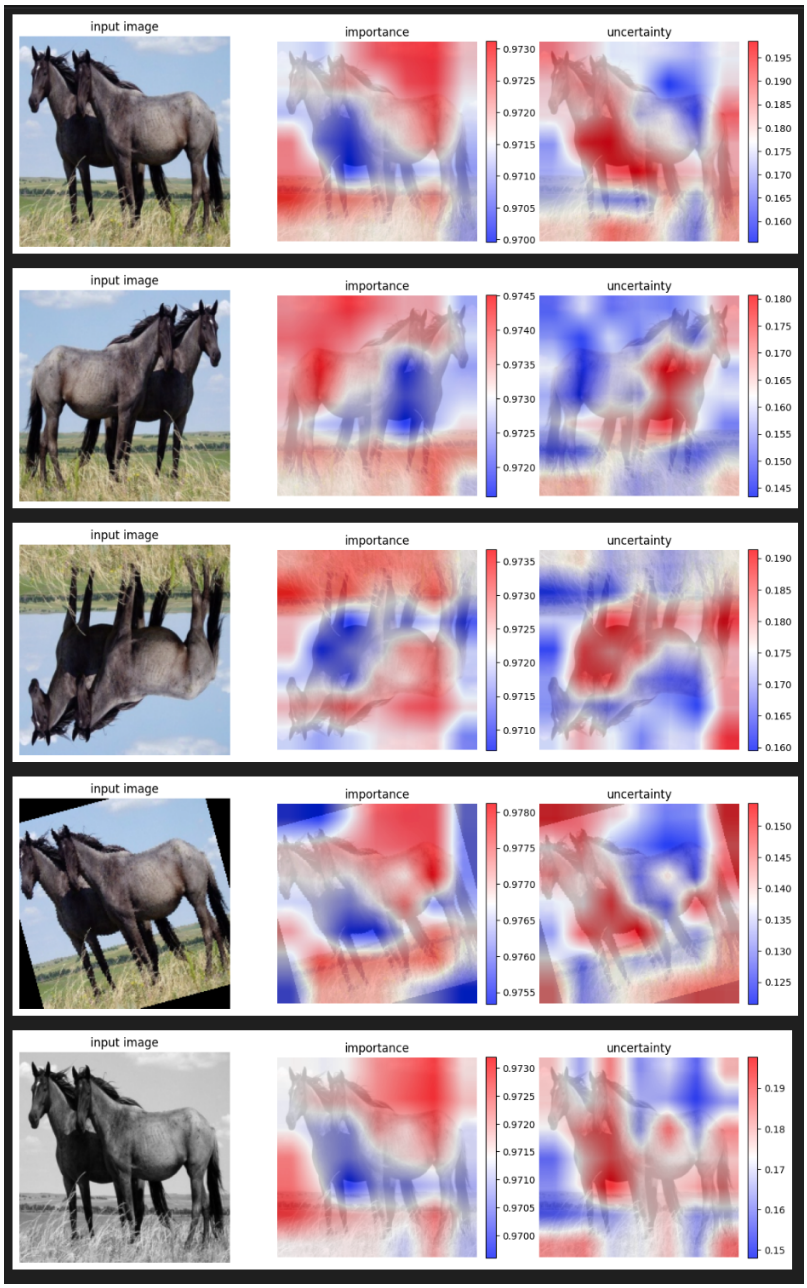


Figure 16. RELAX visualizations for SimCLR (trained from scratch on CIFAR-10). The importance regions are generally focused on the objects (horses), but there is moderate spatial drift and variable uncertainty across augmentations, reflecting a balance between sensitivity and consistency.

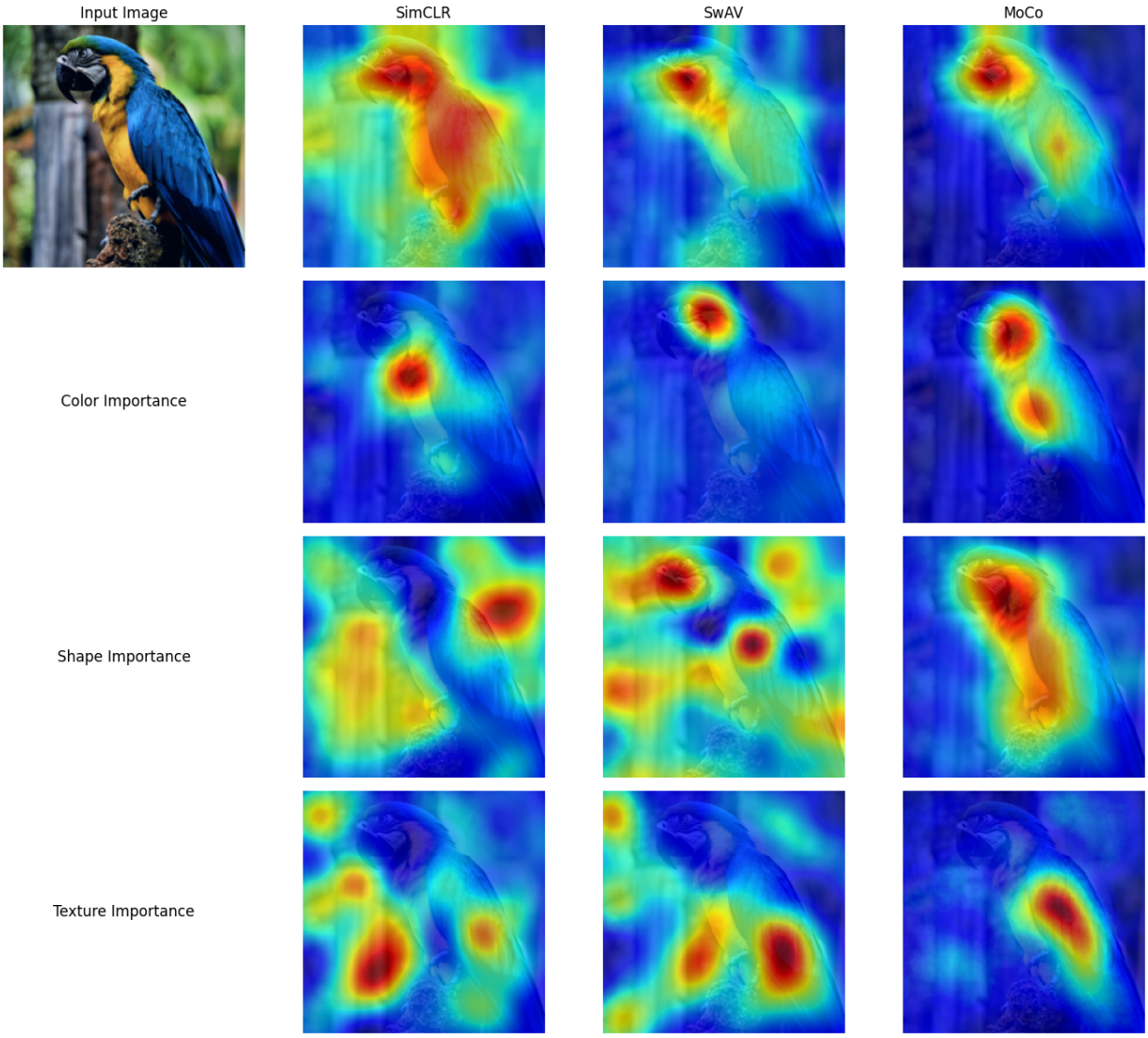


Figure 17. Perceptual importance maps showing the contribution of color, shape, and texture features across three models—SimCLR, SwAV, and MoCo—using the RELAX explanation method.

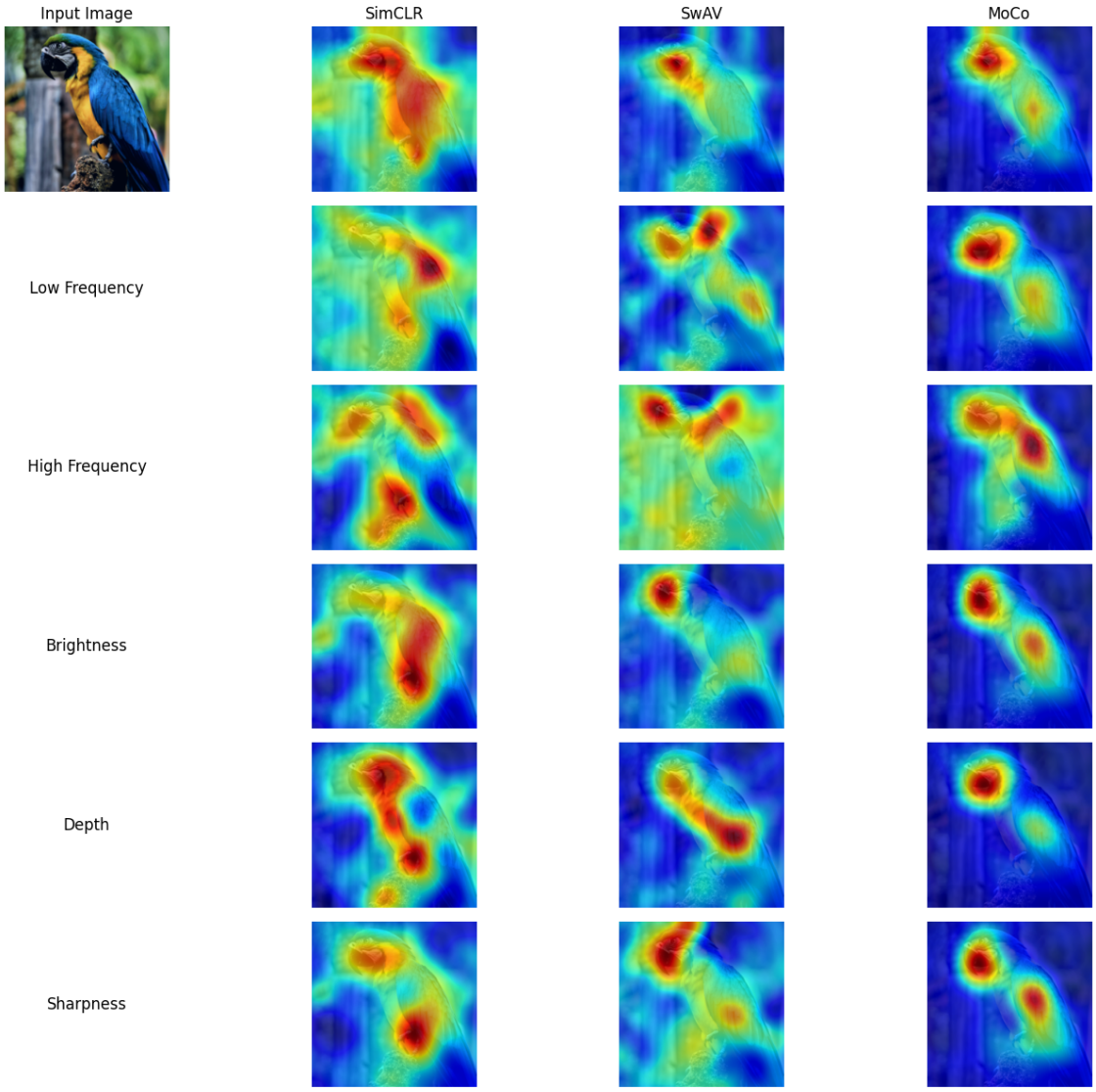


Figure 18. Extended perceptual component attribution maps including low-frequency, high-frequency, brightness, depth, and sharpness, comparing the same three models. These help identify additional cues the models may implicitly rely on.