Linear Explanations for Individual Neurons

Tuomas Oikarinen [1] **Tsui-Wei Weng** [2]

Abstract

In recent years many methods have been developed to understand the internal workings of neural networks, often by describing the function of individual neurons in the model. However, these methods typically only focus on explaining the very highest activations of a neuron. In this paper we show this is not sufficient, and that the highest activation range is only responsible for a very small percentage of the neuron's causal effect. In addition, inputs causing lower activations are often very different and can't be reliably predicted by only looking at high activations. We propose that neurons should instead be understood as a linear combination of concepts, and develop an efficient method for producing these linear explanations. In addition, we show how to automatically evaluate description quality using *simulation*, i.e. predicting neuron activations on unseen inputs in vision setting.

1. Introduction

Current machine learning models are extremely capable at many tasks, yet they are notoriously hard to understand, and most often seen as black boxes. Recently, the field of *mechanistic interpretability* (Olah et al., 2020; Elhage et al., 2021) has emerged to address this issue by providing mechanistic understanding of the inner workings of neural networks, with the ultimate goal of reverse engineering the algorithms that neural networks use to solve problems.

Individual neurons (or channels of a CNN) are the most simple unit of a neural network, and understanding them is a fundamental building block of mechanistic interpretability. Many methods have been developed to understand individual neurons, based on both manual inspection (Erhan et al., 2009; Zhou et al., 2015; Olah et al., 2020) and automated description (Bau et al., 2017; Hernandez et al., 2022; Oikarinen & Weng, 2023). However, most of these methods only focus on understanding and explaining the very highest activations of a neuron. In Section 2 we show that this is not enough, and a neuron's impact on network outputs is distributed quite evenly across all inputs. Therefore, to fully understand a neuron, we need to understand everything it is doing, not just its highest activations.

In Section 3 we propose a solution to these issues, which we call **Linear Explanations (LE)**. In our method, each neuron is described as linear combination of concepts, such as "$w_1\times$ dog + $w_2\times$ pet". This explanation allows us to directly simulate neuron activations as $s(x_i) = w_1\mathbb{P}(\text{dog}|x_i) + w_2\mathbb{P}(\text{pet}|x_i)$, where $\mathbb{P}(\text{concept}|x_i)$ can be estimated by either a human or a model. Our automated method allows for learning accurate linear explanations efficiently, as demonstrated in Section 5, where we evaluate and compare our method and existing automated explanation methods. We believe linear explanations are a natural way to represent neurons, and they can elegantly model the scalar nature of a neuron's activation. In addition, linear explanations can be found efficiently and do not binarize neuron activations (which loses information), unlike alternative methods to capture complex neuron behavior such as Compositional Explanations (Mu & Andreas, 2020).

In Section 4 we propose a new, more rigorous way to automatically evaluate the quality of neuron descriptions via *simulation*, i.e. predicting the neuron's activation on a new input given the explanation. The simulation evaluation was recently proposed for explaining neurons of large language models by (Bills et al., 2023), and has since been a popular evaluation method in language settings (Cunningham et al., 2023; Bricken et al., 2023). We are the first to present an effective and natural way to run simulation on vision models by utilizing SigLIP (Zhai et al., 2023) models as the simulator. We find that existing neuron explanations (Bau et al., 2017; Hernandez et al., 2022; Oikarinen & Weng, 2023) perform poorly under simulation, while our method provides more than $3\times$ higher ablation scores when simulated. Finally, in Figure 3 we show how our method can uncover multiple roles played by the same neuron that would be missed when only looking at highest activations. Our code and results are available at https://github.com/Trustworthy-ML-Lab/Linear-Explanations.

[1]CSE, UC San Diego, CA, USA [2]HDSI, UC San Diego, CA, USA. Correspondence to: Tuomas Oikarinen <toikarinen@ucsd.edu>, Tsui-Wei Weng <lweng@ucsd.edu>.

Proceedings of the 41^{st} International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

2. Motivation: How Important are Different Parts of a Neuron's Activation Pattern?

In this section, we aim to answer the following question:

Is most of a neuron's impact on the network caused by the very highest activating inputs, or are all inputs important?

We do this by ablating the individual neurons of a network, i.e. replacing their activation by 0, and measuring the change in network outputs. We refer to this as the neuron's causal effect.

2.1. Definitions

To describe our results, we first need to define a few metrics: The idea is to measure the causal effect of these neurons in terms of metrics most relevant to the end use case, which in a classification setting are accuracy and cross-entropy loss.

Let $f(\cdot)$ be a neural network of interest. $f(x)$ is the network's output on an input x, i.e. class probabilities. Let $D = \{(x_i, y_i)\}$ be a dataset with images x_i and corresponding ground-truth class labels y_i. We define $I_k(x_i, y_i)$ as the impact of neuron k on the input x_i:

$$I_k(x_i, y_i) := \frac{\Delta\text{Acc}_k(x_i, y_i) - \Delta L_k(x_i, y_i)}{2} \tag{1}$$

where $\Delta\text{Acc}_k(x_i, y_i)$ is the change in accuracy and $\Delta L_k(x_i, y_i)$ is the change in loss as formally defined below:

$$\Delta\text{Acc}_k(x_i, y_i) = \frac{[h(f(x_i)) = y_i] - [h(f_{\sim k}(x_i)) = y_i]}{\sum_{(x_j,y_j)\in D}[h(f(x_j)) = y_j]} \tag{2}$$

$$\Delta L_k(x_i, y_i) = \frac{L(f(x_i), y_i) - L(f_{\sim k}(x_i), y_i)}{\sum_{(x_j,y_j)\in D} L(f(x_j), y_j)} \tag{3}$$

Here $[\cdot]$ is the indicator function, taking value 1 if the expression is True and 0 otherwise, $h(\cdot) := \text{argmax}(\cdot)$, $f_{\sim k}(\cdot)$ is the output of the model without neuron k, i.e. the neuron's activation is replaced by 0 and L is the loss function, such as cross-entropy loss. We use temperature calibration (Guo et al., 2017) before calculating the losses.

The impact $I_k(x_i, y_i)$ is then the average of the neuron's effect on accuracy and loss of the model on an input, as a fraction of the model's total loss and accuracy, with signs chosen such that a positive impact means including the neuron k improves the network's predictions.

Next, we measure how much of a neuron's total impact is caused by the inputs that activate it the highest. To achieve this, we define *Top Impact*, denoted as $TI_k(\beta)$, with the input argument $\beta \in [0, 1]$ representing the fraction of highest activating inputs included:

$$TI_k(\beta) = \frac{\sum_{i=1}^{\beta|D|} |I_k(x_{(i)}, y_{(i)})|}{\sum_{(x_j,y_j)\in D} |I_k(x_j, y_j)|} \tag{4}$$

where $x_{(i)}, y_{(i)}$ are ordered in descending order of neuron k's activation. I.e. $g(A_k(x_{(i)})) \geq g(A_k(x_{(i+1)}))\,\forall i$, where $A_k(x_i)$ is the activation of a neuron or a channel in CNNs on input x_i, and g is a summary function such as mean or max that takes the 2D activation map of a CNN channel into a single scalar (or identity for scalar neurons) as defined in (Oikarinen & Weng, 2023). A neuron where all inputs are equally important should have $TI_k(\beta) = \beta$.

2.2. Results

To measure how important highly activating inputs are to network predictions in practice, we ablated out all the neurons (channels) of ResNet-50 (He et al., 2016) one at a time and measured the change in performance on ImageNet validation data. Results in terms of *Top Impact* are shown in Table 1. We used mean as summary function g. We can see that if we wish to understand most of a neuron's impact (i.e. high TI), we need to look at a large fraction of the neurons inputs (high β), not just the most highly activating inputs.

This is in contrast to many popular methods of single neuron explanation, which often focus exclusively on the most highly activating inputs, i.e. very small β. For example MILAN (Hernandez et al., 2022) only looks at 15 most highly activating inputs, which is equivalent to $\beta = 0.0003$. In Table 1, we can see that these inputs only explain 0.258% of the neuron's impact on average. Similarly, CLIP-Dissect (Oikarinen & Weng, 2023) with soft-wpmi activation function only looks at 100 ($\beta = 0.002$) most highly activating inputs, which make up 1.522% of the neuron's impact. While Network Dissection (Bau et al., 2017) looks at all inputs, it only aims to explain top 0.5% of location specific neuron activations which causes similar issues. Based on the findings in Tab 1, we believe most inputs of a neuron are important, and only focusing on highest activating inputs (small β) is not sufficient to faithfully explain individual neurons, or to evaluate how good such explanations are. To resolve this issue, we propose a new explanation method (Section 3) and a new evaluation metric (Section 4) that focus on explaining the *entire* range of a neuron's activations.

β	Conv 1	Layer 1	Layer 2	Layer 3	Layer 4	All
0.0003	0.054%	0.084%	0.110%	0.268%	0.319%	0.258%
0.002	0.319%	0.454%	0.574%	1.369%	2.007%	1.522%
0.02	2.743%	3.526%	4.160%	8.654%	13.59%	10.22%
0.1	12.42%	14.80%	16.53%	28.28%	39.38%	31.42%
0.5	55.23%	59.49%	62.08%	76.85%	84.55%	77.46%

Table 1. Average *Top Impact* for neurons in different layers of ResNet-50 (ImageNet). We can see that while highly activating inputs (small β) are more impactful than the average input, especially on later layers, they still explain only a small portion of the neuron's total effect. E.g. $\beta = 0.002$ only accounts for around 1.52% of Total Impact.

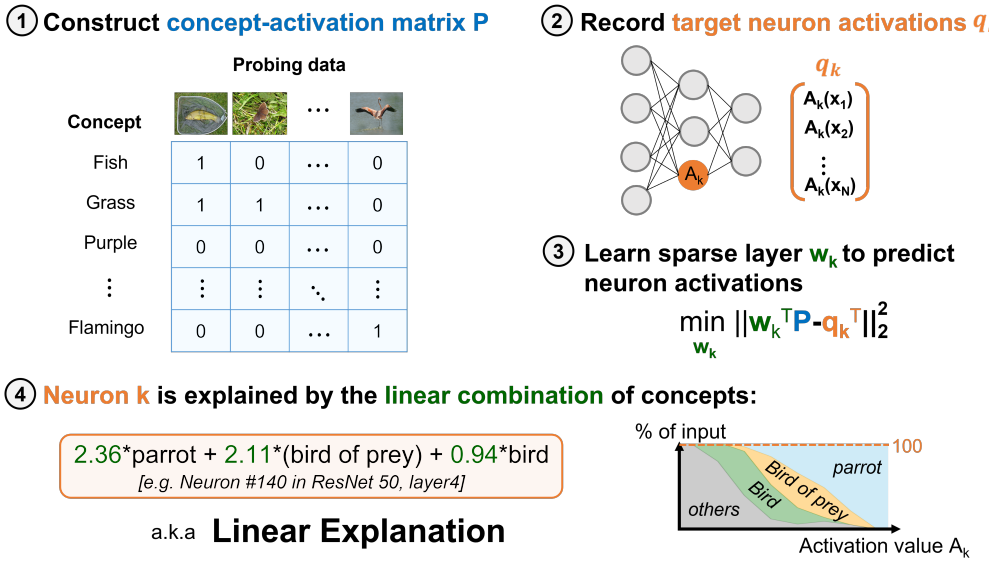

Figure 1. Overview of our proposed method: Linear Explanations.

3. Method

In this section we present our method **Linear Explanations** or **LE**. Our method consists of two main parts: (i) Constructing a concept activation matrix P, and (ii) Learning a linear combination to explain the neuron. An overview of our method is shown in Figure 1.

3.1. Constructing a concept activation matrix

An essential part of our explanation is creating the concept activation matrix P. Each entry P_{ij} represents how much of concept c_i is present in input x_j in a way that is aligned with human perception. That is $P_{ij} \approx \mathbb{P}(c_i|x_j)$, for each input x_j of the probing dataset $\mathcal{D}_{\text{probe}}(|\mathcal{D}_{\text{probe}}| = N)$ and each concept c_i in the concept set $\mathcal{S}(|\mathcal{S}| = M)$.

In this paper we experiment with two different methods of determining P described below: **Label** and **SigLIP**.

Label: Use labeled data. If we have access to labeled data for all concepts in the concept set for all inputs in the probing dataset, we can directly use these labels as our concept activation matrix P. This is the ideal case and works in situations where such data is available, such as the Broden (Bau et al., 2017) dataset or the class labels on a validation dataset. We denote this approach as **LE(Label)**. However, we often do not have access to labels for all concepts and/or images we want to utilize, and collecting additional labeled data can be expensive. This leads us to our second method **SigLIP**.

SigLIP: Pseudo-labels from Multimodal Models. In cases where sufficient labeled data is not available, it is often beneficial to create artificial labels using pretrained foundation models such as CLIP (Radford et al., 2021). In our experiments we used SigLIP (Zhai et al., 2023)(ViT-L-16-384) as our *explainer model*. SigLIP(ViT-L-16-384) is a more recent model similar to CLIP, and was chosen because of its improved performance, as well as use of sigmoid function during pretraining. Training with sigmoid activation is important for creating a reliable concept activation matrix P, as it is essentially a multi-label classification task, i.e. most images contain multiple concepts. Because of this, softmax based models will likely perform poorly on constructing a concept activation matrix P.

To ensure the pretrained SigLIP model is aligned with human concept predictions, we add additional calibration parameters a and b and optimize these using publicly available data. Let E_I be the image encoder and E_T be the text encoder of the SigLIP model. We generate concept activation matrix P as follows: $P_{ij} = \sigma(a \cdot E_T(c_i)E_I(x_j) + a \cdot b)$. Since our goal is to have $P_{ij} \approx \mathbb{P}(c_i|x_j)$, i.e. to be similar to human predictions, we learn a and b by minimizing binary cross-entropy loss L_{BCE} on concepts we have labels for. In particular we use ImageNet validation data, with added superclass labels according to WordNet hierarchy (Miller, 1995), which turns this into a multi-label classification task. a and b are then determined as:

$$\min_{a,b} \sum_{x_i \in D} \sum_{c_j \in C} L_{BCE}(\sigma(a \cdot E_T(c_j)E_I(x_i) + a \cdot b), y_{i,j}) \quad (5)$$

where C is the set of [super]class names, c_j is the name of j-th [super]class, $y_{i,j}$ is a binary label indicating whether input x_i belongs to [super]class c_j and L_{BCE} is binary cross-entropy loss. The hyperparameters a and b are explainer model specific, and independent of $\mathcal{D}_{\text{probe}}$ or $\mathcal{S}$.

3.2. Learning Linear Explanations

Once we have our concept activation matrix $P \in \mathbb{R}^{M \times N}$, our next task is to learn a sparse linear model that can predict the activation of our target neuron based on the presence of a few concepts in our input. We do this in two steps described below. Let $q_k \in \mathbb{R}^N$ be the activation vector of neuron k, defined as $q_k = \begin{bmatrix} g(A_k(x_1)) \\ g(A_k(x_2)) \\ \vdots \\ g(A_k(x_N)) \end{bmatrix}$, where $A_k(x_i)$ is the activation of neuron k on input x_i, and $g(\cdot)$ is a summary function(mean) which is used in case the neuron's activation is not a scalar, for example when neuron k is a channel of a CNN.

Throughout the paper we use a 70-10-20 split to divide our $\mathcal{D}_{\text{probe}}$ into train, validation and test set splits to avoid overfitting our explanations. We denote the train subset of q_k and P as q_k^{train} and P^{train} respectively.

3.2.1. Learn a relatively sparse w_k

First, we use the GLM-Saga package (Wong et al., 2021) to learn a sparse linear weight $w_k \in \mathbb{R}^M$ to minimize the following objective:

$$\mathcal{L}_{\text{MSE}} = ||w_k^\top P^{\text{train}} - (q_k^{\text{train}})^\top||_2^2 + \lambda R_\eta(w_k) \qquad (6)$$

where $R_\eta(w_k) = (1-\eta)\frac{1}{2}||w_k||_2^2 + \eta||w_k||_1$ and η is a hyperparameter, set to 0.99 in our experiments. It is worth noting that we optimize the predictions to be accurate on the entire $\mathcal{D}_{\text{probe}}^{\text{train}}$ (i.e. $\beta = 1$), so our explanations describe the entire activation range, not just the most highly activating inputs as discussed in Sec 2. Our goal is to learn w_k that can accurately predict neuron activations based on just the concept information embedded in the concept activation matrix P, while also being sparse for interpretability. The sparsity constraint is reflected in the term $||w_k||_1$ of the regularization $R_\eta(w_k)$. This is a surrogate term for the exact sparsity goal of minimizing the ℓ_0 norm (i.e. $||w_k||_0$), as the ℓ_1 norm is convex and much easier to optimize than the non-convex ℓ_0 norm. However we found it hard/unstable to find extremely sparse (around 5 concepts per neuron) w_k using only this method.

In Section 3.2.2 we discuss how to overcome this using greedy search guided by the found values w_k. For discussion on results without using greedy search, see Appendix B.4.

3.2.2. Greedy search

Our basic idea for greedy search is to use the weights w_k found in previous step as a heuristic to find a very sparse explanation for the neuron of interest. This greedy-search process is described in detail in Algorithm 1 and Appendix A.3. For each neuron, we test $r = 10$ available concepts with the highest weight, and choose the one that can train the best model together with the already selected concepts. We continue this process until we reach the maximum number of concepts $v = 10$, or until adding another concept does not improve performance enough. This is modeled by the tolerance parameter ϵ. This lets our method dynamically decide the number of concepts for each neuron, adding more complexity to the explanation only when it is needed. The tolerance parameter can be adjusted to make a tradeoff between simpler and shorter v.s. more complete explanations. This process requires training many models per each neuron, but they are extremely small linear models (10 or less parameters) which can be trained to optimal in a fraction of a second. Each of these models is trained to minimize MSE loss of predicting neuron activations as a linear function of only the selected concepts (with no regularizer), i.e. the first term in Eq (6). More concretely, given set of concept indices discovered from Greedy search $\mathcal{I} = \{i_1, ..., i_u\}$, $u \leq v$, our final explanation E is then:

$$E = \{(w_{k,1}^*, c_{i_1}), \cdots, (w_{k,u}^*, c_{i_u})\} \qquad (7)$$

where $w_k^* = \text{argmin}_{w \in \mathbb{R}^u} ||w^\top P_{\mathcal{I}}^{\text{train}} - (q_k^{\text{train}})^\top||_2^2$.

4. Improving Evaluation of Explanations via Simulation

An important part of creating explanations for individual neurons is being able to evaluate how faithfully these descriptions actually correspond to target neuron behavior. This has traditionally been done with methods such as evaluating whether the description matches the most highly activating images (Bau et al., 2017; Hernandez et al., 2022; Oikarinen & Weng, 2023; Kalibhat et al., 2023), but this only evaluates a very small portion of the neuron's activations, which is not sufficient for understanding the neuron as we have shown in Section 2. In addition, (Zimmermann et al., 2023) showed that while the very highest activations of a neuron may often be understandable, understanding a larger part of the activation range quickly becomes difficult for humans.

Inspired by recent work in language models (Bills et al., 2023), we instead propose to evaluate our explanations using *simulation*. See Figure 2 for an overview of our simulation pipeline. The basic idea of *simulation* is as follows:

1. Generate a human understandable explanation E for the neurons using an *Explainer*[1].
2. Use a *Simulator* to predict neuron activations $s(x, E)$ on new inputs x, based on only the explanation E and the input.

[1]Explainer could be a neural network, a human or an algorithm using human-annotated labels

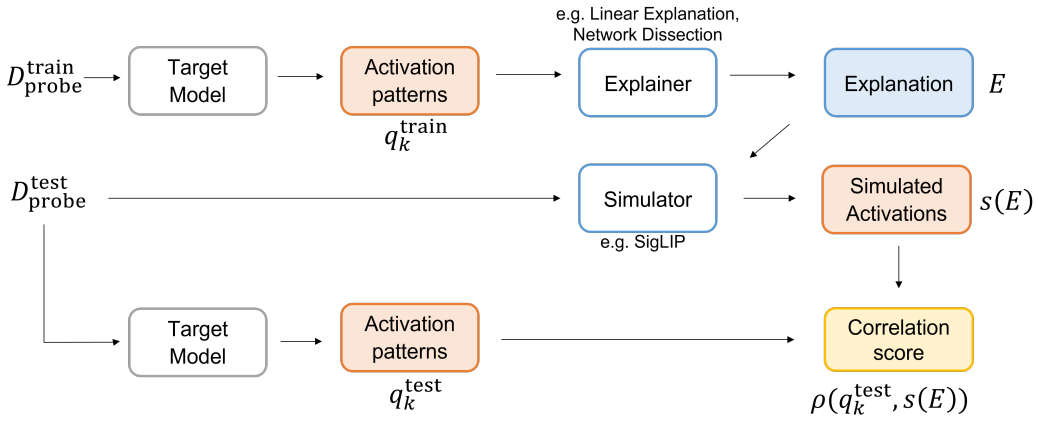

Figure 2. An overview of the simulation pipeline with correlation scoring.

3. Score how well the simulated activations fit the actual neuron activations, using either correlation or ablation scoring.

The intuition behind simulation is that a good explanation should allow the simulator model (or a human) to accurately predict the neuron activations. (Bills et al., 2023) used GPT-4 as both the explainer and the simulator, showing the 5 most highly activating text excerpts from a subset of its training data to generate an explanation, and then simulate its activations on 5-10 new excerpts to evaluate how good these explanations are.

In this section, we propose how to naturally extend the idea of simulation from language to the vision domain, and discuss some conceptual and practical improvements we can do in this modality. We argue a good simulator should have the following properties:

1. The simulator needs to be able to take any textual concept, and predict how highly it activates on any given image (or subpart of the image).

2. The simulator and the explainer should be **different** models. Using the same model may overestimate the quality of the explanations. For example, the explainer could use an uninterpretable code, or a misunderstood concept to explain a neuron and still receive a high score if the simulator shares this misunderstanding.

3. Third, a simulator should be *human aligned* i.e. predict activations similar to how a human would.

Following the above, we propose using CLIP (Radford et al., 2021) or similar models to perform this image level simulation. In particular, we choose the SigLIP-SO400M-14-384 (Zhai et al., 2023) as the simulator, because it is the most powerful sigmoid trained model available. Note this is a different SigLIP model than the one we use to generate explanations of **LE(SigLIP)**.

To ensure that our simulator is human-aligned, we base our simulation on predicting $\mathbb{P}(c_i|x_j)$, i.e. how likely is the concept represented by c_i is to be present in image x_j. This is important as it is aligned with how humans usually think about concepts, instead of directly predicting real values as done by (Bills et al., 2023). We estimate $\mathbb{P}(c_i|x_j)$ similar to how we constructed matrix P in Section 3, i.e.

$$\mathbb{P}_{sim}(c_i|x_j) = \sigma(a_{sim}E_T^{sim}(c_i)E_I^{sim}(x_j) + a_{sim}b_{sim}) \tag{8}$$

Like before, we optimize the hyperparameters a_{sim} and b_{sim} on ImageNet validation data with superclasses as defined in Eq. (5). Different from explainer (Section 3), the simulator does not use a fixed concept set, but instead evaluates all the concepts present in explanation E.

For explainers that produce a single text explanation c_e (Bau et al., 2017; Hernandez et al., 2022; Oikarinen & Weng, 2023), their explanation can be written as a linear explanation of length 1, i.e. $E = \{(1, c_e)\}$. Once we have an explanation E, the initial simulated activation $s(x_j, E)$ is calculated in the following way:

$$s(x_j, E) = \sum_{(w_i,c_i)\in E} w_i \mathbb{P}_{sim}(c_i|x_j) \tag{9}$$

With $s(x_j, E)$, we can evaluate the quality of explanation E using two scoring methods:

A. Correlation Scoring $\rho(k, E)$: The explanations E are scored based on the correlation coefficient between the simulated activations and the neuron's real activations patterns.

$$\rho(k, E) = \sum_{x\in\mathcal{D}_{\text{probe}}^{\text{test}}} \frac{\hat{s}(x, E) \cdot \hat{g}(A_k(x))}{|\mathcal{D}_{\text{probe}}^{\text{test}}|} \tag{10}$$

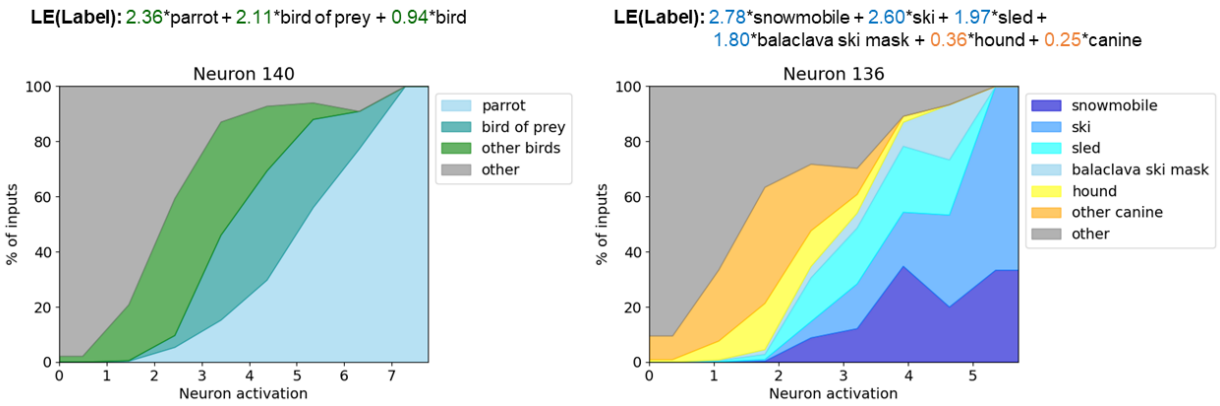

Figure 3. An area chart of the activations of two neurons in layer4 of ResNet-50. We can see Neuron 140 is mostly monosemantic, but represents different types of birds at different activation ranges. In contrast, neuron 136 has two distinct roles, snow and skiing related concepts at high activations and dog-like animals at lower activations.

where $\hat{s}$ and $\hat{g}(A_k)$ are s and $g(A_k)$ normalized to have mean 0 and standard deviation of 1 on the test distribution.

B. Ablation scoring $\alpha(k, E)$: In ablation scoring, we directly replace the actual neuron activation with our simulated neuron activations, and measure how much this changes the final model outputs. Unlike correlations scoring, for ablation scoring we need to make sure simulated activations have the right scale. This leads us to ablation simulated value $s_{abl}(x, E, c, d)$, where c and d are scaling parameters used to match the magnitude of predicted activations with actual neuron activations:

$$s_{abl}(x, E, c, d) = c \cdot s(x, E) + d \tag{11}$$

We measure ablation performance with an objective adapted for classification setting, defined as $\alpha_{init}(k, \mathcal{D}, E, c, d) =$

$$1 - \frac{\sum_{(x,y)\in\mathcal{D}} |L(f_{k\leftarrow s_{abl}(x,E,c,d)}(x), y) - L(f(x), y)|}{\sum_{(x,y)\in\mathcal{D}} |L(f_{k\leftarrow\mu}(x), y) - L(f(x), y)|} \tag{12}$$

where $f(x)$ is the output of the target model, L is the loss function (e.g. cross-entropy-loss), $f_{k\leftarrow s_{abl}(x,E,c,d)}(x)$ indicates replacing the activations of neuron k with the simulated values $s_{abl}(x, E, c, d)$, while $f_{k\leftarrow\mu}(x)$ means replacing the neuron k's activation with its mean value. This is the same as the ablation objective of (Bills et al., 2023), except we have replaced Jensen-Shannon divergence with cross-entropy loss as it is a more relevant metric in the classification task. Note this formulation requires using $\mathcal{D}$ where we have access to ground truth labels, which is the case when using validation data.

To get the parameters c, d , we optimize with gradient descent on the validation split:

$$c^*, d^* = \arg\max_{c,d} \alpha_{init}(k, \mathcal{D}^{\text{val}}_{\text{probe}}, E, c, d) \tag{13}$$

Our final simulation score is then evaluated on the test split, and defined as:

$$\alpha(k, E) = \alpha_{init}(k, D^{\text{test}}_{\text{probe}}, E, c^*, d^*) \tag{14}$$

A perfect simulation will reach $\alpha(k, E) = 1$, while random guess should receive a score of 0.

Here we used optimization to find the parameters c and d. (Bills et al., 2023) instead calculated c and d based on correlation between the predicted and true activation. We evaluate the difference between these choices in Appendix B.5, and find our optimization method gives noticeably higher ablation scores, but with a higher computational cost.

Finally we note that our SigLIP based simulation pipeline is much more computationally efficient than the GPT-4 pipeline of (Bills et al., 2023) because we do not need to recalculate image embeddings when simulating a new explanation. This allows us run simulation on the entire test split of probing data (10,000 images for ImageNet), which is many orders of magnitude larger than the number of samples (Bills et al., 2023) used for simulations.

5. Experiment Results

Setup We mostly focus our analysis on second to last layer features similar to (Kalibhat et al., 2023) and (Bykov et al., 2023) because they are the highest level features learned by the model, and are the features used when transfer learning from that model. Additionally, in the ResNet (He et al., 2016) family of models, they are are CNN channels followed by a global avg pooling layer, meaning they can be meaningfully understood (and simulated) either as having 2d or scalar activations without losing any information.

We evaluate two variants of our method: **Linear Explanation (Label)**, which uses the labels in $\mathcal{D}_{\text{probe}}$ to construct

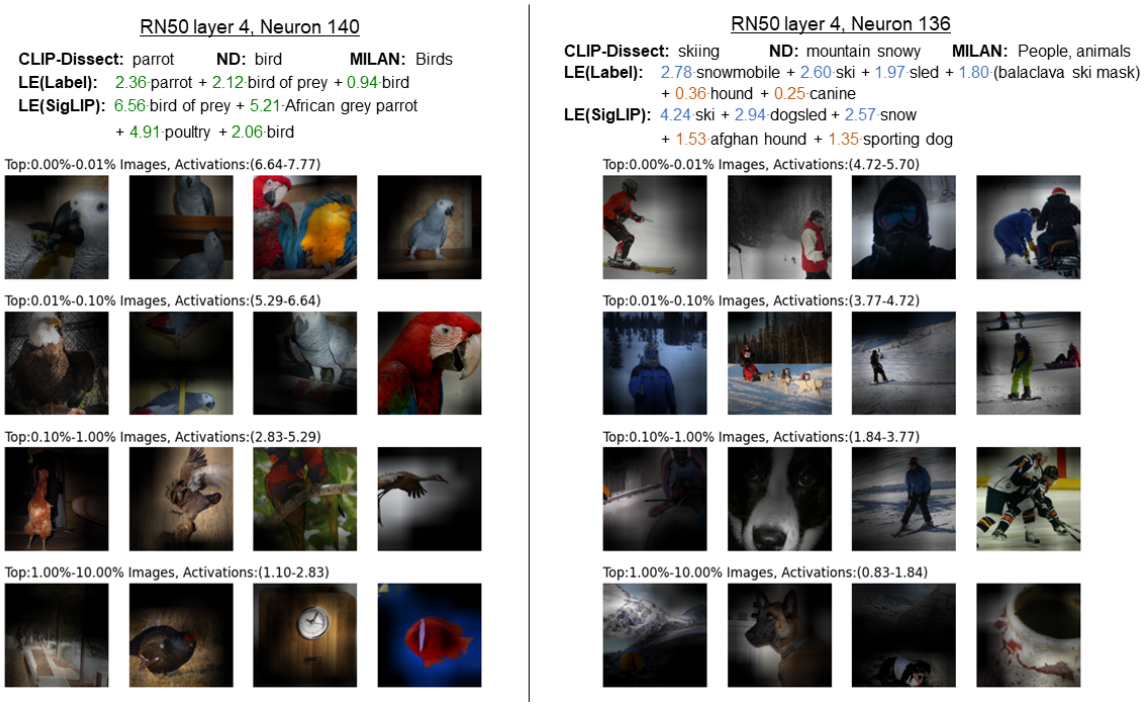

Figure 4. Descriptions and highly activating images from different ranges of example neurons. We can see Linear Explanation provides a more complete description than baselines in both cases.

P, as well as superclass labels in the case of ImageNet and CIFAR-100. The second variant is **Linear Explanation (SigLIP)**, which uses SigLIP-ViT-L-16-384 model (different from our simulator model) to construct the concept activation matrix P.

$\mathcal{D}_{\text{probe}}$**:** As probing data we use the validation dataset of the dataset the model was trained on. We randomly split this probing data into train(70%), validation(10%) and test(20%) splits. Since we simulate on the entire test split, our setting corresponds to the most challenging *random-only* setting from (Bills et al., 2023).

Concept set $\mathcal{S}$**:** For **LE (Label)**, the concept set is the label names in the original dataset (+ superclass names). For **LE(SigLIP)**, we use union of the label names, the labels in Broden (Bau et al., 2017), and a list of 6800 English nouns. For SigLIP we filtered these concepts to only use concepts whose average top-5 activation in $\mathcal{D}_{\text{probe}}$was ≥ 0.5 to avoid using concepts not present in the data. See Appendix B.2 for an ablation study using different concept sets.

5.1. Qualitative results

In Figure 3, we display example **LE(Label)** explanations, as well as an area chart visualization for a few example neurons. This choice of visualization was inspired by (Goh et al., 2021), as it can visualize the neuron behavior across the entire activation range, and with our label based method we can construct them automatically. To construct this, we divided neuron activations into 8 evenly spaced buckets between 0 and the max activation of the neuron, and then plot fraction of inputs within that activation range that belong to each class/superclass. Note the later buckets have much fewer input in them. We can see that these neurons are very well explained by the Linear Explanation, and LE can help reveal polysemanticity across different activation ranges, such as Neuron 136 which activates on dog related concepts on lower activations(orange region), while its top activations are snow related concepts(blue region). Existing methods miss the dog related role and only explain the top activations, as seen in Figure 4. We display explanations from ours and other methods in a more traditional way by visualizing inputs from different activation ranges in Figure 4, and for many more neurons in Appendix C.3.

5.2. Simulation results: Correlation Scoring

Table 2 shows the average correlation scores between simulated and actual neuron activations, across many different models and network architectures, including both CNNs and ViTs. We can see Linear Explanations, especially SigLIP, significantly outperform existing methods, reaching around 0.4 correlation on average, twice as high as existing methods. There is relatively large variance in how interpretable individual units are between different architectures and datasets, but performance between methods is quite consistent. The

Target model	Network Dissection	MILAN	CLIP-Dissect	LE (Label)	LE (SigLIP)
ResNet-50 (ImageNet)	0.1242 ± 0.002	0.0920 ± 0.002	0.1871 ± 0.002	0.2924 ± 0.002	**0.3772 ± 0.002**
ResNet-18 (Places365)	0.2038 ± 0.005	0.1557 ± 0.005	0.2208 ± 0.005	0.3388 ± 0.004	**0.4372 ± 0.003**
VGG-16 (CIFAR-100)	n/a	n/a	0.2298 ± 0.004	0.4330 ± 0.004	**0.4970 ± 0.004**
ViT-B/16 (ImageNet)	n/a	n/a	0.1722 ± 0.004	0.3243 ± 0.005	**0.3489 ± 0.005**
ViT-L/32 (ImageNet)	n/a	n/a	0.0549 ± 0.002	0.1879 ± 0.004	**0.2182 ± 0.004**

Table 2. Average correlation scores between simulated and actual neuron activations, across all neurons in the second to last layer of the respective models. For ViT models we report the MLP neurons in the last transformer block. We do not include Network Dissection and MILAN results for the last two models, as those methods are designed for 2d activations, while the final layers of these models have effectively scalar activations.

Target model	Network Dissection	MILAN	CLIP-Dissect	LE(Label)	LE(SigLIP)
ResNet-50 (ImageNet)	0.0165 ± 0.0003	0.0137 ± 0.0003	0.0215 ± 0.0004	0.0433 ± 0.0007	**0.0727 ± 0.0008**

Table 3. Average Ablation scores between simulated and actual neuron activations, average over all neurons in layer4.

average length of our LE explanations are shown in Table 4(App. A.4). Correlation scores for other layers of a network are discussed in Appendix C.1. In Table 5(App. B.1) we perform an ablation study by using a different simulator model, showing that the trends we observe here hold across different simulators.

5.3. Simulation results: Ablation scoring

In Table 3, we score the explanations using ablation scoring, reporting the average score across all neurons of layer 4 in ResNet-50. We find that existing methods perform very poorly under ablation scoring, with all methods averaging scores of at most 0.02, where simply predicting mean activation on all inputs would give a score of 0, and a perfect prediction can reach 1. Our methods improve significantly over existing methods, with Linear Explanation (SigLIP) reaching 0.0727 average ablation score, more than 3 × better than previous methods, but still score quite low overall. This highlights the need for further refinement in neuron explanation methods. This is consistent with results of (Bills et al., 2023) in language models, where they found most neuron explanations scored very close to 0 under random-only ablation scoring. We also found that there is a clear quadratic relationship between the correlation and ablation score of an explanation across all explanation methods, as shown in Figure 5. This shows correlation score is predictive of ablation score.

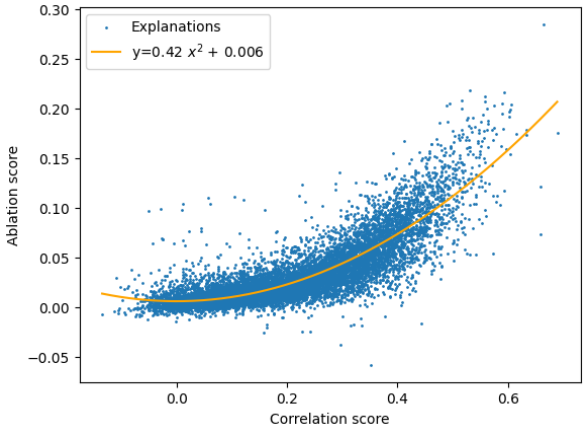

Figure 5. The relationship between correlation and ablation score of different explanations.

Discussion. Overall we found that **LE(SigLIP)** outperformed **LE(Label)** by a significant margin in our evaluations. We believe this can be attributed to a few causes: First, LE(SigLIP) can use a larger concept set, and as such can detect a wider range of concepts. Second it is likely more aligned with the simulator model, i.e. thinks of concepts in a more similar manner to the simulator. In contrast, LE(Label) utilizes some concepts that are very hard for a simulator to understand based on name alone, such as *leporid*, which is a superclass of rabbits. In section B.1 we tested using a simulator from a different model family, but SigLIP's advantage still persisted. Regardless, it may be beneficial to use LE(Label) in some cases, for example to have more transparency and consistency in the explanations.

Computational Efficiency. Both of our explanation methods take around 4 seconds per explained neuron on a machine with a single Tesla V100 GPU. In addition, **LE(SigLIP)** has a one-time cost of a single forward pass of $\mathcal{D}_{\text{probe}}$ through the SigLIP model which takes around

30 minutes. This does not need to be recalculated when explaining new neurons or models with the same data.

6. Related Work

6.1. Automated Interpretability in Vision

Several methods have been proposed to automatically explain the roles of individual neurons in neural networks. Network Dissection (Bau et al., 2017) is the first and likely most popular. They use a dataset with pixel-wise labeled concepts, and try to find concepts with high Intersection over Union (IoU) with binarized neuron activations. This method has a few downsides, such as treating neuron activations as binary and inability to detect concepts missing from their annotated dataset. Compositional Explanations (Mu & Andreas, 2020) extends Network Dissection to deal with polysemantic neurons, that may activate on multiple unrelated concepts, by searching for logical compositions of concepts, i.e. a neuron could activate on Cat OR Dog. This is similar to our method in that they propose a way to compose a more complex explanations out of simpler primitive concepts. However we believe linear composition is preferable over logical composition because it naturally operates on scalar activations while logical composition requires binarizing neuron activations, and linear functions are much faster to search for than logical formulas are.

Recently (Rosa et al., 2023) proposed an Extension to Compositional Explanations, addressing the problem that Compositional Explanations only explain the very highest activations. This was motivated by findings similar to our Section 2 showing that lower activation ranges are also important for network predictions. They propose dividing the activation range of a neuron into 5 buckets, and generating a separate compositional explanation for each. While this does provide a more complete picture of the neurons, the explanation is getting rather hard to understand, with 15 concepts per neuron that interact in complicated ways. We believe Linear Explanations are able achieve a similar level of completeness of description with much less complexity.

Other relevant methods include MILAN (Hernandez et al., 2022) which trains a neural net to describe the most highly activating inputs of a neuron, DnD (Bai et al., 2024) which utilizes pre-trained models to produce generative explanations of highest activating inputs, (Bau et al., 2020) who propose doing Network Dissection with a Segmentation model instead of labeled data, and (Bykov et al., 2023) who propose explaining neurons with Compositions of validation dataset labels similar to our LE(Label), but they use logical compositions instead of linear composition, and AUC to evaluate instead of ability to predict neuron activations.

Finally, recent papers CLIP-Dissect (Oikarinen & Weng, 2023) and FALCON (Kalibhat et al., 2023) have proposed methods that don't require labeled concept information at all by relying on supervision from multimodal models such as CLIP (Radford et al., 2021). This is related to how our LE(SigLIP) works, but both previous methods only focus on explaining the very highest activating inputs.

6.2. Language Models Can Explain Neurons in LMs

(Bills et al., 2023) proposes an Automated Interpretability approach to explain individual neurons in large language models. They propose an explanation method similar to MILAN (Hernandez et al., 2022), and very different from ours, adapted to a language setting. They show a large model (GPT-4) the most highly activating inputs of a neuron and ask it to find what they have in common as the explanation for that neuron, which is studied in more detail by (Lee et al., 2023). (Bills et al., 2023) also propose a new evaluation method: simulation, which evaluates explanations based on how well they can be used to predict activations on new inputs, which we improve upon and extend to vision setting in this work.

6.3. Unreliability of explaining only top activations.

Recent work (Nanfack et al., 2023; Srivastava et al., 2023) have shown that Neuron explanations based on only highest activations are not robust, and can be be easily manipulated. Similarly, feature visualizations can be easily fooled as shown by (Geirhos et al., 2023). In light of this, our hope is that more holistic explanations explaining entire activation range are less susceptible to such attacks.

6.4. Linear Probes

Linear Probing is a common method for finding explainable neurons in language models (Alain & Bengio, 2018; Sajjad et al., 2022; Gurnee et al., 2023; Fong & Vedaldi, 2018). In linear probing, a (sparse) linear model is trained to predict a concept based on neuron activations, with the goal to understand if/where the network represents that concept. Our method is effectively the *inverse* of linear probes, where we instead learn a sparse linear function of concepts to predict a neuron activation.

7. Conclusion

We have shown that only describing highly activating inputs as done by previous work is not sufficient to understanding a neuron, and proposed a new solution called **Linear Explanations**. Our method explains neurons as linear combination of concepts that produces highly accurate and complete neuron descriptions that are still easy to comprehend. Additionally, we developed a new way to more rigorously evaluate neuron explanations in vision models via simulation.

Acknowledgements

This work is supported in part by National Science Foundation (NSF) awards CNS-1730158, ACI-1540112, ACI-1541349, OAC-1826967, OAC-2112167, CNS-2100237, CNS-2120019, the University of California Office of the President, and the University of California San Diego's California Institute for Telecommunications and Information Technology/Qualcomm Institute. Thanks to CENIC for the 100Gbps networks. T. Oikarinen and T.-W. Weng are supported by National Science Foundation under Grant No. 2107189 and 2313105. T.-W. Weng also thanks the Hellman Fellowship for providing research support.

Impact Statement

Our paper proposes improved methods to understand neural networks, and as such we expect it's impact on society to be mostly positive. While our method is only a small part of this, we believe better understanding of networks reduces the chances of harm caused by deploying unsafe or unreliable models in important settings. One potential downside of explanations is that they risks giving the illusion of understanding, without actually being faithful to the model. We hope to have reduced the chances of this with significant focus on better evaluations of neuron explanations.

References

Alain, G. and Bengio, Y. Understanding intermediate layers using linear classifier probes, 2018.

Bai, N., Iyer, R. A., Oikarinen, T., and Weng, T.-W. Describe-and-dissect: Interpreting neurons in vision networks with language models, 2024.

Bau, D., Zhou, B., Khosla, A., Oliva, A., and Torralba, A. Network dissection: Quantifying interpretability of deep visual representations. In *CVPR*, 2017.

Bau, D., Zhu, J.-Y., Strobelt, H., Lapedriza, A., Zhou, B., and Torralba, A. Understanding the role of individual units in a deep neural network. *PNAS*, 2020.

Bills, S., Cammarata, N., Mossing, D., Tillman, H., Gao, L., Goh, G., Sutskever, I., Leike, J., Wu, J., and Saunders, W. Language models can explain neurons in language models. https://openaipublic.blob.core.windows.net/neuron-explainer/paper/index.html, 2023.

Bricken, T., Templeton, A., Batson, J., Chen, B., Jermyn, A., Conerly, T., Turner, N., Anil, C., Denison, C., Askell, A., Lasenby, R., Wu, Y., Kravec, S., Schiefer, N., Maxwell, T., Joseph, N., Hatfield-Dodds, Z., Tamkin, A., Nguyen, K., McLean, B., Burke, J. E., Hume, T., Carter, S., Henighan, T., and Olah, C. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023. https://transformer-circuits.pub/2023/monosemantic-features/index.html.

Bykov, K., Kopf, L., Nakajima, S., Kloft, M., and Höhne, M. M. Labeling neural representations with inverse recognition. In *NeurIPS*, 2023.

Cunningham, H., Ewart, A., Riggs, L., Huben, R., and Sharkey, L. Sparse autoencoders find highly interpretable features in language models, 2023.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., DasSarma, N., Drain, D., Ganguli, D., Hatfield-Dodds, Z., Hernandez, D., Jones, A., Kernion, J., Lovitt, L., Ndousse, K., Amodei, D., Brown, T., Clark, J., Kaplan, J., McCandlish, S., and Olah, C. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021. https://transformer-circuits.pub/2021/framework/index.html.

Elhage, N., Lasenby, R., and Olah, C. Privileged bases in the transformer residual stream, 2023. URL https://transformer-circuits.pub/2023/privileged-basis/.

Erhan, D., Bengio, Y., Courville, A., and Vincent, P. Visualizing higher-layer features of a deep network. *University of Montreal*, 1341(3):1, 2009.

Fong, R. and Vedaldi, A. Net2vec: Quantifying and explaining how concepts are encoded by filters in deep neural networks. In *ICCV*, 2018.

Geirhos, R., Zimmermann, R. S., Bilodeau, B., Brendel, W., and Kim, B. Don't trust your eyes: on the (un)reliability of feature visualizations, 2023.

Goh, G., Cammarata, N., Voss, C., Carter, S., Petrov, M., Schubert, L., Radford, A., and Olah, C. Multimodal neurons in artificial neural networks. *Distill*, 2021. doi: 10.23915/distill.00030. https://distill.pub/2021/multimodal-neurons.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In *ICML*, 2017.

Gurnee, W., Nanda, N., Pauly, M., Harvey, K., Troitskii, D., and Bertsimas, D. Finding neurons in a haystack: Case studies with sparse probing, 2023.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

Hernandez, E., Schwettmann, S., Bau, D., Bagashvili, T., Torralba, A., and Andreas, J. Natural language descriptions of deep visual features. In *ICLR*, 2022.

Kalibhat, N., Bhardwaj, S., Bruss, C. B., Firooz, H., Sanjabi, M., and Feizi, S. Identifying interpretable subspaces in image representations. In *ICML*, 2023.

Lee, J., Oikarinen, T., Chatha, A., Chang, K.-C., Chen, Y., and Weng, T.-W. The importance of prompt tuning for automated neuron explanations. In *NeurIPS ATTRIB Workshop*, 2023.

Miller, G. A. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

Mu, J. and Andreas, J. Compositional explanations of neurons. In *NeurIPS*, 2020.

Nanfack, G., Fulleringer, A., Marty, J., Eickenberg, M., and Belilovsky, E. Adversarial attacks on the interpretation of neuron activation maximization, 2023.

Oikarinen, T. and Weng, T.-W. Clip-dissect: Automatic description of neuron representations in deep vision networks. In *ICLR*, 2023.

Olah, C., Cammarata, N., Schubert, L., Goh, G., Petrov, M., and Carter, S. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. https://distill.pub/2020/circuits/zoom-in.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., and Sutskever, I. Learning transferable visual models from natural language supervision, 2021.

Ribeiro, M. T., Singh, S., and Guestrin, C. "why should i trust you?" explaining the predictions of any classifier. In *KDD*, 2016.

Rosa, B. L., Gilpin, L. H., and Capobianco, R. Towards a fuller understanding of neurons with clustered compositional explanations. In *NeurIPS*, 2023.

Sajjad, H., Durrani, N., and Dalvi, F. Neuron-level interpretation of deep nlp models: A survey. In *TACL*, 2022.

Srivastava, D., Oikarinen, T., and Weng, T.-W. Corrupting neuron explanations of deep visual features. In *ICCV*, 2023.

Wong, E., Santurkar, S., and Madry, A. Leveraging sparse linear layers for debuggable deep networks. In *ICML*, 2021.

Yu, Q., He, J., Deng, X., Shen, X., and Chen, L.-C. Convolutions die hard: Open-vocabulary segmentation with single frozen convolutional clip. In *NeurIPS*, 2023.

Zhai, X., Mustafa, B., Kolesnikov, A., and Beyer, L. Sigmoid loss for language image pre-training. In *ICCV*, 2023.

Zhou, B., Khosla, A., Lapedriza, A., Oliva, A., and Torralba, A. Object detectors emerge in deep scene cnns. In *ICLR*, 2015.

Zimmermann, R. S., Klein, T., and Brendel, W. Scale alone does not improve mechanistic interpretability in vision models. In *NeurIPS*, 2023.

A. Appendix: Additional Information

A.1. Limitations

One limitation of our current method is that it does not take into account the location specific activations. This does not change our results on second-to-last layer neurons which is our main focus, but hinders our ability to explain and simulate lower layers well. However this not a fundamental feature of the method, rather an effect of the tools and data we use. Concept-activation matrix P doesn't have to be at the level of images. We can similarly instead run the same method with pixel/superpixel level supervision using for example Broden (Bau et al., 2017) labels, or an open-vocabulary semantic segmentation model such as (Yu et al., 2023). This will however increase the computational cost as we will have to deal with much larger P, and is a limitation we aim to address in the future.

Another limitation is that the simulation scores we achieve, while better than existing methods are still quite low, especially ablation. While this is in line with previous work (Bills et al., 2023), it is worth discussing. This could be caused by 3 things: 1. Neurons are inherently not interpretable, 2. Our explanations are not good enough or 3. Our simulator is not good enough. While significant gains can definitely be made improving the simulator and explanations (and they can easily be replaced in our pipeline by more powerful models in the future), we believe a large portion of the challenge is caused by the neurons inherently not having a simple explanation. However, recent work in language models (Bricken et al., 2023; Cunningham et al., 2023) shows promise that we can generate more interpretable units of inspection via methods like sparse autoencoders, to which our method can be applied out-of-the-box.

A.2. Additional Related Work: Input Importance

A lot of classical explainable AI methods are focused on a different problem from ours, specifically that of **Input Importance**, where the goal is to answer the question: *Which parts of input x are the most important in the model making prediction $f(x)$?*. In contrast, our work is a neuron explanation method, where the goal of our work is to answer the question: *Given a neural network model f, what are the functionalities of each individual neuron?*.

For example, the LIME paper (Ribeiro et al., 2016) is a popular Input Importance work using linear methods for that task. We will use it to illustrate the major difference between the input importance methods and our neuron explanation method.

3 main differences between our method and LIME (Ribeiro et al., 2016):

1. **Goal**: LIME aims to explain the final prediction of a model with respect to an input, while we explain a single hidden layer neuron.
2. **Scope**: We produce global explanations, while LIME learns a local explanation that only works close to a specific input.
3. **Input:** Lime learns a linear model in terms of input features (or groups of input features), while our method creates an explanation based on interpretable concepts.

A.3. Greedy search algorithm

Algorithm 1 shows the details of our greedy search procedure discussed in Section 3. For the tolerance ϵ, we only add another concept if it improves the correlation(on validation set) between predicted and actual neuron values by more than ϵ. In our experiments, (v, r, ϵ) are set to be 10, 10, 0.02 respectively.

A.4. Explanation length

Table 4 shows the average explanation lengths of our methods for second to last layers of different models. We can see that explanation lengths vary quite a bit, with in particular ViT neurons having short explanations. We think this is likely caused by the fact that the MLP neurons we studied in ViT seem to be more bimodal, either being very uninterpretable or rather monosemantic and interpretable, both of which result in short explanations in our case. If more similar explanation lengths are desired, this can be achieved by tuning the tolerance parameter of the greedy search for each model.

Algorithm 1 Greedy search. Python pseudo-code.

```
SC = {} #selected concepts
BC = {} #bad concepts
ρ* = 0 #best correlation
while |SC| < v do
  CC = {}
  while |CC| < r do
    CC.add(argmax_{i∉CC∪SC∪BC}(w_{k,i}))
  end while
  ρ' = ρ* #current best corr
  for j in CC do
    model = train_model(P^train_{SC∪j}, q^train_k)
    ρ = get_correlation(model, q^val_k)
    if ρ < ρ* + ϵ then
      BC.add(j) #concept is bad
    else if ρ > ρ' then
      ρ' = ρ
      best_c = j
    end if
  end for
  if ρ' < ρ* + ϵ then
    break #early stop if no improvement
  else
    SC.add(best_c)
    ρ* = ρ'
  end if
end while
return SC
```

Target model	LE (Label)	LE (SigLIP)
ResNet-50 (ImageNet)	4.37	4.68
ResNet-18 (Places365)	4.70	4.25
VGG-16 (CIFAR-100)	7.60	6.08
ViT-B/16 (ImageNet)	1.97	1.82
ViT-L/32 (ImageNet)	1.53	1.51

Table 4. Average Explanation lenghts of our method for second to last layer neurons of different models discussed in Table 2.

B. Appendix: Ablations

B.1. Different simulator model.

Since both our simulator and SigLIP explainer model use a similar (but different) model from the SigLIP family, it is natural to ask whether the good performance is caused by this similarity of models. To address this, we redo our simulation with correlation scoring experiment (Table 2) using a simulator model from original CLIP (Radford et al., 2021) family, namely CLIP-ViT-L-14-336. Results are shown in Table 5. We can see the that all scores are lower than with our original simulator model (SigLIP-SO400M-14-386), likely because this CLIP model is not as powerful and it is not designed for multilabel classification required from a simulator. Our Linear Explanation (SigLIP)'s performance is reduced the most by switching the simulator model, indicating that using a related simulator model does boost it's performance, but it is still the best method with a different simulator.

Target model	Network Dissection	MILAN	CLIP-Dissect	Linear Explanation (Label)	Linear Explanation (SigLIP)
ResNet-50 (ImageNet)	0.1003	0.0789	0.1725	0.2545	**0.2961**
ResNet-18 (Places365)	0.1626	0.1279	0.1951	0.2825	**0.3154**

Table 5. Correlation scores of different explanations in layer4 of the models, using CLIP ViT-L-14-336 as the simulator model. Results are similar to Table 2, but all methods score lower, probably due to weaker simulator model.

B.2. Different concept sets

In table 6, we tested how the choice of concept set affects the results of our LE(SigLIP). We compared our original (class labels + Broden labels + common nouns) against two alternatives: 20k, a set of 20,000 most common English words from (Oikarinen & Weng, 2023), as well as only using the ImageNet class labels. We can see our concept set noticeably outperforms both alternative choices. Interestingly it even outperforms the larger 20k concept set, perhaps because it is more noisy and lacking some more precise terms(class names) useful in describing ImageNet images.

Concept set	Correlation score
Original (8438)	**0.3772**
20k (20,000)	0.3561
ImageNet labels (1000)	0.3378

Table 6. Average simulation correlation scores for final layer neurons of ResNet-50(ImageNet) layer4 of LE(SigLIP) with different choices of concept set. We can see our concept set performs the best. Number in brackets represents the size of the concept set

B.3. Different explainer model

We also conducted a study testing the importance of the choice of explainer model for the performance of LE(SigLIP). In particular, we replaced our original explainer SigLIP-ViT-L-16-384 with CLIP ViT-L-14-336 from the original CLIP paper (Radford et al., 2021). From the results in Table 7 we can see a stronger explainer model makes a big difference, but even when using a weaker CLIP model our method outperforms other methods.

Explainer model	Correlation score
Original (SigLIP-ViT-L-16-384)	**0.3772**
CLIP ViT-L-14-336	0.3243

Table 7. Comparison of different explainer models for LE(SigLIP) on laeyr4 neurons of ResNet-50(ImageNet).

B.4. No greedy search

To assess how important our greedy search procedure described in section 3.2.2 is for explanation quality, we evaluated our method without the greedy search procedure by simply taking the top-k concepts of our relatively sparse w_k (with weights

retrained with top-k concepts only) as the final explanation for the neuron. Simulation results are shown in Table 8. Overall the results are mixed, showing we can reach roughly similar quality explanations without the greedy search, but we still find using it preferable as greedy search can dynamically determine the desired explanation length for each neuron, giving longer descriptions to more complex neurons and simple explanations to monosemantic neurons.

Method	LE(label)	LE(siglip)
Original	0.2924 (4.37)	**0.3772 (4.68)**
No greedy search (top-5)	**0.3136 (5.00)**	0.3767 (5.00)
No greedy search (top-4)	0.2962 (4.00)	0.3548 (4.00)

Table 8. Average correlation score of different explanations for neurons in layer4 of ResNet-50(ImageNet). Number in brackets is the average description length.

B.5. Scaling for Simulation with Ablation Scoring

In section 4 we define our ablation scoring function $\alpha_{init}(k, \mathcal{D}, E, c, d)$. This requires finding the optimal scaling parameters c^* and d^*. For our main results we used the optimization method for finding these parameters:

Optim:

$$c^*, d^* = \arg\max_{c,d} \alpha_{init}(k, \mathcal{D}^{\text{val}}_{\text{probe}}, E, c, d) \tag{15}$$

In contrast, (Bills et al., 2023) use a closed form solution intended to maximize the explained variance to select these parameters. In their method (which we will call **Norm**):

Norm:

Let

$$\rho_{\text{val}}(k, E) = \sum_{x \in \mathcal{D}^{\text{val}}_{\text{probe}}} \frac{\hat{s}(x, E) \cdot \hat{g}(A_k(x))}{|\mathcal{D}^{\text{val}}_{\text{probe}}|} \tag{16}$$

where $\hat{s}$ and $\hat{g}(A_k(x))$ are normalized to have mean 0 and standard deviation 1 on the validation set. Then:

$$c^* = \rho_{\text{val}}(k, E) \cdot \frac{\sigma(\mathcal{D}^{\text{val}}_{\text{probe}}, g(A_k(\cdot)))}{\sigma(\mathcal{D}^{\text{val}}_{\text{probe}}, s(\cdot, E))} \tag{17}$$

$$d^* = -c^* \cdot \mu(\mathcal{D}^{\text{val}}_{\text{probe}}, s(\cdot, E)) + \mu(\mathcal{D}^{\text{val}}_{\text{probe}}, g(A_k(\cdot))) \tag{18}$$

where:

$$\mu(\mathcal{D}, f) = \frac{1}{|\mathcal{D}|} \sum_{x_i \in \mathcal{D}} f(x_i) \tag{19}$$

$$\sigma(\mathcal{D}, f) = \sqrt{\frac{\sum_{x_i \in \mathcal{D}} (f(x_i) - \mu(\mathcal{D}, f))^2}{|\mathcal{D}|}} \tag{20}$$

This is theoretically justified as maximizing the explained variance of the neuron. However, our goal is not to maximize explained variance, which leads us to believe direct optimization might be more effective (**Optim**). In table 9 we compare the two methods when evaluating explanations for layer4 neurons of ResNet-50(ImageNet). We can see the optimization method results in noticeably (20-50%) larger ablation scores overall, but doesn't affect the order between explanation methods.

However, optim requires several (100 in our experiments) backwards passes over the validation data, and as such is much more computationally costly. For second to last layer neurons this can be done very efficiently with only a few seconds per neuron, but for evaluating earlier layers the **Norm** strategy is likely better.

Scaling Method	Network Dissection	MILAN	CLIP-Dissect	LE(Label)	LE(SigLIP)
Norm	0.0118 ± 0.0004	0.0092 ± 0.0004	0.0187 ± 0.0005	0.0334 ± 0.0007	$\mathbf{0.0602 \pm 0.0009}$
Optim	0.0165 ± 0.0003	0.0137 ± 0.0003	0.0215 ± 0.0004	0.0433 ± 0.0007	$\mathbf{0.0727 \pm 0.0008}$

Table 9. Testing the scaling function for ablation scoring with simulation. Average ablation score across all neurons in layer4 of ResNet-50(ImageNet). We can see optimizing the scaling parameters c, d results in a 20-50% increase in ablation scores, while not affecting the ordering between methods.

C. Appendix: Additional Results

C.1. Additional layers.

ResNet lower layer neurons. Table 10 and 11 show the correlation scores of different methods across different layers of ResNet-50(ImageNet) and ResNet-18(Places365). We can see performance significantly degrades on the lower layers, which we think is partially caused by neuron functions being harder to describe, but also because our simulator cannot simulate pixel level activations, which becomes more impactful when describing lower layer neurons. However we can still see Linear Explanation, especially with SigLIP continues to outperform all existing methods. Finally in Table 12 we report the results on two different layers on VGG-16.

Target layer	Network Dissection	MILAN	CLIP-Dissect	Linear Explanation (Label)	Linear Explanation (SigLIP)
layer1	0.0313	0.0430	0.0670	0.1008 (1.98)	**0.2313 (3.32)**
layer2	0.0324	0.0242	0.0527	0.1037 (1.91)	**0.2180 (3.22)**
layer3	0.0794	0.0773	0.1045	0.1308 (2.32)	**0.2652 (3.69)**
layer4	0.1231	0.0920	0.1871	0.2924 (4.37)	**0.3772 (4.68)**

Table 10. Average correlation scores of explanations by different methods on different layers of ResNet-50 trained on ImageNet. We can see correlation scores noticeably decrease on lower layers, but Linear Explanations perform the best on all layers. The number in brackets represents average number of concepts per explanation.

Target layer	Network Dissection	MILAN	CLIP-Dissect	Linear Explanation (Label)	Linear Explanation (SigLIP)
layer1	0.0223	0.0344	0.0603	0.1384 (2.42)	**0.2574 (3.66)**
layer2	0.0261	0.0154	0.0419	0.1351 (2.41)	**0.2803 (3.84)**
layer3	0.0727	0.0768	0.0704	0.1593 (2.73)	**0.2842 (3.71)**
layer4	0.2038	0.1557	0.2208	0.3388 (4.70)	**0.4372 (4.25)**

Table 11. Average correlation scores of explanations by different methods on different layers of ResNet-18 trained on Places365. The number in brackets represents average number of concepts per explanation.

Model	Target layer	CLIP-Dissect	LE(label)	LE(SigLIP)
VGG16 (CIFAR100)	classifier[4] (second to last, fc)	0.2298	0.4330 (7.60)	**0.4970 (6.08)**
VGG16 (CIFAR100)	features[43] (last CNN)	0.2616	0.5107 (7.58)	**0.5430 (6.07)**

Table 12. Average correlation score across neurons in different layers of VGG16(CIFAR100). We tested the second to last layer (features[4]) which is a fully connected layer, as well as the last convolutional layer features[43]. Number in brackets is average explanation length. Both layers reach quite high correlation scores, but require long explanations, indicating the neurons are polysemantic.

Different layers of ViT In table 13, we study the interpretability of different layers in ViT models. We can see that the residual stream neurons are in general noticeably less interpretable than the MLP neurons. This is likely caused by the

fact that the residual stream does not (mostly) have a "priviliged basis", which means individual neurons are not more interpretable than random directions in the activation space. In other layers/architectures such a priviliged basis is created by axis-aligned activation functions such as ReLU. See (Elhage et al., 2023) for more discussion on priviliged bases. Because of this, we focus on analyzing the MLP neurons in our work. This is in line with work investigating individual neurons in transformer language models, which typically focus on these MLP neurons. Interestingly, we found that some of the neurons in the MLP layers were extremely interpretable with the highest correlation scores > 0.9, which we did not see in CNN models. On the other hand, the MLP layer also had several lowly activating/dead neurons that were not particularly interpretable, bringing down the average.

When evaluating the final layer neurons of ViT (both residual stream and MLP), we only recorded the activations of the CLS token, as this is the only part passed on to the classification head, and the other activations do not matter. For earlier ViT layers, we took average across the CLS token and all spatial activations, though further research is needed to better know how important the CLS token is compared to spatial activations at different layers of the network.

Model	Target layer	CLIP-Dissect	LE(label)	LE(SigLIP)
ViT-B/16 (ImageNet)	11/11 - Residual Stream (end)	0.0326	0.0813	**0.1455**
ViT-B/16 (ImageNet)	11/11 - MLP	0.1722	0.3243	**0.3489**
ViT-B/16 (ImageNet)	10/11 - Residual Stream (end)	0.0860	0.1293	**0.2643**
ViT-B/16 (ImageNet)	10/11 - MLP	0.1067	0.1940	**0.3088**

Table 13. Average simulation correlation scores of different methods for different layers of ViT-B-16. We can see MLP layers are more interpretable on average than the residual stream.

C.2. Dead neurons in ViT

When investigating the neurons in Vision Transformer, we came across a curious phenomenon: several neurons of the MLP layers are completely dead, i.e. don't activate on any inputs. Such neurons are not meaningful to explain, and seem to waste model capacity. This was especially noticeable in our Experiments with ViT-L/16, where every single neuron in the last transformer block is dead. We tested this further, and found that the last (23rd) transformer layer does not do anything, and can be deleted from the model without affecting classification accuracy at all. Because of this weirdness, we did not include ViT-L/16 in our main results and instead used ViT-L/32.

In Table 14 we quantify the number of dead neurons in the second to last layer of different models (same layers as in Table 2). We defined a neuron as dead if it's maximum activation was < 0.01 across the probing data. We can see other ViT models suffer from some dead neurons but not very many, with ViT-L/16 being an outlier. The CNN based models have no dead neurons.

Target model	ResNet-50 (ImageNet)	ResNet-18 (Places-365)	VGG-16 (CIFAR-100)	ViT-B/16 (ImageNet)	ViT-L/32 (ImageNet)	ViT-L/16 (ImageNet)
Dead neurons:	0%	0%	0%	5.27%	6.57%	100%

Table 14. Fraction of dead neurons in different models.

In Table 15, we report the average correlation scores on "active" neurons only, i.e.we don't count the dead neurons for models with some dead neurons. Overall this does not make a big change as the fraction of dead neurons was quite small, but most scores improve by around 5% over Table 2, which is roughly the fraction of dead neurons.

Target model	CLIP-Dissect	LE (Label)	LE (SigLIP)
ViT-B/16 (ImageNet)	0.1807 +- 0.004	0.3399 +- 0.005	**0.3629 +- 0.005**
ViT-L/32 (ImageNet)	0.0590 +- 0.002	0.1984 +- 0.004	**0.2254 +- 0.004**

Table 15. Average correlations scores of different explanation methods without dead neurons on ViT. We can see this improves average correlation scores by around 5% over Table 2

C.3. Additional Qualitative Results

In Figures 6, 7, 8, 9, 10 we display our explanations as well as previous work for several neurons from layer4 of ResNet-50, and show that our explanations generally capture the neurons behavior well, even if it doesn't have a simple function. In addition, we notice that the LE(Label) and LE(SigLIP) methods typically produce similar concepts, but SigLIP version often returns higher weights for these. This is likely due to differences in concept activation matrices P used by the two models, and highlights the need to calibrate the magnitude of predictions when simulating to align these with the simulator.

In Figures 11, 12, 13, 14, 15 we display similar figures for other models. Some interesting things we can see are that in the case on extremely monosemantic neurons, such as ViT-B/16 neuron 1541 in Figure 14, our method correctly returns only one concept explanation for that neuron, and that concept is more accurate than that of CLIP-Dissect. We also found a rather interesting neuron activating on both military ships/aircraft as well as bald eagles as shown in Figure 15, which our methods described very accurately.

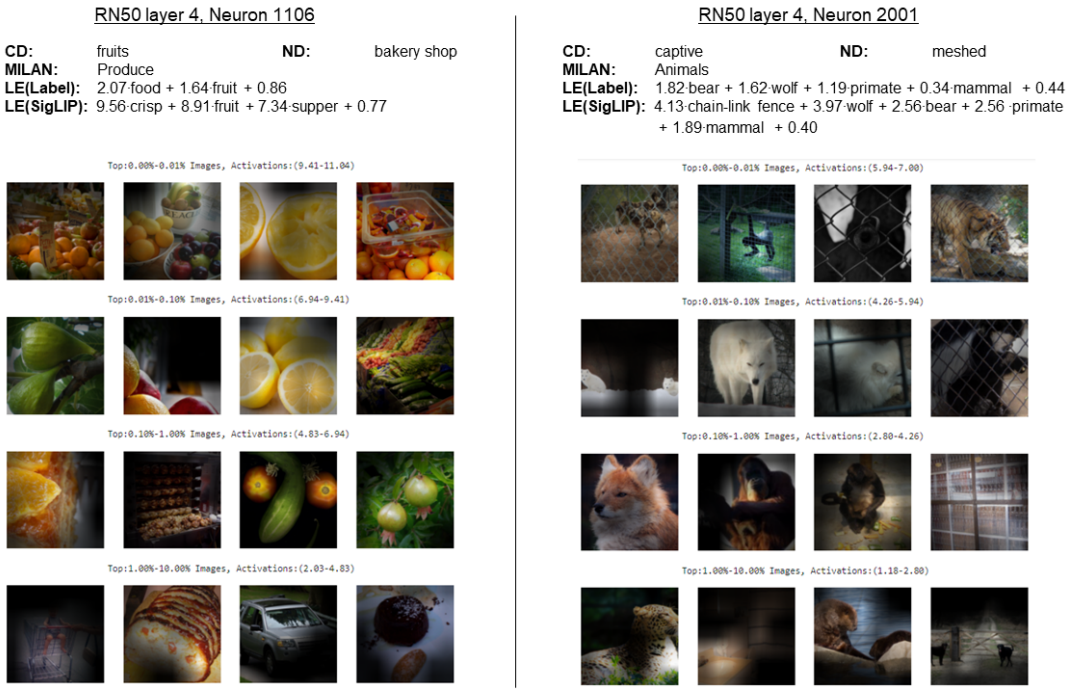

Figure 6. Example interpretable neurons.

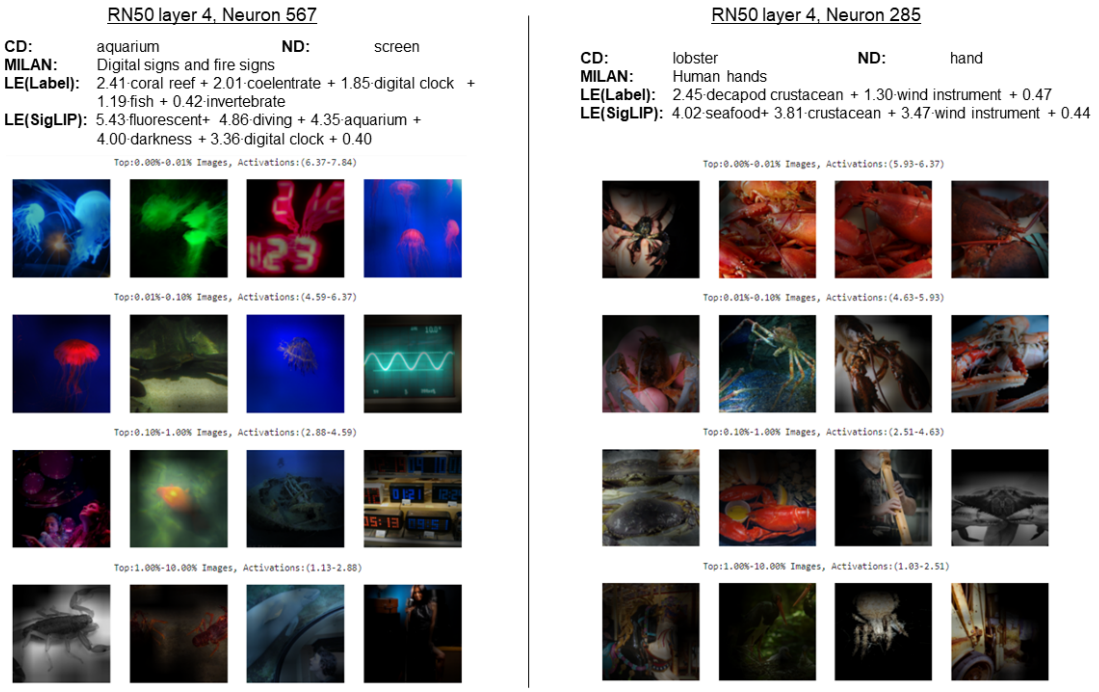

Figure 7. Example interpretable neurons.

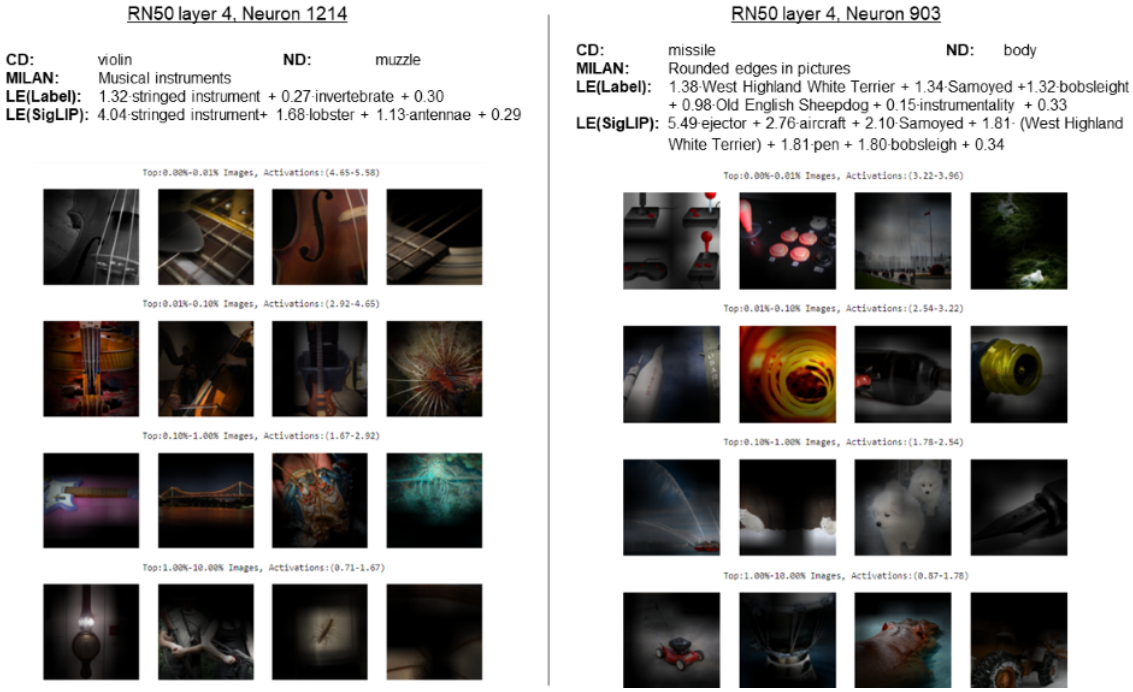

Figure 8. Randomly selected neurons.

Figure 9. Randomly selected neurons.

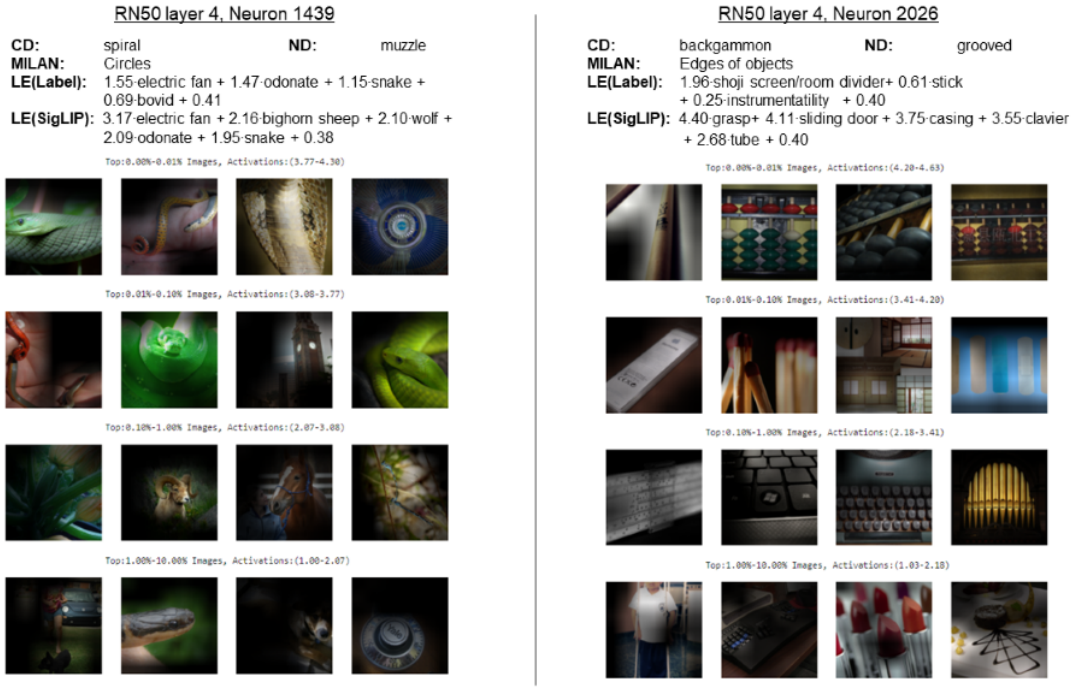

Figure 10. Randomly selected neurons.

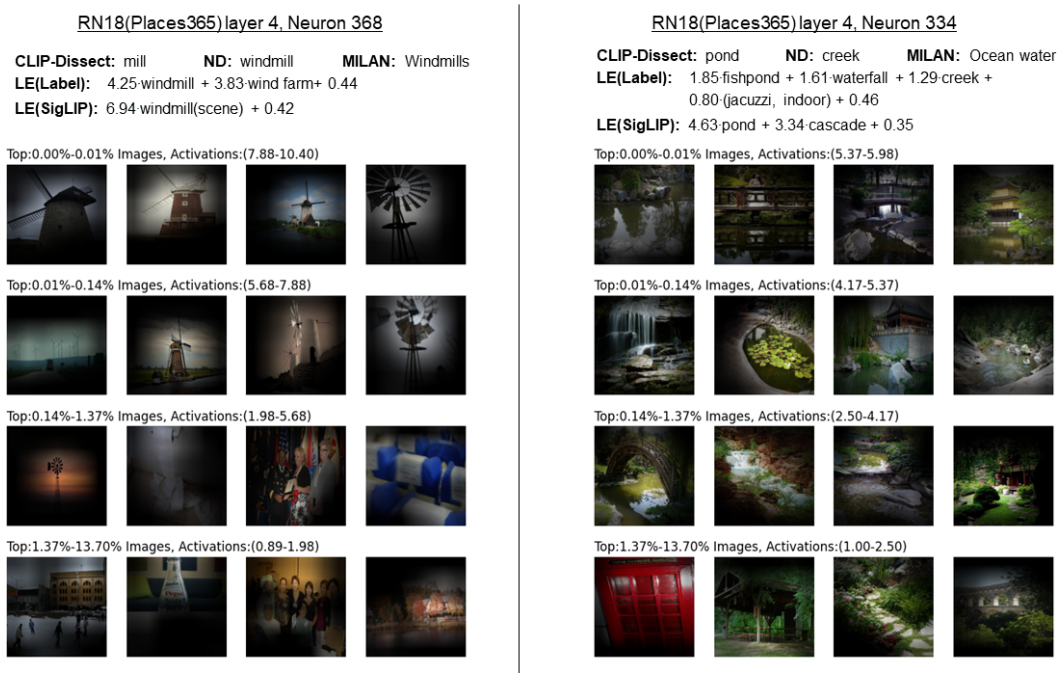

Figure 11. Example interpretable neurons of ResNet-18(Places365).

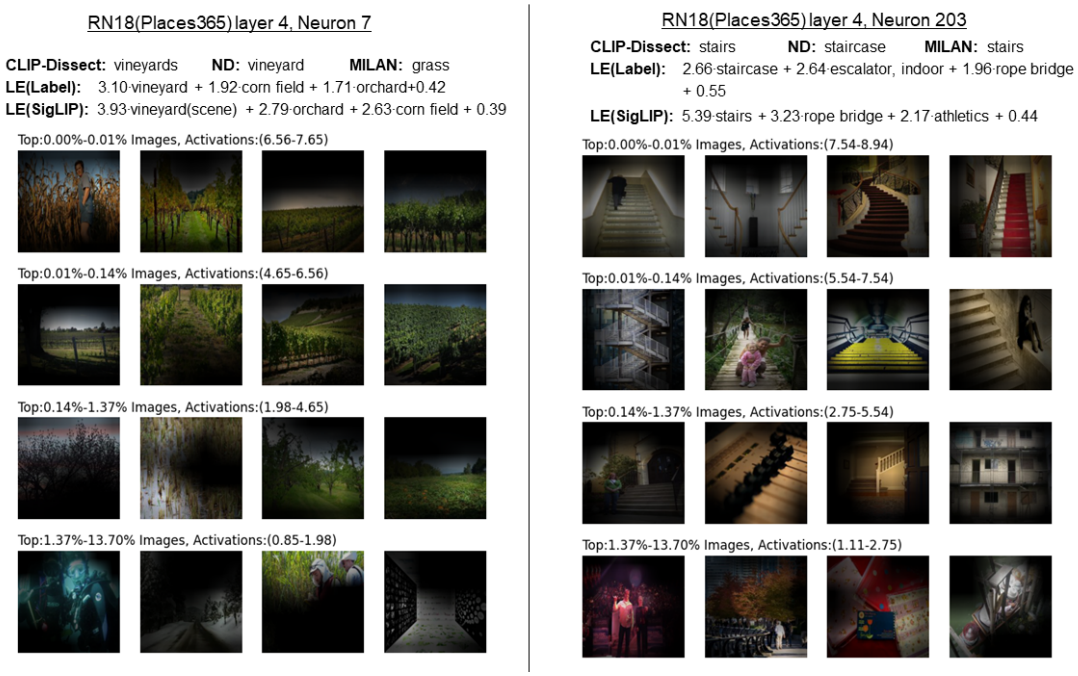

Figure 12. Example interpretable neurons of ResNet-18(Places365).

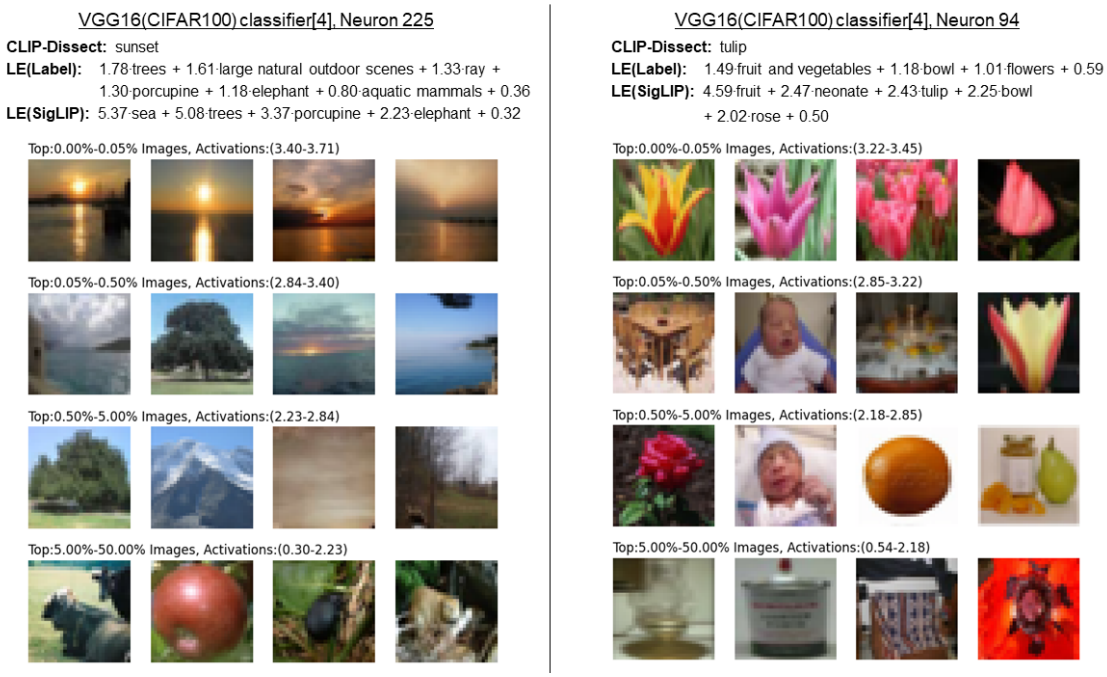

Figure 13. Example interpretable neurons of VGG-16(CIFAR-100).

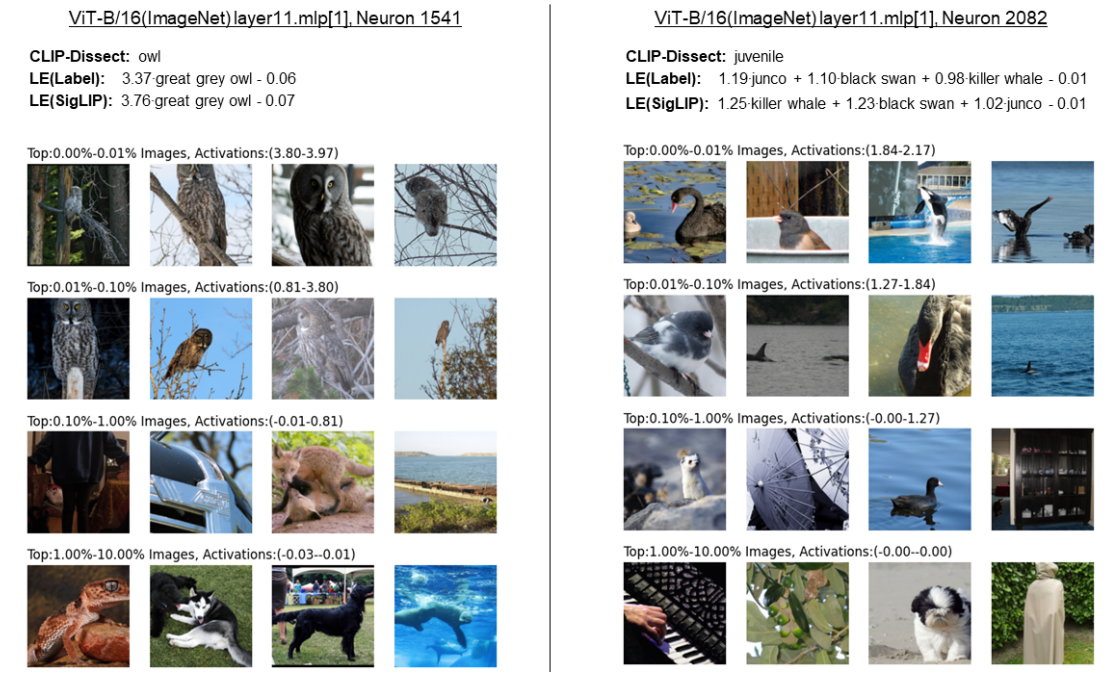

Figure 14. Example interpretable neurons of ViT-B/16(ImageNet).

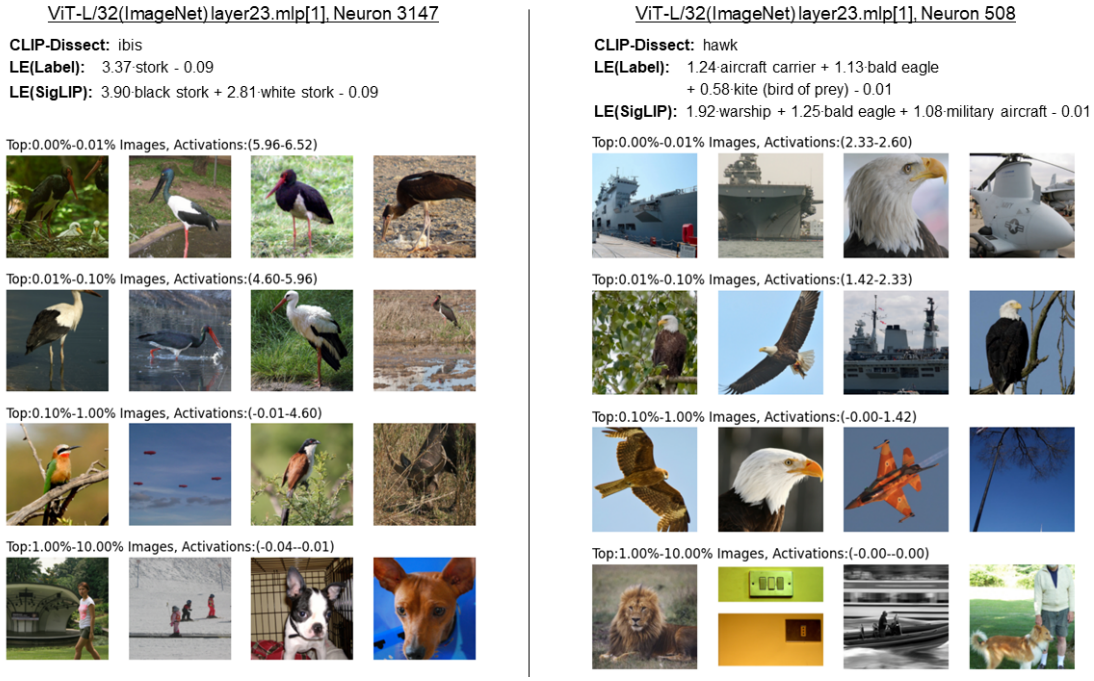

Figure 15. Example interpretable neurons of ViT-L/32.