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 and employ Grad-CAM and similar concept-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] or MoCo[3]).
- Apply Grad-CAM[5], RELAX[6] or other explainability techniques to visualize the important regions in SSL feature maps.
- Explore concept-based explainability techniques (e.g., TCAV, ProtoPNet) to align SSL representations with human-interpretable concepts.
- Evaluate the trade-off between interpretability and performance.

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 that leads 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).

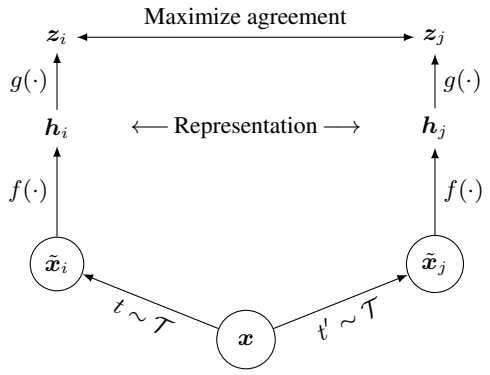

Figure 1. A simple framework for contrastive learning of visual representations. Two separate data augmentation operators are sampled from the same family of augmentations ($t \sim \mathcal{T}$ and $t' \sim \mathcal{T}$) and applied to each data example to obtain two correlated views. A base encoder 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 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.

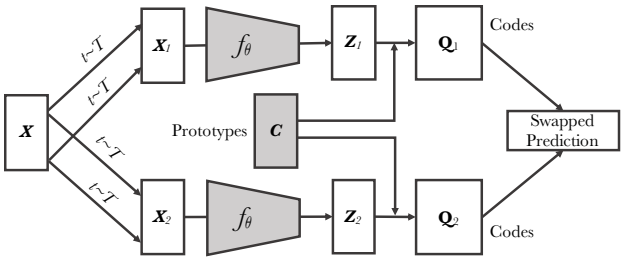

Figure 2. We first obtain "codes" by assigning features to prototype vectors. We then solve a "swapped" prediction problem wherein the codes obtained from one data augmented view are predicted using the other view. Prototype vectors are learned along with the ConvNet parameters by backpropagation.

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

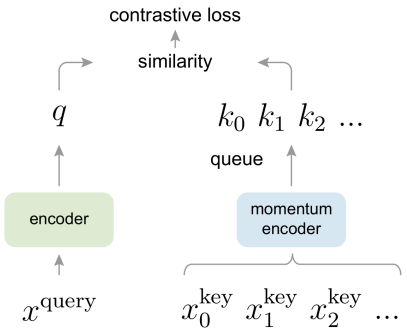

Figure 3. Momentum Contrast (MoCo) trains a visual representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss. The dictionary keys $\{k_0, k_1, k_2, ...\}$ are defined on-the-fly by a set of data samples. The dictionary is built as a queue, with the 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)[4] 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.

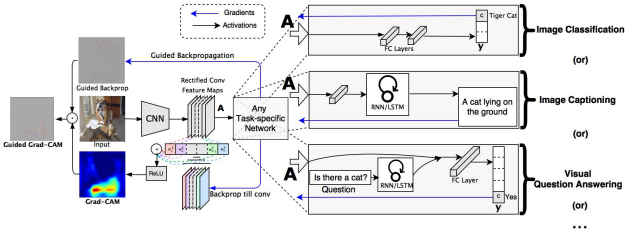

Figure 4. Grad-CAM visualization of a ResNet-50 model for the class "zebra." The heatmap highlights the regions in the input image that are most relevant for the model's prediction.

3.2.2. RELAX: Representation Learning Explainability

- Representation Learning Explainability (RELAX)[7] 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:

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

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

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.

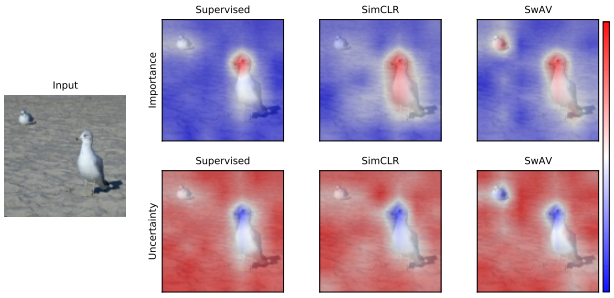

Figure 5. RELAX explations and uncertainty estimates for a VOC image.

3.2.3. Explaining Representation Learning with Perceptual Components [8]

Introduce 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 represen- tations, 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

5.1. TSNE plot Visualization

5.2. Effect of Augmentation on Representation and Explainability

5.3. GradCAM plots Visualization

5.4. Perpectual Component Explainability

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

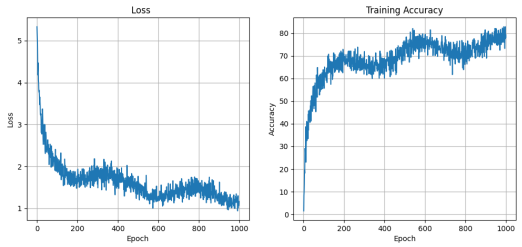

Figure 6. Loss and Accuracy plot for SimClr

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 7, 8, and 9 show the RELAX explanation and uncertainty on CIFAR-10 dataset. In the figure 8, model is able to give importance to the aeroplane in the image and uncertainty is high in the background. In the figure 9, 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.

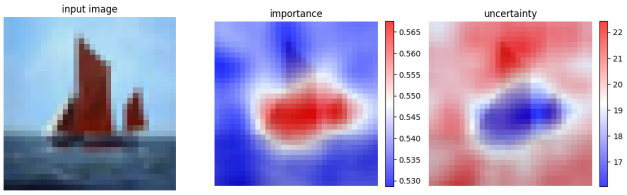

Figure 7. RELAX explanation and uncertainty on CIFAR-10 dataset

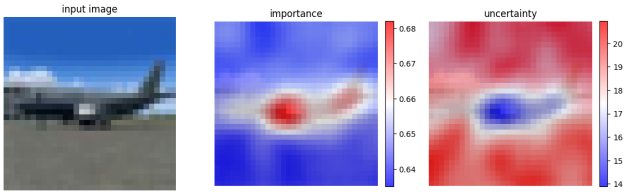

Figure 8. RELAX explanation and uncertainty on CIFAR-10 dataset

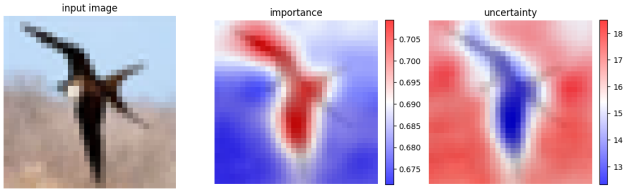

Figure 9. RELAX explanation and uncertainty on CIFAR-10 dataset

6. Observations

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

vances in neural information processing systems, 33:9912–9924, 2020. 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, 4

[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] 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. 3

[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*, pages 618–626, 2017. 1

[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, 5

[7] Kristoffer K. Wickstrøm, Daniel J. Trosten, Sigurd Løkse, Ahcène Boubekki, Karl Øyvind Mikalsen, Michael C. Kampffmeyer, and Robert Jenssen. Relax: Representation learning explainability, 2022. 3

[8] 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. 4