UAB
Universitat Autònoma
de Barcelona

Bachelor's Thesis
Degree in Computational Mathematics and Data Analytics

Conceptual Description of Neurons based on VLMs: Application to CNN Analysis

Manel Carrillo Maíllo

Supervisors
Maria Vanrell and Guillem Arias
Year
2024/2025
Call
June

Acknowledgements

I would like to express my deepest gratitude to my supervisors, Maria Vanrell and Guillem Arias, for their constant dedication, methodological guidance, and valuable contributions throughout this work. Their vision in the field of neural network interpretability has been key to deepening this line of research and has decisively contributed to my technical and scientific learning.

I would also like to thank the *Computer Vision Center* (CVC) for providing me with access to its computational resources, which have been essential for carrying out and optimizing the experiments performed.

Finally, thanks to all the people who have supported me during this process, both academically and personally.

Resum

In this Bachelor's Thesis, an innovative methodology is presented to analyze and interpret the internal behavior of convolutional neural networks (CNNs), focusing on the semantic description of internal neurons through vision-language models (VLMs).

The proposed approach integrates several stages: receptive field extraction, construction of visual mosaics, design of *prompts* adapted to the abstraction level of each layer, generation of textual descriptions with LLaVA, and subsequent analysis using dimensionality reduction techniques (UMAP) and clustering (KMeans).

The results, obtained on a ResNet-50 trained with ImageNet Fused, highlight consistent conceptual patterns across layers, as well as the semantic evolution of neuron clusters. Moreover, two practical use cases have been developed to demonstrate the applicability of the system: a **semantic debugging** module to identify systematic classification errors, and a **neuron retrieval** tool via natural language queries. This contribution fosters the transparency and interpretability of CNNs, offering a step towards more explainable and auditable neural networks.

Contents

1 Introduction

Convolutional neural networks (CNNs) have revolutionized the field of computer vision, achieving remarkable results in tasks such as *classification*, *detection*, and *segmentation* of images. However, one of their main drawbacks is the lack of interpretability: especially in contexts where transparency or trust is required, such as in the medical domain, autonomous driving, or fraud detection, where a wrong decision can have critical consequences.

Several techniques have been developed to explore the internal representations of CNNs. Methods such as *Grad-CAM* [25], *Saliency Maps* [26], or *Activation Maximization* [31] have enabled the identification of image regions relevant to a specific prediction. Other approaches have leveraged activation visualizations, feature inversion [15], or neuronal contribution analysis [30]. However, these techniques do not allow for an open and detailed understanding of what each neuron represents, nor do they provide a conceptual analysis at the network level.

Despite advances in the field of so-called *explainable artificial intelligence* (*XAI*), we currently lack methods capable of precisely and systematically describing what specific information each neuron in a CNN captures. Most approaches focus on identifying visual regions relevant to the model's final decision, but they do not provide an open and systematic understanding of the internal behavior of neurons. This limitation hinders the detection of biases or conceptual errors and restricts the ability to perform in-depth qualitative analysis of a trained model's behavior. Furthermore, many existing techniques are not scalable and often rely on ad-hoc or subjective visualizations.

An emerging alternative is to study the concepts captured by internal neurons through the images that most strongly activate them, in order to infer their underlying semantic meaning. This line of work has been partially explored in studies such as *Network Dissection* [2], by Bau et al., which proposes to quantify interpretability by aligning neuronal activations with predefined visual labels. However, this approach relies on a closed set of concepts and does not generate open-ended descriptions, nor does it provide a deeper or more expressive semantic interpretation.

This thesis builds on that perspective and expands it through techniques from computer vision and natural language processing to interpret the internal behavior of a convolutional neural network. Specifically, a *ResNet-50*B architecture pretrained on the *ImageNet* dataset [7][1] is employed.

From this architecture, for each neuron, the images that maximally activate its response (i.e., that generate a high neuronal activation) are identified. These images are then analyzed using vision-language models (VLMs), which are capable of generating natural language descriptions from the predominant visual patterns. This strategy enables the construction of a semantic representation of the network's internal behavior, allowing for interpretation neuron by neuron.

This approach opens new possibilities for qualitative explainability, semantic *debugging*, and the identification of conceptual biases within the model. Overall, it proposes an open, scalable, and expressive methodology for the deep understanding of the internal representations of a CNN.

1.1 Contributions

This work presents a contribution structured into three major components, establishing an innovative framework for the open and semantic analysis of the behavior of internal neurons in convolutional neural networks:

[1] We use the *ImageNet Fused* version, a semi-structured subset of ImageNet with images grouped by *WordNet* [18] synsets. A synset (or synonym set) is a grouping of terms that share the same semantic meaning; for example, the synset `n02123045` includes the terms *tabby*, *tabby cat*, *alley cat*, or *domestic cat*

1. **A method for assigning semantic concepts to each individual neuron using vision-language models.**

   This contribution includes:

   - A multimodal *prompting* scheme adapted to the abstraction level of each layer, enabling the generation of precise, expressive, and interpretable descriptions of the concept captured by each neuron.
   - A semantic vectorization process that transforms descriptions into textual and multimodal *embeddings*, ensuring the flexibility and scalability of the system to analyze any CNN and all of its neurons.

2. **A methodological proposal for validating the quality of the descriptions.**

   Several evaluation techniques have been integrated to study the coherence, diversity, and conceptual organization of the results, including:

   - Lexical analysis (TF-IDF, term distribution).
   - Semantic metrics (cosine similarity, cluster internal cohesion, adjusted Rand index).
   - Dimensionality reduction techniques (UMAP) and clustering (KMeans) to visualize semantic evolution across layers.

3. **A proposal for practical application of the results as an explainability tool, structured into two use cases:**

   - *Use Case 1*: A global model analysis tool that enables indexing, exploration, and connection of neurons according to their semantic meaning, offering a structured view of the network's conceptual task.
   - *Use Case 2*: A contextual *debugging* tool that facilitates the detection of neurons responsible for misclassifications and bias patterns, by comparing semantic activations between correct and incorrect predictions.

2 Context

This section presents the fundamental concepts that underpin the proposed approach, including the hierarchical structure of CNNs, the descriptive capabilities of vision-language models, and the semantic representation and comparison mechanisms that will be employed in the subsequent stages of the analysis.

2.1 Convolutional Networks and Neuronal Activations

Convolutional neural networks (CNNs) are a fundamental architecture of *deep learning*, widely used in computer vision tasks such as image classification, object detection, and semantic segmentation. This type of network is composed of a sequence of layers, each with a specific function within the image transformation process.

The typical layers found in a CNN are:

- **Convolutional layers**, which extract visual patterns through the application of filters;
- **Activation layers** such as ReLU, which introduce non-linearity after convolutional layers;
- ***Pooling* layers**, which reduce the spatial resolution of the activation maps;

- **Fully connected layers**, non-convolutional layers in which neurons lose spatial information and all inputs are connected to all outputs. They are responsible for combining information and generating the final prediction.

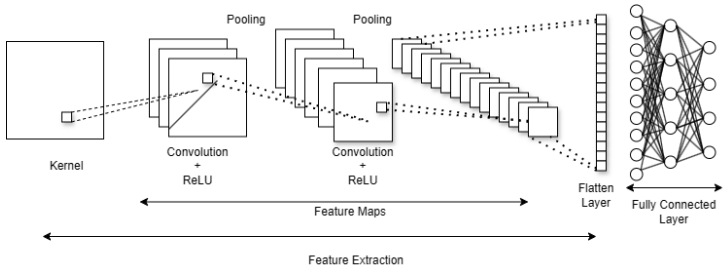


Figure 1: Graphical representation of a convolutional neural network.

Convolutional layers, which most strongly characterize CNNs, are based on the application of filters (also called kernels) over the input image to detect specific visual patterns.

Each filter is a small weight matrix that is repeatedly applied to the image through a sliding operation known as the *sliding window*. At each position, the filter overlaps with a local region of the image, computes a weighted sum of the pixel values, and generates an activation. This process is repeated across the entire image, and the resulting set of values forms a *feature map*.

These *feature maps* indicate the intensity with which the pattern represented by the filter is present in different parts of the image. For instance, a filter may be trained to detect horizontal edges, textures, curves, or other useful visual characteristics. In a typical CNN, dozens or even hundreds of filters per layer are used, each learning to detect a different type of pattern.

Mathematically, given an input image $I \in \mathbb{R}^{H \times W}$, where H is the height and W the width, and a convolutional filter (or kernel) $K \in \mathbb{R}^{k \times k}$, its application is defined as:

$$O(x, y) = (I * K)(x, y) = \sum_{m=0}^{k-1} \sum_{n=0}^{k-1} I(x + m, y + n) \cdot K(m, n) \tag{1}$$

where:

- $O(x, y)$ is the activation produced at position (x, y) of the resulting *feature map*.
- $I(x + m, y + n)$ is the value of the image pixel at position $(x + m, y + n)$.
- $K(m, n)$ is the value of the filter at position (m, n).

This value indicates the filter's response to the analyzed local region, measuring its degree of correspondence with the visual pattern the filter has learned to detect. By sliding the filter across the image, a new representation—the *feature map*—is obtained, highlighting the regions where the pattern is most present.

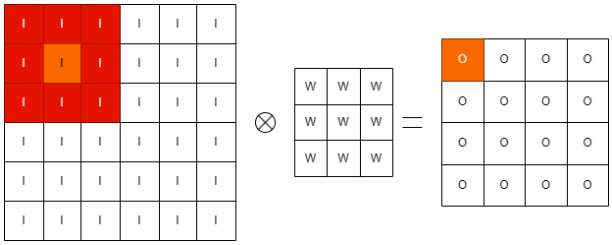


Figure 2: Example of the convolution operation in a CNN.

Meaning and behavior by layer

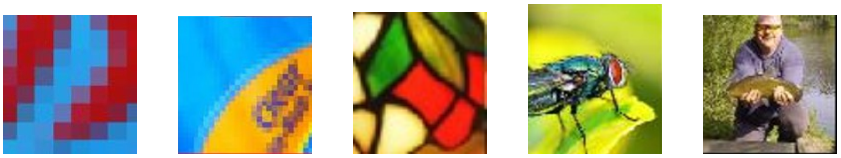
In this study, we employ a *ResNet-50* [9] architecture, a deep convolutional neural network widely used for image classification. This architecture exhibits a hierarchical structure that combines convolution, normalization, and activation layers, organized into four main stages (`layer1` to `layer4`). As we progress through these layers, internal representations evolve from low-level visual features to high-level concepts. For a detailed description of the architecture, see Appendix B.

In the case of ResNet-50, we observe a clear hierarchy in the semantic behavior of neurons across the different stages of the network. In this context, when referring to the `relu` layer, it is understood as the output of the `ReLU` activation applied to the initial convolutional layer (`conv1`), before entering the residual blocks.

- **Initial layers (e.g., `relu`):** Respond to low-level features such as edges, textures, contrasts, or simple geometric patterns. Neurons have a small receptive field and respond strongly to local stimuli [2]. These activations are considered essential visual primitives for subsequent stages.
- **Intermediate layers (e.g., `layer1`, `layer2`):** These convolutional stages begin to integrate multiple elementary patterns to form object parts or more complex visual configurations. Semantic meaning becomes more abstract, and activations reflect structural relationships within the scene.
- **Deep layers (e.g., `layer3`, `layer4`):** These deep convolutional stages detect global visual configurations such as complete objects or semantic scenes. The receptive field of these neurons is large, and their response may depend on the full context of the image.

Finally, it should be noted that fully connected layers are only applied in the final stage of the network, after global pooling, and are not part of the convolutional stages analyzed in this study.

This hierarchical behavior is essential to our work, as it guides the semantic prompting strategy and the layer-wise analysis for generating descriptions with VLMs.



(a) ReLU (b) Layer 1 (c) Layer 2 (d) Layer 3 (e) Layer 4

Figure 3: Progressive activation patterns in a ResNet-50.

[2] The *receptive field* of a neuron in a CNN is the region of the input image that directly influences its activation. In early layers, this field is small and responds to simple local patterns (such as edges or textures), while in deeper layers it can cover larger regions and capture more complex structures.

2.2 Vision-Language Models (VLMs)

Vision-Language Models (VLMs) are multimodal learning systems that integrate visual and textual information within the same model. These models have been trained to generate descriptions, answer questions, or perform conceptual inferences from images, and have become a key tool for tasks such as *image captioning*, *visual question answering*, or multimodal reasoning [27].

General Architecture of a VLM

Vision-language models are designed to seamlessly connect visual and textual information. In general terms, a VLM receives as input an image together with an instruction or *prompt* (i.e., a natural language statement guiding the model's desired output), and generates as output a natural language text that describes or reasons about the visual content. To achieve this multimodal capability, a VLM integrates several components that process, transform, and combine the two modalities.

A VLM combines three main components:

- **Visual encoder**: processes the image and generates *embeddings* representing its content. It may consist of a convolutional neural network (CNN), a visual transformer (ViT), or a contrastively trained model such as CLIP [21].
- **Projection layer**: transforms the visual *embeddings* to be compatible with the language model. This layer learns to map the visual latent space into a space understandable by the textual component.
- **Language model (LLM)**: a generative model based on the *Transformer* architecture [29][3], capable of understanding and producing natural language text. In the context of a VLM, it receives as input the transformed visual *embeddings*, together with a textual *prompt*, and generates a response conditioned by both modalities. Models such as *GPT* [3], *LLaMA* [28], or *Vicuna* [5] are used, trained to capture complex semantic relationships in text and adapt to the visual content provided.

When the model receives an image and a textual instruction (*prompt*), the visual encoder transforms the image into a set of vectors. These vectors are incorporated as special tokens into the language model, which generates a coherent response combining both visual and textual information (see Figure 4).

This architecture constitutes the basis of the systems employed in this study for the open generation of descriptions of neuronal activations. In our case, these images correspond to visual mosaics that synthesize the activation patterns of specific neurons in the network, thus providing a pathway to their semantic interpretability.

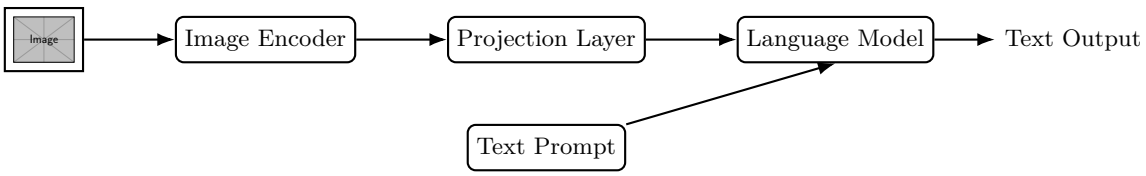


Figure 4: General architecture of a VLM: the visual encoder transforms the image into embeddings, which are projected into the language model space. The textual *prompt* and visual embeddings are combined to generate the textual response.

To quantitatively analyze these descriptions generated by vision-language models, in this work the pro-

[3]Transformers are a neural network architecture introduced by Vaswani et al. [29], based on attention mechanisms that enable processing entire sequences in parallel and capturing long-range relationships between words or tokens. They have become the standard in language models due to their effectiveness and scalability.

duced text is transformed into vector representations that capture its semantic meaning (*embeddings*).

2.3 *Embeddings* and Cosine Similarity

Embeddings are vector representations in a latent space that encode the semantic meaning of textual or visual elements—such as words, sentences, or images—so that their content can be numerically processed by a computer.

Unlike simple and independent representations such as one-hot encoding for text—where each word is represented as an orthogonal vector without considering context—or pixel-based encodings in the case of images, *embeddings* capture semantic or structural relationships between elements. Thus, two sentences with similar meanings, or two images with related visual content, will appear close in the vector space, while those that differ conceptually will be located farther apart.

Two main types of *embeddings* can be distinguished, depending on the nature of the information represented: **textual embeddings**, which capture the semantics of natural language, and **multimodal embeddings**, which integrate visual and textual information within a shared latent space.

Both types of vectors allow the semantic content to be represented computationally and are particularly useful for analytical techniques such as meaning comparison, conceptual clustering, or visualization of latent structures.

Textual *Embeddings*

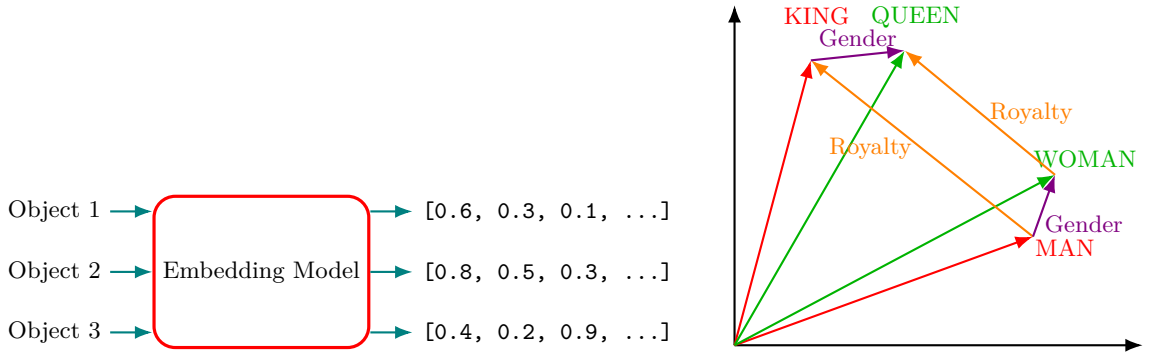
Textual embeddings are obtained from complete sentences using language models such as **Sentence-Transformers** [23]. Unlike traditional word-based models such as Word2Vec [17] or GloVe [20], which assign a single fixed vector per word regardless of context, *Sentence-Transformers* generate contextual representations that take into account the entire sentence.

These models are based on the BERT architecture [8][4], but are optimized to produce sentence-level embeddings through contrastive learning techniques [4][5]. This strategy enables sentences with similar meanings to be projected into close vectors within the latent space, enhancing their semantic comparability.

This ability makes them particularly useful for compactly and informatively representing the descriptions generated by vision-language models, enabling quantitative analysis, visualization, and subsequent semantic clustering.

[4]BERT (*Bidirectional Encoder Representations from Transformers*) is a pretrained architecture based on Transformers that employs bidirectional attention mechanisms to capture the context of words within a text.

[5]Contrastive learning is a technique that trains the model to bring the representations of similar pairs closer together and push apart those of dissimilar pairs in latent space. It is particularly useful for similarity and semantic representation tasks.



(a) Conversion of objects into vector embeddings.

(b) Conceptual embedding space with semantic relations (*royalty*, *gender*).

Figure 5: Process of generating and representing vector embeddings.

Multimodal *Embeddings*

On the other hand, multimodal embeddings are generated by models jointly trained on images and text, such as **CLIP** (Contrastive Language–Image Pretraining) [21]. These models project both visual and textual information into a shared latent space, so that an image and its corresponding description are positioned closely in this space, while unrelated elements are projected farther apart.

This alignment across domains enables the joint representation of visual and linguistic patterns, and is particularly useful for capturing visual aspects that might not be reflected in purely textual representations such as those of *Sentence-Transformers*. This makes multimodal embeddings a powerful tool for the semantic analysis of neuronal activations.

In our case, we explored the combination of representations generated with CLIP and Sentence-Transformers using techniques such as vector concatenation or averaging, with the aim of obtaining a richer and more robust representation of neuronal descriptions.

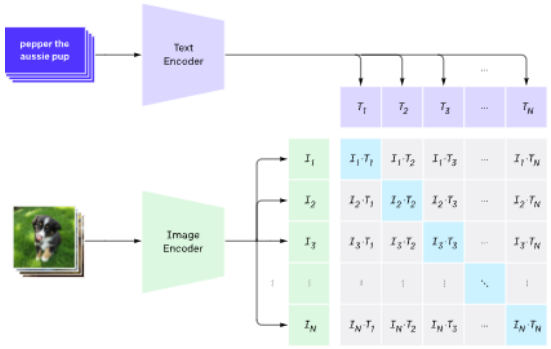


Figure 6: Operation of CLIP: an image encoder and a text encoder project elements from two different domains (visual and textual) into a shared latent space. The model learns to maximize similarity between corresponding image-text pairs and minimize it with the rest. Image adapted from [21].

Both textual and multimodal embeddings provide vector representations in a latent space that preserve the semantic meaning of the described elements. Once these representations are generated, it is crucial to have a metric that quantifies conceptual similarity between vectors to perform operations such as retrieving related elements, clustering descriptions, or analyzing latent relationships.

Cosine Similarity

Cosine similarity is one of the most widely used measures for comparing *embeddings*. It is defined as the cosine of the angle between two vectors $\vec{u}$ and $\vec{v}$, and allows estimating the semantic proximity between their representations, regardless of their magnitude:

$$\text{sim}_{\cos}(\vec{u}, \vec{v}) = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\| \cdot \|\vec{v}\|}$$

This metric is particularly useful in high-dimensional spaces and is widely applied to compare sentences, images, or neuronal activations represented as vectors. In the context of this study, it is used to detect semantic similarities between descriptions, group neurons with conceptually related behaviors, and visualize latent structures using techniques such as UMAP.

3 Related Work

This section presents two related lines of research that ground and contextualize the proposal of this study. On the one hand, existing approaches for interpreting CNNs based on the analysis of individual neurons are reviewed, with the aim of understanding how they capture and represent internal visual patterns. On the other hand, the growing use of language models to automate the generation of semantic explanations of models is explored, facilitating a more open and expressive understanding of their behavior. This dual focus reflects the two conceptual foundations on which the proposed methodology is built.

3.1 Visual Interpretation Based on Activations

Over the years, multiple techniques have been developed to attempt to understand the internal functioning of convolutional neural networks, particularly in computer vision applications. One of the most explored lines has been the direct visualization of internal activations, with the goal of revealing which visual patterns neurons learn at different levels of the network.

Mahendran and Vedaldi [15] proposed a feature inversion technique that reconstructs the original image from the activations of a given layer. This allows analyzing what information each layer retains and observing how the visual representation evolves from low levels to more abstract ones.

Another very influential approach is Grad-CAM (Gradient-weighted Class Activation Mapping) [25]. This technique generates heatmaps from the gradients of the prediction with respect to the last convolutional layers, identifying which regions of the image are most relevant for a given classification. Grad-CAM is useful for visually validating whether the CNN is focusing on the expected regions, such as the face of an animal instead of the background.

A different approach is that proposed by Bau et al. with *Network Dissection* [2], which quantifies the interpretability of each unit in a CNN by associating it with human visual concepts such as "window," "wood texture," or "metal grid." To achieve this, they use a dataset with semantic annotations to compare the regions activated by each neuron with object and attribute masks. This makes it possible to label individual neurons as detectors of specific concepts and measure their specialization.

Despite their usefulness, these approaches share a common limitation: their interpretation is based solely on the visual space, without a direct expression in natural language. This hinders open semantic understanding and the reuse of information by humans or other multimodal systems. Moreover, these techniques often provide local explanations (per image or specific neuron), but do not allow for aggregated analysis of the network's internal behavior, nor comparison across neurons or clusters. They also

do not provide a mechanism for semantically exploring activated units or adapting the desired level of abstraction.

For this reason, recent works have begun to integrate vision and language models to generate more expressive and accessible interpretations.

3.2 VLMs Applied to Semantic Explanation

With the emergence of vision-language models (VLMs), several works have begun to explore the possibility of generating textual descriptions to explain the behavior of internal neurons in a CNN. This approach aims to complement classical visual techniques with more expressive interpretations in natural language.

An example of this line is *CLIP-Dissect*, proposed by Oikarinen and Weng [19], which leverages the multimodal CLIP model to assign open labels to individual neurons. The method selects images that strongly activate a unit and computes their similarity with textual descriptions. Although it simplifies labeling, the system heavily depends on CLIP's internal vocabulary and does not allow adjusting the desired semantic level, often producing overly general or unrepresentative labels.

Other approaches, such as *Describe-and-Dissect* (DnD) [1], combine multiple models to generate one description per neuron based on activating images and subsequent text synthesis. Although providing richer descriptive output, this method involves a complex pipeline with high computational cost and limited flexibility to explore or compare neuronal clusters.

Overall, these proposals point towards a more open explanation of CNNs' internal functioning, but leave key issues unresolved, such as semantic control, neuron comparability, and aggregated analysis of internal behavior. Furthermore, none of these works presents a practical use case demonstrating how generated descriptions can help understand or improve model behavior in real situations.

This work seeks to contribute in this direction with a more modular, scalable, and interpretive approach, applied to concrete scenarios of classification and error analysis.

4 Method

This section presents the proposed method to analyze and interpret the internal behavior of a convolutional neural network at the level of individual neurons. The process combines computer vision techniques, generative language models, and semantic representation to generate an open, quantitative, and scalable explanation of the functioning of each latent unit.

The complete method is divided into four main phases:

1. **Activation extraction**: all images from the dataset (*ImageNet Fused*) are applied to the analyzed CNN (ResNet-50), and internal activations of each neuron are collected for different layers (`ReLU`, `layer1-4`). For each neuron, the 100 receptive fields generating the maximum response are identified, thus representing the visual regions that most strongly activate it.[6]
2. **Visual mosaic construction**: from the most activating receptive fields, a 10×10 mosaic image is built that synthesizes the set of typical visual patterns associated with each neuron. This mosaic provides a compact visual representation of its behavior.
3. **Semantic description generation**: the vision-language model LLaVA (Language and Vision Assistant) is used to generate a coherent textual description of the visual content of each mo-

[6] Approximate number of neurons (channels) per layer: `ReLU` (64), `layer1` (256), `layer2` (512), `layer3` (1024), `layer4` (2048).

saic. A hierarchical and context-aware *prompting* strategy, adapted to the abstraction level of each layer (*layer-aware prompting*), is employed to guide the model's response toward specific semantic patterns (shapes, objects, scenes...).

4. **Conversion to embeddings**: each generated description is encoded into a semantic vector using two language models: (a) `sentence-transformers/clip-ViT-L-14`, which produces a purely textual embedding, and (b) `openai/CLIP`, which generates multimodal embeddings. This vector representation enables the application of quantitative analysis techniques such as visualization (UMAP), clustering (KMeans), or semantic similarity measures.

This method allows the internal neurons of a CNN to be studied not only as numerical filters but as latent semantic detectors, thereby facilitating their interpretation, comparison, and debugging. Figure 7 shows a conceptual diagram of the process.

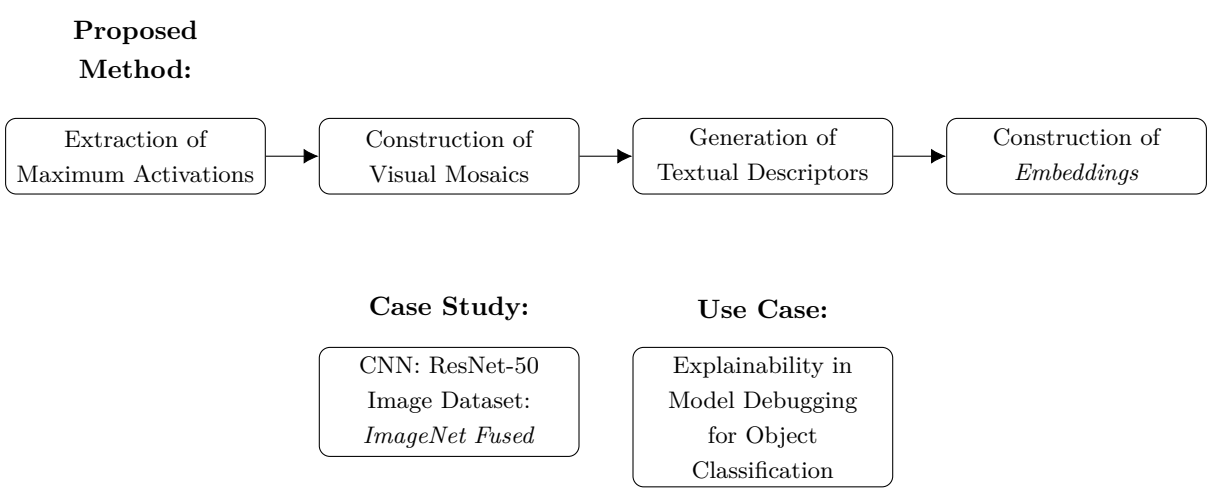


Figure 7: Conceptual diagram of the proposed method and its use case.

4.1 Extraction of Maximum Activations and Construction of Visual Mosaics

To study the internal behavior of a convolutional network and identify the visual patterns that activate its neurons, an initial phase is carried out based on large-scale activation extraction and the construction of visual mosaics.

This process begins by using *hooks* provided by the *PyTorch* framework, a technique that allows intercepting the internal outputs of a layer while the model processes an image, without altering its functioning. In this work, *hooks* were inserted in the 5 layers of ResNet-50 with the aim of recording the raw activations of all neurons while processing the images from the *ImageNet Fused* dataset. This dataset is an optimized version of ImageNet, which preserves its semantic diversity while reducing redundancies and class imbalances (see A for further details).

During processing, for each image and each layer, the following data are systematically collected:

- Activation values (both raw and normalized) of each neuron.
- Spatial coordinates (x, y) within the activation map where maximum responses occur.
- The identifier of the source image that triggered each activation.

After this phase, all activations obtained per neuron are sorted, and the **100 maximum activations** are selected for each one. Based on the recorded spatial coordinates and the architectural parameters of the

network (*stride*, *kernel size*, *padding*), the **receptive field** associated with each activation is computed, and the corresponding fragment of the original image is retrieved. These visual fragments represent the most characteristic stimuli that activate each neuron.

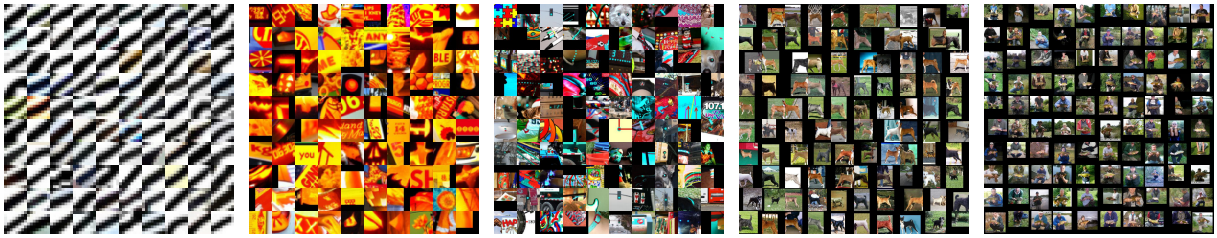
With these 100 fragments, a **visual mosaic** of 10×10 images is built for each neuron, ordered according to activation intensity. This mosaic acts as a visual condenser of the recurring patterns associated with the neuron and forms the basis for the subsequent generation of semantic descriptions.

Although an alternative would be to generate an independent description for each of the 100 fragments, this approach was discarded for three main reasons:

- **Computational efficiency**: generating thousands of individual descriptions, especially when analyzing multiple layers and neurons, would imply a very high cost in terms of time and resources.
- **Semantic coherence**: many fragments show only parts of objects or ambiguous textures, which may lead to incoherent, erroneous, or contextless descriptions. The mosaic offers a global view that helps the model better capture the underlying concept of the neuron.
- **Interpretive clarity**: a visual mosaic synthesizes in a single image the most recurrent patterns, facilitating interpretation both by the model and by a human analyst.

Thus, the mosaic not only optimizes the process but also enhances the quality and usefulness of the generated descriptions.

This procedure is applied to all neurons in the analyzed layers and marks the start of the semantic interpretation process at the neuronal level.



ReLU **Layer 1** **Layer 2** **Layer 3** **Layer 4**

Figure 8: **Evolution of visual patterns across the layers of a ResNet-50**. Each mosaic shows the 100 image fragments that most strongly activate a representative neuron per layer.

4.2 Generation of Textual Descriptions

Once the visual mosaics have been generated for each neuron, the next step is to interpret their content at the semantic level. The goal is to obtain, for each mosaic, a natural language description that synthesizes the predominant visual patterns that activate that neuron.

To address this task, it is necessary to select a vision-language model (VLM) capable of understanding the combined visual stimuli and generating expressive, coherent, and informative descriptions. Since several options are available, a preliminary evaluation was carried out to identify which models perform best in this context.

The alternatives considered are described below:

CLIP Interrogator 2.1 A tool that generates textual descriptions from images by combining visual embeddings and language models. It enables optimized descriptions based on encoded visual information and provides precise control over the level of detail and style of the descriptions. Based on

the CLIP model [21].

BLIP-2 A model designed to generate natural language descriptions from images, noted for its efficiency and descriptive quality. Its modular architecture facilitates the production of coherent textual outputs, with the ability to capture both basic visual attributes and contextual details [11].

LLaVA A model oriented toward the generation of textual responses conditioned by visual information. It offers high expressive capacity in the generated descriptions, combining fine-grained visual understanding with rich and contextualized natural language [13].

First, a comparative analysis is presented to determine which model is most suitable for our task. Next, the procedure used to generate textual descriptions from the visual mosaics is described. Finally, a qualitative and quantitative evaluation of the obtained descriptions is provided.

4.2.1 Comparative Analysis of Description Models

This section evaluates which of the considered models provides more coherent, precise, and informative descriptions for the semantic interpretation of neuronal activations.

To ensure a fair comparison, the three analyzed models were applied to the same visual mosaics, generating descriptions for each from the same visual input.

Figure 9 shows a representative example of this comparison. The selected neuron exhibits recurrent activation in scenes where people are holding a fish. The diagram displays the descriptions generated by each model, allowing for direct qualitative comparison in terms of semantic richness, contextual accuracy, and linguistic expressiveness.

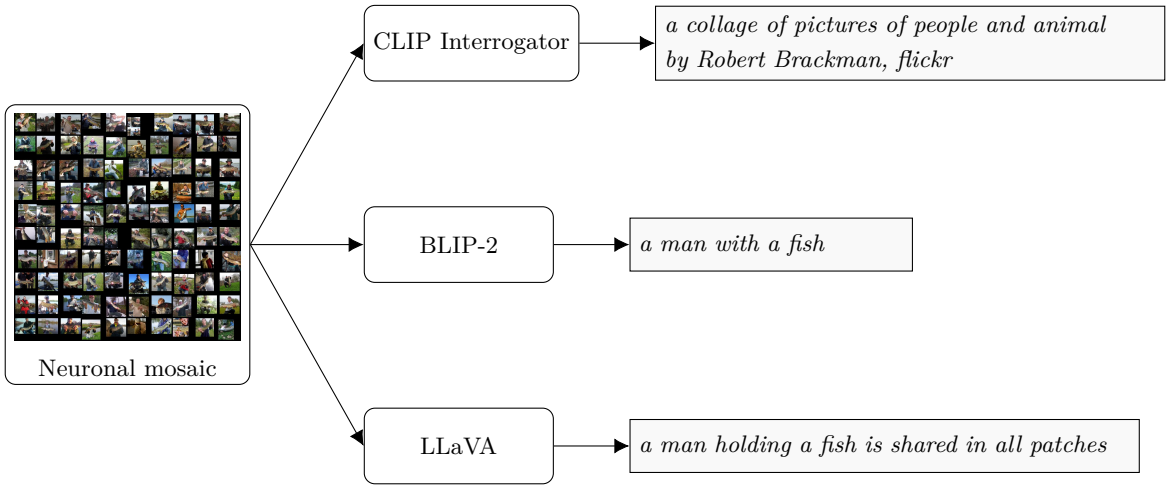


Figure 9: Comparison of descriptions generated by different VLMs from the same neuronal mosaic.

This comparison highlights relevant qualitative differences between the models.

CLIP Interrogator does not generate descriptions autonomously but instead selects the most similar phrase from a large set of pre-existing candidates, using CLIP embeddings and a cosine similarity measure. This process usually leads to *concise and generic labels* that rarely capture the specific visual semantics of neuronal mosaics.

BLIP-2, in contrast, does generate text autonomously, but its outputs tend to be short and often too general or fragmentary. The model fails to capture the visual complexity of mosaics or establish relationships among the different fragments.

On the other hand, **LLaVA** produces richer, more complete, and contextualized descriptions. It can combine information about predominant objects and shared actions among fragments, generating responses that faithfully reflect the common visual patterns of the neuron. This expressive capacity makes it the most suitable model for the semantic interpretation of neurons activated by multiple related visual contexts.

Justification for the Choice of LLaVA In addition to the qualitative results presented above, where LLaVA has demonstrated the ability to generate more complete, precise, and contextually relevant descriptions than BLIP-2 and CLIP Interrogator, this decision is further supported by the results of the *benchmark* **LLaVA-Bench** [14, 12]. This benchmark evaluates the model's performance across three key dimensions for our objective:

- **Visual conversation:** 83.1% of GPT-4 performance
- **Detailed description:** 75.3% of GPT-4 global performance
- **Complex reasoning:** 96.5% of GPT-4 global performance
- **Total aggregate score:** 85.1% of GPT-4 global performance

These scores position LLaVA as the open-source multimodal model closest to GPT-4's behavior, making it particularly well-suited for tasks requiring the interpretation of complex visual information and the generation of coherent and informative linguistic responses.

For all these reasons, LLaVA was selected for the generation of all neuronal descriptions in this study.

4.2.2 Adaptive Method for Constructing Neuronal Descriptors

To ensure optimal exploitation of LLaVA's capabilities in the context of neuronal semantic analysis, this work developed a method based on *multimodal prompt engineering*, specifically designed to generate rigorous, coherent, and interpretable textual descriptors that explain the semantic content activated by the internal neurons of a CNN. This systematic algorithmic strategy provides a replicable, adaptive, and scalable methodology for interpreting neuronal concepts in any visual domain.

Multimodal prompt engineering consists of optimizing the textual instructions provided to models capable of combining visual and textual information, guiding the responses generated by such systems. This technique differs from traditional *prompt engineering*, which only deals with pure language models without the need to integrate visual references.

Several recent studies [6, 32] have defined recommended practices in this area, such as explicitly specifying the expected conceptual and visual level to avoid ambiguities and model hallucinations. Following these recommendations, this work proposes a carefully designed prompting strategy to minimize semantic errors and automatically adjust semantic abstraction depending on the analyzed layer. The proposed method integrates three fundamental elements as follows:

1. The explicit formulation of *prompts* for vision-language models.
2. The application of validated best practices to reduce semantic hallucinations and ensure coherent responses [6, 32].
3. Hierarchical adaptation according to the conceptual level of each neural network layer (*layer-aware prompting*).

Based on the above ideas, we propose an algorithm summarized in Figure 10, in which for a given model, two essential elements must be considered: (a) the context in which the model's task is applied, essentially

defined by the training dataset; (b) the layers that define the model's architecture and form the hierarchy of elements that can be explained.

Algorithm for the Generation of Textual Descriptors

Multimodal Prompt Engineering

For each *DATASET* do *(Context-aware prompting)*
For each *LAYER* do *(Layer-aware prompting)*

a) **Assignment of specialized role (L)**
Ex: Computer Vision Researcher

b) **Determination of elements to explain / not explain:**
   (a) Visual elements to explain
   Ex: textures, shapes, objects...
   (b) Visual elements to avoid
   Ex: speculative or ambiguous elements...

c) **Delimitation of output in format, length, and vocabulary**
Ex: visual concepts, semantic categories

Figure 10: Algorithm for generating textual descriptions within the framework of multimodal prompting.

Next, we detail the essential components of this algorithm, explaining how it adapts to the hierarchical characteristics of the neural network and to the context of the analyzed visual domain.

Functional Structure and Hierarchical Adaptation of the Method

The proposed algorithm is based on a modular prompting architecture designed to guarantee textual descriptions that are accurate, interpretable, and consistent with the principles of semantic coherence and hierarchical adaptation. This system integrates three main components:

- **Assignment of a specialized role:** Each *prompt* incorporates an explicit identity for the model (e.g., "you are an expert in computer vision"), which guides the tone and register of the responses, favoring technical precision and linguistic formality.
- **Explicit conditional instructions:** Clear specification of which visual elements should be described (shapes, patterns, textures, objects) and which should be avoided (ambiguous, speculative, or contextually uninformative elements), in order to reduce errors and semantic hallucinations.
- **Formal control of the output:** The structure and content of the response are delimited, specifying admissible vocabulary and expected semantic categories. This ensures homogeneous linguistic production and facilitates comparison across neurons.

This structure is complemented by an **adaptive hierarchical prompting strategy** (*layer-aware prompting*), which adjusts the semantic abstraction level of the description according to the depth of the layer where the analyzed neuron is located:

- **Initial layers (ReLU):** responses are strictly limited to elementary visual attributes such as lines, orientations, or colors.
  Example prompt: "Focus strictly on low-level visual features... Do not mention any objects."
- **Intermediate layers (Layer 1–2):** favor the description of abstract patterns, repeated textures, and recurring shapes, avoiding references to complete objects.

Example prompt: "Describe the visual concept this neuron appears to respond to: shared textures, repeated shapes..."

- **Deep layers (Layer 3–4):** allow descriptions with higher semantic content, including identifiable objects, actions, or scenes.
  Example prompt: "Briefly describe what consistently appears across these patches... including objects, activities, or semantic categories."

In addition, the system incorporates a layer of contextual adaptation (*context-aware prompting*) that allows the method to be applied to any visual domain. The user may concisely specify the type of data used (for example, medical images or urban scenes), as well as relevant keywords or categories. This information allows dynamically adjusting the instructions according to the analyzed domain.

For instance, in a medical environment, terms such as *"tumor"*, *"healthy tissue"*, or *"blood vessels"* may be used; whereas in autonomous driving applications, words such as *"pedestrian"*, *"vehicle"*, or *"traffic sign"* would be more appropriate. This adaptability guarantees the generalization and transferability of the proposed method.

4.2.3 Evaluation

Once neuronal descriptions were generated, a qualitative evaluation was carried out to assess their coherence and fidelity with respect to the visual patterns present in the mosaics. The objective is to verify whether LLaVA adequately responds to the provided instructions and whether the resulting descriptions consistently reflect the expected abstraction level for each network layer.

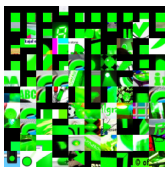
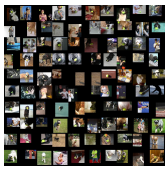
Despite the use of adapted prompting, generating accurate descriptions still presents significant challenges: the model may hallucinate, infer nonexistent contexts, or assign imprecise semantic labels. Moreover, since LLaVA does not have access to the internal information of the CNN, the responses are based solely on the visual stimulus (the mosaic).

For this reason, manual inspection was chosen to identify common errors per layer and subsequently apply a systematic *re-prompting* process. This involves reformulating the instructions to minimize identified ambiguities and improve response quality.

Automatic validation of descriptions using a second model (either another VLM or a text classifier) was discarded, since such systems may introduce their own biases, reinforce shared errors, or validate assumptions not actually supported by the mosaic. Thus, manual inspection remains the most reliable tool to guarantee semantic coherence between the description and the visual stimulus.

Re-prompting was applied to all analyzed layers (ReLU, Layer1–4), reviewing representative samples of highly activated neurons. Once the *prompts* were adjusted, all descriptions used in subsequent stages of the method (conversion to embeddings, clustering, and visualization) were regenerated, ensuring that the later analysis is based on refined and coherent descriptions.

Table 1: Examples of semantic improvement after *re-prompting* in neurons from different layers of ResNet-50.

Mosaic	Layer	Initial Description	Adjusted Prompt	Refined Description
	**ReLU**	Black and white colors.	Focus on low-level visual features: colors, patterns, textures, lines. Do not mention objects or scenes.	Black and white diagonal stripes.
	**Layer 1**	A collage of green objects, including a clock, a remote, and a cell phone.	Focus only on low-level visual features such as colors, contrasts, textures, shapes, and geometric patterns. Do not describe objects or scenes.	Green and white patches with repeating geometric patterns and sharp color contrasts.
	**Layer 4**	This neuron is selective to dogs.	Directly describe recognizable objects, actions, or semantic categories that appear consistently across the patches. Do not mention layout.	This neuron is selective to dogs playing with a tennis ball.

The final *prompts* used for regenerating descriptions, as well as the technical details of the LLaVA model employed in the process, are provided in **Appendix** C. Once this phase was completed, the generated descriptions were converted into vector representations to enable a systematic quantitative analysis of the semantic behavior of neurons.

4.3 Construction of Embeddings

To enable a quantitative analysis of the semantic content captured by each neuron, the descriptions generated by LLaVA were converted into vector representations (*embeddings*). This transformation is essential, since computational systems cannot operate directly on natural language texts. *Embeddings* encode latent meaning into a numerical space, allowing the application of techniques such as comparison, clustering, and visualization.

In this work, two complementary strategies were explored to construct these representations, with the aim of capturing both the linguistic and visual information associated with each neuron:

- **Pure textual embeddings** $\vec{t_i}$, generated with the model `sentence-transformers/clip-ViT-L-14` [24], which transforms each description d_i into a vector $\vec{t_i} \in \mathbb{R}^{768}$. This model was selected due to its alignment with CLIP's multimodal latent space, making it particularly suitable for representing texts with implicit visual content.
- **Fused multimodal embeddings** $\vec{f_i}$, obtained as the average between the textual embedding $\vec{t_i}$ and the visual embedding $\vec{v_i}$ of the corresponding neuronal mosaic m_i, using the model `openai/clip-vit-base-patch32` [22]:

$$\vec{f_i} = \frac{\vec{t_i} + \vec{v_i}}{2}$$

This approach allows comparing two ways of encapsulating the meaning of each neuron: one based exclusively on the generated text, and another that also integrates direct visual information. The objective is to determine which representation yields more coherent, interpretable, and useful semantic clusters for

subsequent analyses.

The results show that multimodal *embeddings* provide higher internal cohesion and better *Silhouette Score* values, which is why they were adopted as the basis for the subsequent analyses.

4.3.1 Analysis of the Proposed *Embeddings*

The goal of this section is to compare the quality of the different types of *embeddings* used to represent neuronal descriptions. To this end, a validation strategy is proposed based on clustering techniques and quantitative measures of cohesion and separation between clusters.

The starting hypothesis is that, if the *embeddings* adequately capture the semantic meaning of the descriptions, they should yield coherent clusters: neurons within the same cluster should share similar visual concepts. Conversely, a poor or noisy representation would produce ill-defined and hard-to-interpret clusters. Thus, cluster quality serves as an indirect indicator of both the vector representation and the original textual descriptions.

To analyze the semantic quality of the embeddings generated in each layer, the **KMeans** algorithm[7] was applied to obtain an interpretable partition into clusters. This strategy allows systematically comparing the two embedding variants —pure textual and multimodal— based on cohesion and separation criteria. The clustering process was applied identically in both cases, selecting the optimal number of clusters by maximizing the *Silhouette Score* and internal compactness.

To objectively assess cluster quality, three complementary metrics were used, described below.

Intra-cluster cohesion. Measures internal similarity within each cluster, by calculating the average cosine similarity between all pairs of embeddings within the same group. Mathematically:

$$\text{Cohesion}(C_k) = \frac{2}{n_k(n_k-1)} \sum_{1 \le i < j \le n_k} \cos(\vec{x}_i, \vec{x}_j)$$

Higher cohesion values indicate more compact and semantically coherent clusters.

Silhouette Score. Evaluates clustering quality by simultaneously considering internal cohesion and separation from other clusters. For each embedding i:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

where $a(i)$ is the average distance to other points within the same cluster, and $b(i)$ is the average distance to the nearest cluster. The global average of $s(i)$ provides an aggregate measure to compare cluster configurations. Higher values indicate well-separated, well-defined groups.

Adjusted Rand Index (ARI). Unlike the two previous metrics, ARI directly compares two partitions of the same set —in this case, the clusters obtained with textual and multimodal embeddings. It measures agreement while correcting for chance:

$$\text{ARI} = \frac{\sum_{ij} \binom{n_{ij}}{2} - \mathbb{E}}{\frac{1}{2}\left[\sum_i \binom{a_i}{2} + \sum_j \binom{b_j}{2}\right] - \mathbb{E}}$$

[7] An unsupervised clustering algorithm that assigns points to k clusters by iteratively minimizing Euclidean distance to centroids.

where $\mathbb{E}$ represents the expected number of agreements under a random distribution. Values close to 1 indicate strong agreement between partitions, while values close to 0 indicate unrelated groupings.

This set of metrics provides a solid basis for comparing the two embedding strategies. The first two allow analyzing the internal quality of each representation, while ARI facilitates comparison between them, offering a comprehensive view of the coherence, separability, and semantic consistency of neuronal clusters.

For each analyzed layer, **Table 2** reports the values obtained for the different metrics, comparing **pure textual embeddings** with **multimodal embeddings** (text + image).

Table 2: Comparison of metrics between textual and multimodal embeddings per layer.

Layer	Embedding	# Clusters	Cohesion	Silhouette	ARI
`Layer1`	Text	109	0.972	0.338	0.587
`Layer1`	Multimodal	74	0.971	0.382	
`Layer2`	Text	105	0.881	0.221	0.415
`Layer2`	Multimodal	149	0.950	0.240	
`Layer3`	Text	313	0.894	0.237	0.585
`Layer3`	Multimodal	283	0.927	0.259	
`Layer4`	Text	964	0.909	0.163	0.410
`Layer4`	Multimodal	735	0.934	0.172	
`ReLU`	Text	6	0.772	0.338	0.540
`ReLU`	Multimodal	4	0.802	0.315	

The results show a **consistent trend** across all layers: multimodal embeddings (combining textual and visual information) tend to achieve slightly higher **intra-cluster cohesion** values and better **Silhouette Scores**, indicating more compact and well-separated semantic groups. This advantage is especially notable in `Layer2`, `Layer3`, and `Layer4`, where cohesion clearly improves compared to pure textual embeddings.

In light of these results, the remainder of the semantic analyses in this work are based on multimodal embeddings, as they maximize **internal coherence** and **interpretive capacity** of neuronal clusters.

To facilitate the visual interpretation of the latent structure obtained, **Appendix D** includes additional visualizations such as Silhouette Score curves and UMAP projections for each layer.

With this set of algorithms, metrics, and validations, a robust framework is available for the semantic analysis of the internal activations of a CNN. The following section presents the concrete application of this method to the study of neurons activated in image classification tasks.

5 Applications: Analysis and Debugging of Neural Networks

Once semantic representations were defined through *embeddings*, the project's main experimental phase began. The objective of this phase is to validate to what extent these representations can provide a deeper understanding of the internal functioning of a convolutional neural network.

Specifically, two complementary use cases are proposed: (1) semantic and hierarchical analysis of neuronal behavior across the network's layers, and (2) practical application of this knowledge for the detection and debugging of misclassifications.

These two scenarios constitute the empirical basis of the project and allow assessing the explanatory potential of the textual descriptors and their associated embeddings as interpretability tools.

5.1 Use Case 1: Semantic Analysis of Internal Neurons

The first use case focuses on characterizing the internal behavior of CNN neurons through semantic analysis of the descriptors generated by LLaVA. Using the corresponding embeddings and textual descriptors, several experiments were conducted to evaluate semantic coherence, conceptual diversity, and thematic organization of neuronal activations in each layer.

The analysis combines techniques from natural language processing (TF-IDF, cosine similarity, term-weight analysis) with complementary methods such as dimensionality reduction and clustering, which are used to facilitate visualization and structural comparison of neuronal representations. This integrated approach makes it possible to identify recurring conceptual patterns, trace semantic and hierarchical evolution throughout the network, and estimate the characteristic abstraction level of each layer.

To provide a systematic and structured view, this use case is divided into three main blocks:

1. **Evolution of concepts per layer**: analysis of how predominant concepts change at each level of the network, using lexical techniques and automatic semantic labeling.
2. **Semantic connection between layers and cluster traceability**: implementation of a system to analyze semantic evolution both at the global layer level and at the detailed cluster level. This system uses embeddings and similarity metrics (such as cosine similarity) and is visualized with flow diagrams such as Sankey plots.
3. **Semantic indexing and retrieval of neurons**: exploration of the possibility of retrieving relevant neurons based on natural language queries, as an interactive interpretability tool.

5.1.1 Evolution of Concepts Across Layers

To initiate the semantic analysis, the vocabulary of the neuronal descriptions was studied per layer, with the objective of identifying the most representative concepts and how these evolve throughout the network.

To ensure a rigorous quantitative analysis of the set of neuronal descriptions generated by LLaVA, the **TF-IDF** method (*Term Frequency - Inverse Document Frequency*) [16] was used, which weights the importance of each term by considering both its local frequency and its global distribution. This avoids very common words overshadowing genuinely informative concepts.

The process followed the steps below:

- Prior removal of *stopwords* and non-informative terms, to reduce semantic noise.
- Calculation of the **TF-IDF** weight for each term t within each document d, where a *document* is defined as the set of neuronal descriptions of a given neuron in a layer ℓ. The applied formula is:

$$\text{tfidf}(t, d) = \text{tf}(t, d) \cdot \log\left(\frac{|\mathcal{D}_\ell|}{1 + \text{df}(t)}\right)$$

  where $\mathcal{D}_\ell$ represents the set of documents corresponding to layer ℓ, $\text{tf}(t, d)$ is the frequency of term

t within document d, and $\mathrm{df}(t)$ is the number of documents in which term t appears.

- Retention of the **50 terms with the highest TF-IDF weight** for each layer, ordered in decreasing order of their weight.
- Generation of *word clouds* (*WordClouds*) based on the obtained TF-IDF weights, to facilitate a qualitative visualization of the most relevant terms.

This strategy provides both a quantitative representation of the relevance of each concept and a qualitative visualization that facilitates global interpretation.

The following results are presented for each layer:

- The most prominent terms and their TF-IDF weight (table).
- The corresponding ***WordCloud***.

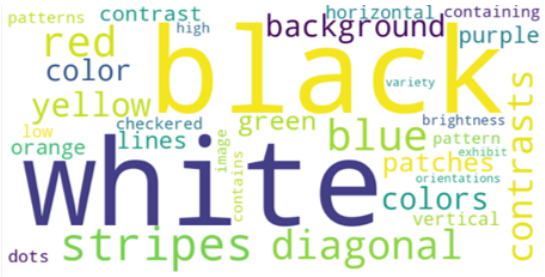
This analysis highlights how the network progressively builds and transforms its internal semantic representation as the hierarchy of layers unfolds.

ReLU Layer. In the initial `ReLU` layer, the descriptors show a strong presence of terms related to **colors**, **contrasts**, and **basic geometric patterns**. The calculation of TF-IDF weights and the corresponding WordCloud reveal that concepts such as basic colors and simple shapes — `white`, `black`, `stripes`, `blue`, and `red` — clearly dominate the descriptions.

This result is consistent with the expected behavior of the first layers in convolutional networks, which typically act as extractors of elementary visual features, sensitive to low-level patterns such as edges, color contrasts, lines, and simple textures.

Concept	Weight (TF-IDF)
white	8.06
black	7.81
stripes	6.06
blue	5.03
red	4.85

Top 5 predominant concepts.



WordCloud of the `ReLU` layer.

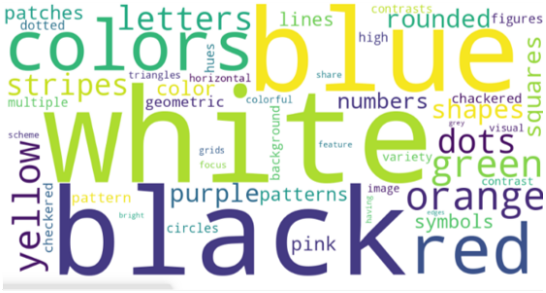
Figure 11: Semantic analysis for the `ReLU` layer: predominant concepts and associated WordCloud.

Layer1. In `Layer1`, the network continues to be dominated by concepts related to **colors** and **contrasts** (`white`, `black`, `blue`, `colors`, `red`), and the WordCloud shows an increase in the frequency of non-basic colors such as `orange`, `purple`, and `pink`. At the same time, sensitivity to **structured shapes** emerges (e.g., `triangle` or `circle`), along with **shape-related concepts** such as `letters`, `shapes`, `dots`, `geometric`, and `numbers`.

This behavior is consistent with the typical evolution of CNNs, where intermediate layers begin to construct more complex compositions from low-level features.

Concept	Weight (TF-IDF)
white	34.43
black	33.31
blue	31.44
colors	30.28
red	29.41

Top 5 predominant concepts.



WordCloud of the `Layer1`.

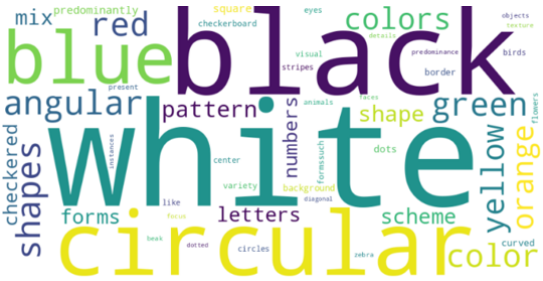
Figure 12: Semantic analysis for the `Layer1`: predominant concepts and associated WordCloud.

Layer2. In `Layer2`, descriptions still show a prominent presence of **colors** (`white`, `black`, `blue`), but **more structured geometric shapes** begin to appear significantly and with high frequencies, such as `circular` and `angular`.

This result is consistent with CNNs transitioning towards more complex representations in intermediate layers, where compositional visual patterns such as circles, angles, and combinations of textures are captured.

Concept	Weight (TF-IDF)
white	36.89
black	35.71
circular	33.07
blue	29.90
angular	25.16

Top 5 predominant concepts.



WordCloud of the `Layer2`.

Figure 13: Semantic analysis for the `Layer2`: predominant concepts and associated WordCloud.

Layer3. In `Layer3`, a major shift occurs in the nature of the activated concepts. The descriptors show a clear transition towards **higher-level concepts**, with a prominent presence of objects and more complex semantic categories. The predominant terms include `various`, `objects`, `dogs`, `people`, and `person`.

This pattern reflects the expected behavior of intermediate-to-upper layers in CNNs, which begin to integrate visual patterns into more semantic representations, linked to categories of objects, living beings, and scenes.

Concept	Weight (TF-IDF)
various	54.87
objects	41.66
dogs	32.53
people	19.29
person	19.25

Top 5 predominant concepts.

WordCloud of the `Layer3`.

Figure 14: Semantic analysis for the `Layer3`: predominant concepts and associated WordCloud.

Layer4. In the deepest analyzed layer, `Layer4`, a clear representation of **high-level semantic concepts** is consolidated. The predominant terms — `people`, `animals`, `activities`, `objects` — reflect that the neurons in this layer capture visual patterns strongly associated with **recognizable objects and scenes**, as well as complex entities involving context and action. This behavior is fully consistent with the role of deep layers in CNNs, aimed at extracting rich and contextualized semantic features.

Concept	Weight (TF-IDF)
people	230.14
various	116.57
objects	104.94
animals	92.28
activities	83.20

Top 5 predominant concepts.

WordCloud of the `Layer4`.

Figure 15: Semantic analysis for the `Layer4`.

These observations clearly reflect how lexical diversity and cohesion evolve across the network: the initial layers show a vocabulary dominated by basic and recurring visual descriptors, while the deeper layers reveal a progressive semantic specialization and greater conceptual diversity linked to complex objects and scenes.

5.1.2 Semantic Connection Between Layers and Cluster Traceability

After analyzing in detail the semantic content of each layer individually, a transversal analysis is now proposed to examine how the concepts captured by neurons evolve and transform as network depth increases.

This study is structured around two complementary approaches, allowing us to understand the hierarchical evolution of meaning within the CNN from two different perspectives:

- **Layer-wise analysis**: studies the global thematic evolution based on the neuronal descriptions generated by *LLaVA*. This analysis relies on the automatic classification of each description into predefined semantic categories, using a *zero-shot* model. The objective is to observe how predominant types of concepts change throughout the layers, from basic visual patterns to full objects and scenes.

- **Cluster-wise analysis**: based on the semantic groupings previously obtained through multimodal embeddings (Section 4.3.1, D). At each layer, neuron clusters are represented by their most representative keywords (obtained via *TF-IDF*), and semantic continuity between clusters of consecutive layers is explored using cosine similarity measures. This approach allows analyzing how concepts are preserved, diverge, or transform across hierarchical levels of the network.

This dual analysis provides a rich and complementary view of the semantic abstraction process within the CNN. On the one hand, it captures global trends in the distribution of conceptual themes; on the other, it offers fine-grained traceability of concepts at the level of coherent neuronal groups, reinforcing the interpretability of the network's internal behavior.

Global Thematic Analysis (Layer-wise) To categorize neuronal descriptions into meaningful conceptual groups, a set of semantic categories was previously defined based on an exhaustive exploratory analysis of the descriptors generated by LLaVA. This process combined qualitative reading of the descriptions, analysis of WordClouds per layer (Figures 11 to 15), and quantitative study of the most relevant terms obtained with TF-IDF. Based on this set of indicators, the following categories were established, covering the main conceptual axes observed:

- *Actions/Scenes*, *Animals*, *Colors*, *Full Objects*, *Numbers and Letters*, *Objects*, *Parts*, *People*, *Textures/Shapes*.

To perform consistent automatic labeling of these categories across all layers, a **zero-shot classification model**[8] based on `DeBERTa-v3-large-mnli-fever-anli-ling-wanli` [10] was used, which probabilistically assigns each neuronal description to the most semantically related group.

This automatic classification — applied consistently across all layers — provides an overview of the evolution of abstraction and the nature of internal representations. **Figure 16** shows how this thematic distribution evolves along the network hierarchy, revealing a progressive transition from basic visual patterns (textures, colors) to high-level semantic representations (objects, scenes, and actions). This trend is consistent with the expected behavior of convolutional networks, where deeper layers capture increasingly abstract and contextualized concepts.

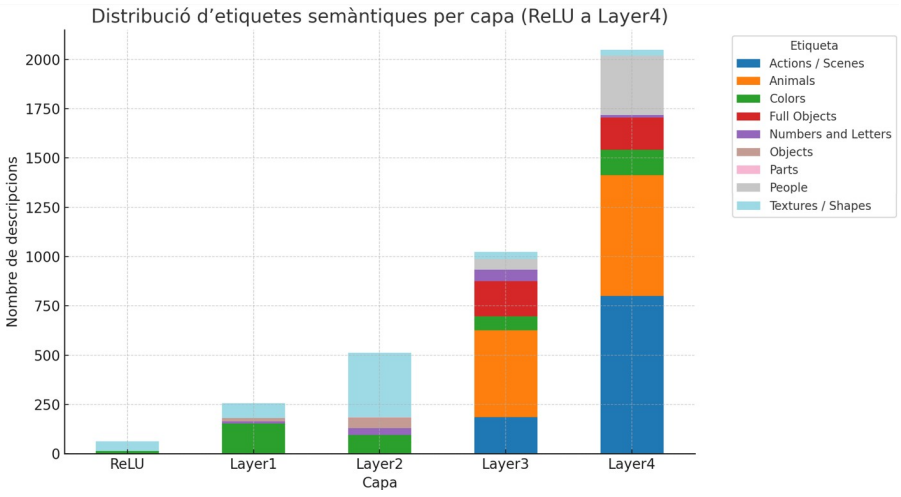


Figure 16: Distribution of semantic labels per layer (from `ReLU` to `Layer4`), obtained via zero-shot classification with `DeBERTa`.

[8] *Zero-shot* models can classify samples into categories unseen during training, relying solely on the semantic meaning of the labels.

The analysis of Figure 16 confirms and complements the previous observations from the lexical (TF-IDF) and visual (WordCloud) studies. As expected, in the first layers (`ReLU`, `Layer1`), low-level categories such as `Textures/Shapes` and `Colors` predominate, reflecting an initial encoding of elementary visual patterns.

From `Layer2`, thematic diversity begins to increase notably, with the progressive emergence of more structured categories such as `Numbers and Letters`, `Parts`, and `Full Objects`, consistent with the patterns observed in the corresponding WordClouds (Figures 13 and 14).

The sharpest conceptual leap occurs in `Layer3`, and especially in `Layer4`, where high-level categories (`Actions/Scenes`, `Objects`, `People`, `Animals`) clearly dominate the semantic representation. This result evidences how the deep layers of the CNN consolidate increasingly complex and contextualized semantic representations, a trend fully consistent with the expected behavior of convolutional networks.

Finally, it is worth noting that thematic distribution also shows an **increase in conceptual diversity** as one moves toward higher layers, a phenomenon already hinted at qualitatively in the WordClouds and here corroborated quantitatively and systematically. This pattern indicates that the network builds a rich and hierarchically organized semantic space, capable of representing a wide range of visual concepts at different levels of abstraction.

Semantic Continuity Analysis (*cluster-wise*) Whereas the previous analysis provided a global view of the dominant concepts per layer, this second approach focuses on the traceability between specific semantic groupings, allowing us to study how *clusters* evolve and transform throughout the network hierarchy.

The *clusters* used in this analysis were previously obtained from the clustering of neuron embeddings, as described in Section 4.3.1. At each layer, these *clusters* group neurons with similar semantic descriptions and provide the basis for constructing coherent semantic traceability between consecutive layers.

To visualize this semantic evolution, a *tracking* procedure based on cosine similarity between the vector representations of each cluster was implemented.

These representations were constructed by computing **TF-IDF** weights applied individually to each cluster (rather than the entire layer as in the global analysis), in order to identify the most representative keywords of each grouping. Subsequently, a *Sentence-Transformer* model (`clip-ViT-L-14`) was used to convert these keywords into a single vector embedding $\vec{e}_i$ for each cluster i. This strategy provides a cleaner representation centered on the conceptual core of each group of neurons, favoring semantic traceability across layers.

Once the embeddings were obtained, the **cosine similarity** between each cluster of layer L and all those in the following layer $L+1$ was calculated as:

$$\text{sim}(\vec{e}_i, \vec{e}_j) = \frac{\vec{e}_i \cdot \vec{e}_j}{\|\vec{e}_i\| \cdot \|\vec{e}_j\|}$$

Each cluster in L is associated with the cluster in $L+1$ with which it shows the highest similarity, thus building a semantic link between layers. The set of these links constitutes a **degree of semantic continuity** across layers.

This information is visualized by means of a **Sankey diagram**, where:

- Each node represents a concept cluster in a given layer.
- The links between nodes indicate semantic continuity between consecutive layers, weighted by cosine

similarity.

- A minimum similarity threshold (≥ 0.5) is applied to filter only robust semantic flows.

This procedure makes it possible to identify patterns of conceptual continuity (clusters that are consistently preserved or evolve coherently), bifurcations (a concept splitting into multiple sub-concepts), and the emergence of new concepts as information is transformed through the network architecture.

A representative example of the semantic flow of concepts between consecutive layers is presented below, visualized through a Sankey diagram (Figure 17). This visualization shows how certain dominant concepts evolve and transform progressively through the architecture of the network.

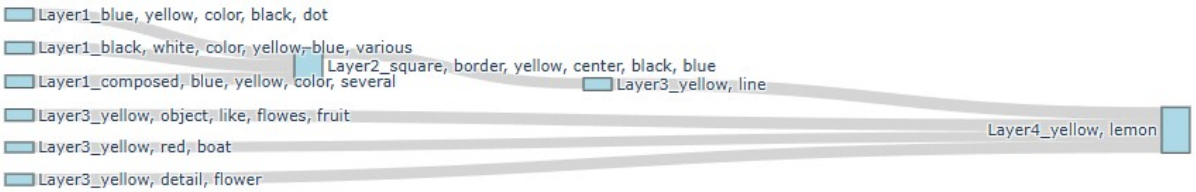


Figure 17: Example of semantic concept flow between consecutive layers, visualized through a Sankey diagram.

In the example shown (Figure 17), a clear semantic flow related to the concept `yellow` can be observed, which remains present across different layers, evolving from basic visual descriptors (`color, dot`, `border, yellow, center`) to more concrete and semantic representations in deeper layers, such as `object, like, flowers, fruit`, and finally consolidating into a specific object concept: `lemon` in `Layer4`.

This pattern clearly illustrates the ability of the network to progressively integrate and transform visual information, moving from low-level elements to high-level semantic representations associated with specific objects. At the same time, bifurcations and conceptual specializations can be identified as the flows progress, evidencing the richness and complexity of the CNN's internal semantic space.

Another illustrative example of semantic evolution is shown in **Figure 18**. In this case, the flow highlights how geometric patterns and basic visual features such as `circular`, `angular`, `shape`, `blue`, `white`, present in intermediate layers, progressively transform into a concrete object concept in `Layer4`: `blue, white, truck`.

This case clearly exemplifies the ability of the network to integrate information about shapes, colors, and visual structure in order to build high-level semantic representations. Moreover, a convergence of semantic trajectories from different clusters of earlier layers is observed, eventually consolidating into a coherent object category representation. This property reflects the hierarchical and compositional nature of the network's learning process.

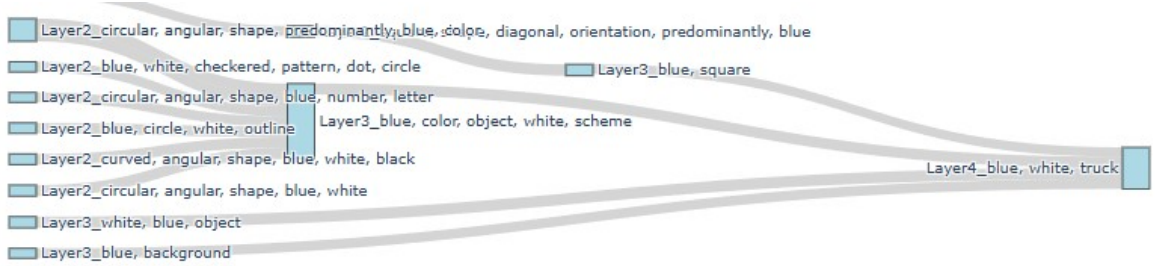


Figure 18: Additional example of semantic flow: transition from geometric and visual patterns to concrete object representations (`white truck`).

Detailed Cluster Tracking.
In addition to the global visualization, a detailed layer-by-layer analysis was performed to study how the concepts detected by each cluster evolve. Specifically, for each cluster C_i in layer L, the cluster C_j in the next layer $L+1$ with the highest semantic similarity (calculated on the embeddings of the most representative words according to TF-IDF) was identified.

This procedure establishes a direct *cluster-to-cluster* correspondence, which allows us to:

- Observe which concepts remain consistent across consecutive layers.
- Detect bifurcations or specializations of general concepts.
- Analyze possible semantic reorganizations between layers.

Table 3: Examples of concept transitions between consecutive layers.

Source layer	Cluster	Source concept	Target layer	Cluster	Target concept	Sim.
ReLU	0	red, black, patch, yellow, background, white	Layer1	48	red, yellow, color	0.77
ReLU	1	white, black, stripe, diagonal, contrast	Layer1	69	black, white, horizontal, stripe	0.89
Layer3	281	yellow, line	Layer4	568	yellow, lemon	0.85
Layer3	282	animal, people, wearing, glass	Layer4	103	people, animal, object, child, dog, car	0.76

The observed transitions illustrate both cases of **conceptual persistence** (e.g., color patterns consistently preserved across the network, such as *yellow* → *yellow, lemon*) and processes of **progressive semantic transformation** (e.g., simple visual patterns such as *stripe* or *shape* evolving into more complex object or scene representations). Phenomena of **bifurcation** are also detected, in which general concepts (such as *animal, people*) branch into more specialized categories in later layers.

This body of evidence complements the Sankey visualization, providing detailed traceability at the cluster level and revealing the internal mechanisms that drive semantic evolution within the network. It demonstrates that the CNN not only builds hierarchical representations but also exhibits dynamics of consolidation, differentiation, and conceptual emergence that are highly structured.

5.1.3 Semantic Indexing and Retrieval of Neurons

To complement the cross-layer analysis and provide interactive access to the network's internal conceptual map, a system of **reverse semantic retrieval** has been implemented, enabling the identification of the most relevant neurons from a query formulated in natural language.

This mechanism is based on the hypothesis that, if the neuron descriptions generated by LLaVA are semantically coherent, it should be possible to retrieve specific neurons from a free-text query describing visual concepts, scenes, or actions. This provides a direct tool to explore which concepts the network has learned and how they are distributed across its layers.

Figure 19 shows representative examples of semantic retrieval for three queries formulated in natural language: `"person face"`, `"a dog with a ball"`, and `"a person playing an instrument"`. For each of them, the two neurons with the highest cosine similarity to the query vector are displayed, along with

their visual mosaics, automatically generated descriptions, corresponding layer, and similarity score.

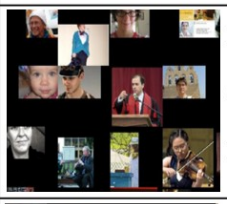
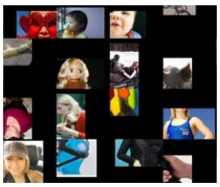
CONSULTA	NEURONA1	NEURONA2
"Person face"	 **Desc:** " A person's face" **Similitud** :0.950 **Layer** : 3	 **Desc:** " A woman's face" **Similitud** : 0.875 **Layer** : 3
"A dog with a ball"	 **Desc** : "A dog playing with a soccer ball" **Similitud** : 0.843 **Layer** : 4	 **Desc** : "a dog playing with a tennis ball." **Similitud** : 0.838 **Layer** : 4
"A person playing an instrument"	 **Desc**: "A person playing an harp" **Similitud** :0.902 **Layer** : 4	 **Desc**: "a man playing a violin **Similitud** :0.88 **Layer** : 4

Figure 19: Examples of semantic retrieval from natural language queries. The two most similar neurons for each query are shown, with their visual mosaic, description, layer, and cosine similarity score.

Validation and interpretation of results. The results show that the system is able to retrieve neurons with descriptions highly relevant and coherent with the query, even when the concept is composite or contains multiple attributes (such as objects, actions, and context). For example, the query `"a dog with a ball"` retrieves neurons with descriptions such as *"a dog playing with a soccer ball"* and *"a dog playing with a tennis ball"*, located in Layer 4, with high similarity scores.

This behavior demonstrates that the network not only captures basic visual attributes but also complex scenes and semantic relations, thereby reinforcing the quality and utility of the proposed description and indexing system.

This tool can be particularly useful for:

- Qualitatively verifying the semantic coherence of the generated descriptions.
- Exploring neurons specialized in specific concepts.
- Detecting biases or unexpected patterns within the latent space.
- Facilitating conceptual navigation and multimodal interpretation of neural behavior.

5.1.4 Conclusions of Use Case 1: Semantic Analysis of Internal Neurons

This first use case has enabled a rigorous analysis of the semantic behavior of the internal neurons of a CNN, showing how the architecture hierarchically builds conceptual representations of the visual world. Through the combination of thematic analysis, cross-layer analysis, and reverse neuron retrieval, the semantic evolution and organization of neuronal activations have been precisely characterized.

The results obtained show that: (1) the initial layers are dominated by low-level visual patterns (colors, textures, basic shapes), while the deeper layers progressively encode higher-level concepts (objects, scenes,

actions), validating the existence of a coherent semantic hierarchy; (2) cross-layer analysis through cluster tracking reveals a gradual and structured transformation of conceptual information throughout the network; and (3) reverse semantic retrieval facilitates interactive exploration of the latent space, providing an interpretable and agnostic mechanism for qualitative analysis.

The proposed pipeline is also generalizable to any modern CNN architecture, making it a practical tool for scalable semantic interpretation in both research and industrial applications.

5.2 Use Case 2: Contextual Debugging of Classification Errors

After having characterized in depth the global semantic behavior of the network (Use Case 1), this second use case explores how these same descriptions can be applied in the context of debugging specific errors.

Unlike the first scenario, oriented towards the structural and hierarchical analysis of internal representations, this study is situated in a more applied context of error diagnosis. It focuses on concrete examples in which the model fails, evaluating how individual neurons can contribute to explaining an incorrect prediction through their activations and the concepts they encode.

5.2.1 Semantic Debugging Methodology

To carry out this process of semantic debugging, a methodology has been designed that combines quantitative analysis of neural activations with qualitative analysis of their semantic content. These are the 4 steps of the proposed methodology:

- Calculation and comparison of **neural activations** between correctly and incorrectly classified images.
- Analysis of **mean activation differences** and **z-scores** to detect anomalous neurons in the context of the error.
- Qualitative analysis of the semantic content of these neurons using the **descriptions generated by LLaVA** (resulting from the Use Case 1 pipeline).
- Visualization and interpretation of the most involved neurons, both in terms of activation and semantic concept.

This approach allows the establishment of a **semantic debugging framework** based on interpretable descriptions, which complements traditional metrics such as Accuracy or Confidence. In domains where interpretability is essential (e.g., medical or legal), this type of analysis can provide insights to:

- Identify anomalous neurons activating irrelevant concepts.
- Evaluate the semantic bias of the model.
- Guide improvement strategies, such as targeted data augmentation or the regularization of certain patterns.

This study shows that the generated neuron descriptors can act not only as a tool for semantic analysis of internal representations but also as a practical instrument for contextual debugging and improvement of deep vision models. The following sections detail the stages of the methodology and the results obtained.

Activation Capture. As the first stage of the debugging process, a system for systematic capture of the model's internal activations has been implemented.

For each image i, the activations of the ResNet-50 are captured at each layer: `ReLU`, `layer1`, `layer2`, `layer3`, and `layer4`.

The activations are calculated per channel c, and projected into a vector per image through two complementary aggregations:

$$\max_{i,c} = \max_{h,w} A^{(i)}_{c,h,w} \qquad \text{mean}_{i,c} = \frac{1}{H \cdot W} \sum_{h,w} A^{(i)}_{c,h,w}$$

where $A^{(i)}$ represents the three-dimensional activation map, and H, W are its spatial dimensions.

This dual representation (**max** and **mean**) makes it possible to capture both the **peak maximum activation** per neuron (indicative of highly activating punctual patterns) and the **global activation tendency** (indicative of more diffuse or stable patterns).

Segmentation by Prediction and Class Selection. To prepare the debugging analysis, an initial quantitative analysis of the global behavior of the model on the validation set was carried out. For each image, the final prediction and the ground-truth class were recorded, constructing a **complete confusion matrix**. From this, the **accuracy per class** as well as the distribution of errors across classes were calculated.

This study made it possible to identify a set of **problematic classes** with a significantly unfavorable balance between correct and incorrect classifications. Table 4 summarizes the classes with the lowest accuracy, which constitute the starting point for selecting cases to be analyzed in depth.

Table 4: Classes with the highest frequency of misclassifications.

Class	Correct	Incorrect	Accuracy (%)
maillot	453	897	33.56
water_jug	573	777	42.44
tiger_cat	663	687	49.11
horned_viper	671	679	49.70
Eskimo_dog	690	660	51.11
velvet	721	391	64.84
sidewinder	881	469	65.26

However, many of these errors are not solely attributable to model deficiencies but also reflect **inherent dataset limitations**, such as labeling inconsistencies or hierarchical structures with semantic overlaps.

For example:

- In the case of `Eskimo dog`, a **semantic collapse** with `Siberian husky` has been observed, where many images of the parent class (more generic) visually depict individuals of the specific subclass.
- In classes such as `agaric` or `mushroom`, the internal hierarchy of the dataset causes ambiguities, as similar images may be assigned to both the generic class and more specific subclasses.

For this reason, class selection for debugging was not based solely on accuracy criteria but combined a quantitative analysis with a detailed qualitative inspection. The applied criteria were as follows:

- **Sufficient error volume**: priority was given to classes with a significant number of errors (>50) to ensure statistical robustness.
- **Systematic errors**: classes were selected where incorrect predictions were concentrated in a few specific classes, indicating consistent visual confusion patterns.
- **Dataset quality**: classes severely affected by labeling problems or semantic collapse were excluded.

- **Interpretability potential**: priority was given to cases where the confusion could be analyzed from a semantic and visual perspective, providing interpretative value.

Candidate Class Example: *palace*. As an illustrative example of the selection process, the class `palace` is presented, as it brings together several characteristics that make it potentially suitable for a semantic debugging study.

It is important to emphasize that the purpose of this section is not to establish `palace` as the exclusive focus of analysis, but rather to illustrate how the previously described selection criteria are applied. This same process can be replicated for any other class presenting relevant confusion patterns.

In the case of `palace`, a **sufficient volume of errors** (226) is observed, with an **accuracy of 79%**, and a distribution of errors strongly centered on other architecturally similar classes such as `castle`, `monastery`, or `vault`. This type of systematic confusion, based on clear visual similarities, is especially interesting for the semantic analysis of neural activations.

In contrast, sporadic errors involving unrealistic confusions or those without clear visual basis (e.g., an isolated confusion with `carousel`) are considered **anecdotal errors** and are not suitable for this type of study, as they do not allow consistent neuronal patterns to be inferred.

Table 5 shows the main classes incorrectly predicted as `palace`, evidencing the presence of a coherent confusion pattern that could be further explored in a detailed analysis.

Table 5: Main classes incorrectly predicted as *palace*.

Predicted class	**Error frequency**
Castle	63
Monastery	34
Bell cote	30
Vault	25
Dome	21
Mosque	17
Church	12
Carousel	1

This example highlights the importance of selecting classes with **systematic and interpretable errors**, avoiding anecdotal confusions that could introduce noise into the qualitative analysis.

Statistical Neuron Analysis. To quantify the differential behavior of each neuron in the presence of predictive errors, a statistical analysis has been implemented based on the systematic comparison of their mean activations in correctly classified images (**OK**) and misclassified images (**KO**). The goal is to identify neurons that show significantly different activation patterns depending on the success or failure of the prediction, as a potential indicator of their involvement in error generation.

For each neuron n, the difference between mean activations in the two sets is computed as:

$$\Delta\mu_n = \mu_n^{KO} - \mu_n^{OK}$$

where μ_n^{KO} is the mean activation of the neuron on misclassified images, and μ_n^{OK} is the mean activation

on correctly classified images. This value reflects the neuron's global tendency to either over-activate or deactivate in the presence of errors.

To interpret this difference in the context of the neuron's global variability, normalization by **z-score** is applied:

$$z_n = \frac{\Delta\mu_n}{\sigma_n}$$

where σ_n represents the combined standard deviation of the two sets. The value z_n provides a dimensionless measure of the significance of the activation change.

In this study, a neuron is considered to display significant differential behavior when $|z_n| \geq 2$, following the classical threshold associated with deviations greater than two standard deviations. Within this framework, values of $z_n \geq 2$ suggest that the neuron tends to abnormally activate in misclassified images, potentially acting as an error-inducing signal, while values of $z_n \leq -2$ indicate atypical inhibition in the presence of errors. Values within $|z_n| < 2$ are considered within the expected variability range, and these neurons are not selected for further qualitative analysis.

These metrics enable the construction of a **ranking of differential neurons** potentially involved in error generation and constitute the quantitative basis for the selection of neurons that will later be analyzed from a semantic and visual perspective.

Visualization and Interpretation. Once the neurons with significant differential activation have been identified, a set of visualizations has been automated to facilitate their qualitative interpretation. These visualizations combine quantitative, spatial, and semantic perspectives with the aim of revealing contextual patterns that may be inducing misclassifications.

For each anomalous neuron, the following analysis supports are generated:

- **Histograms of $\Delta\mu_n$**: represent the distribution of activation differences per layer, helping to identify which layers are particularly involved in systematic errors.
- **Z-score heatmaps**: visualize the intensity and relative localization of anomalous neurons within each layer, highlighting which channels present the most extreme statistical deviations.
- **Semantic descriptions**: generated from the visual mosaics associated with each neuron through a vision-language model, these descriptions provide a natural language interpretation of what each neuron "sees," revealing possible sources of semantic confusion (such as dominant backgrounds, partial objects, or visual patterns unrelated to the target class).
- **Top-activating images**: the images that most strongly activate each anomalous neuron are collected, enabling qualitative inspection of the visual stimuli that may be driving erroneous activation.

This toolkit complements the statistical analysis with a qualitative and interpretable view of neuronal behavior, strengthening the system's ability to diagnose confusion patterns in misclassification errors.

To illustrate the practical potential of this approach, a detailed case study focused on the recurrent confusion between the classes *palace* and *castle* is presented below.

5.2.2 Case Study: Confusion between Palaces and Castles

This case study analyzes the recurrent confusion between the classes *palace* and *castle*, two architectural typologies that, despite differing functionally, share visual traits that may induce misclassification. Palaces are ornamental residential buildings, associated with luxury, symbolic power, and monumental

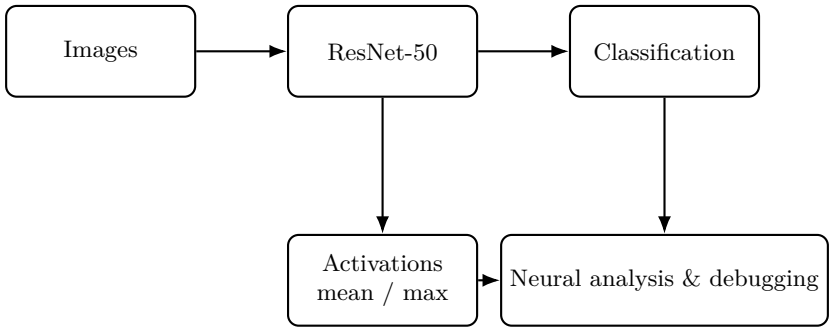


Figure 20: General scheme of the contextual debugging process applied to classification errors.

architecture; by contrast, castles have a defensive purpose and are characterized by walls, towers, fortified structures, and often large natural surroundings or green landscapes surrounding the construction.

The contextual debugging system is now applied to analyze this error in detail, identifying the neurons involved, their semantic descriptions, and the visual stimuli that may be driving the misclassification.

To contextualize the difficulty of the classification task, Figure 21 shows representative examples of each class within the ImageNet dataset. Despite the architectural differences, common elements can be observed that may induce confusion, particularly in scenes dominated by the surrounding landscape.

Palace *Palace* *Castle* *Castle*

Figure 21: Representative examples of *palace* and *castle* images from ImageNet.

Classification Performance Analysis To justify the detailed study of this confusion, the predictive behavior of the model with respect to the *palace* class has been analyzed. This type of analysis makes it possible to discern whether the confusion with *castle* is occasional or part of a systematic error pattern linked to shared visual features.

Although *palace* shows a relatively high accuracy (79.48%), a relevant proportion of errors occur with the *castle* class, an architectural typology with which it shares certain visual elements.

Moreover, *palace* ranks 153 out of 1000 in the global class ranking by accuracy, indicating that while it is not among the most difficult classes, its error rate is significant enough to warrant specific analysis.

Table 6 summarizes these observations.

Table 6: Classification statistics for the *palace* class.

Metric	**Value**
Total number of *palace* images	1350
Correctly classified as *palace*	1073
Accuracy (%)	79.48
Total errors	277
Errors classified as *castle*	63
Percentage of errors toward *castle*	22.74 %

Visual Examples of Correct and Incorrect Classifications. To complement the quantitative analysis, representative examples of correctly classified *palace* images, as well as misclassified ones predicted as *castle*, are shown. These samples provide insight into the visual contexts that may have influenced the neural activations detected as anomalous.

(a) *palace* → *palace* (b) *palace* → *palace* (c) *palace* → *castle* (d) *palace* → *castle*

Figure 22: Representative examples of correct and incorrect classifications for the *palace* class.

A preliminary visual inspection of *palace* and *castle* images reveals a potentially relevant pattern for predictive confusion. In particular, many of the images misclassified as *castle* correspond to palaces situated in open or natural environments, such as gardens, fields, or areas with water. At the same time, a manual review of ImageNet samples in the *castle* class often shows scenes with abundant vegetation and green landscapes, whereas *palace* images are more likely to depict isolated or architecturally centered buildings. This imbalance in visual context could lead the neural network to rely on the environment as a discriminative cue, thereby increasing confusion between the two classes.

To quantitatively test this hypothesis, the statistical analysis previously described has been applied to the *castle–palace* confusion, with the aim of identifying neurons exhibiting differential activation patterns depending on prediction success. The following presents the results obtained for the ima

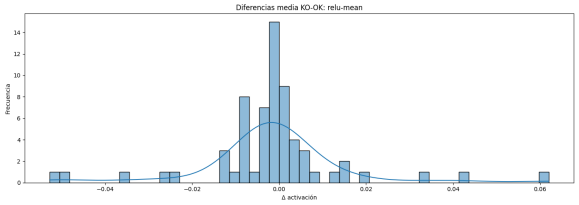
5.2.3 Layer-wise Analysis: Activation Distribution and Anomalous Neurons

To analyze the model's internal dynamics in misclassifications, the distribution of mean activation differences ($\Delta\mu_n$) and z-score values (z_n) is studied per layer. These metrics allow the detection of neurons with differential behavior between correctly classified (**OK**) and misclassified (**KO**) images.

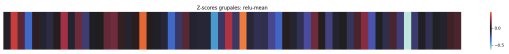
For each analyzed layer (`ReLU`, `layer2`, `layer4`), the following are generated:

- Histograms of $\Delta\mu_n$, reflecting the average bias of neuronal activation.
- Heatmaps of z_n values, enabling localization of neurons with notable statistical deviations.

These visualizations provide an overview of per-layer activity and facilitate the detection of neurons potentially involved in predictive errors.

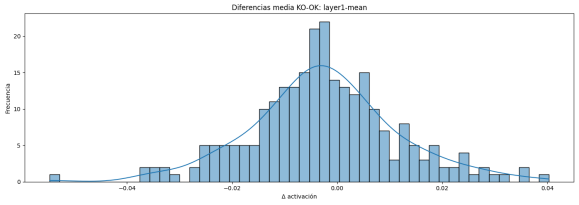


(a) Histogram of $\Delta\mu_n$ at `ReLU`.

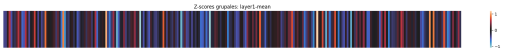


(b) Heatmap of z-scores at `ReLU`.

Figure 23: Statistical distribution of neurons for the `ReLU` layer.

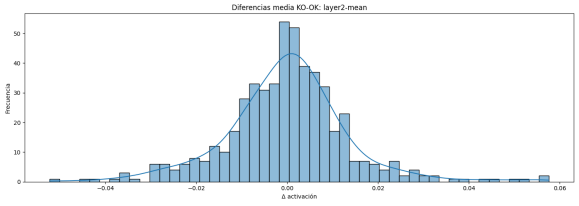


(a) Histogram of $\Delta\mu_n$ at `layer1`.

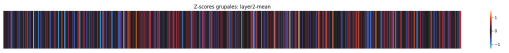


(b) Heatmap of z-scores at `layer1`.

Figure 24: Statistical distribution of neurons for the `layer1` layer.

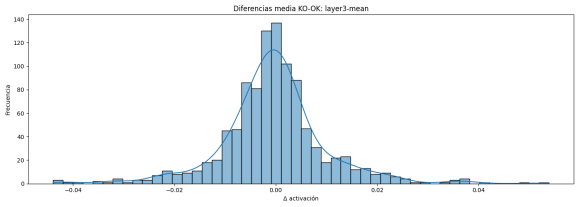


(a) Histogram of $\Delta\mu_n$ at `layer2`.

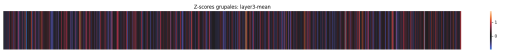


(b) Heatmap of z-scores at `layer2`.

Figure 25: Statistical distribution of neurons for the `layer2` layer.

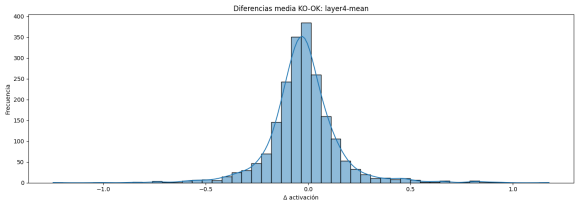


(a) Histogram of $\Delta\mu_n$ at `layer3`.

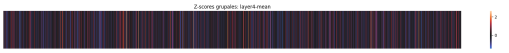


(b) Heatmap of z-scores at `layer3`.

Figure 26: Statistical distribution of neurons for the `layer3` layer.



(a) Histogram of $\Delta\mu_n$ at `layer4`.



(b) Heatmap of z-scores at `layer4`.

Figure 27: Statistical distribution of neurons for the `layer4` layer.

Quantitative Layer-wise Analysis. Based on mean activation differences ($\Delta\mu_n$) and normalized z-scores (z_n), the aggregated behavior of neurons in each network layer has been analyzed. These metrics make it possible to identify units with differential responses between correctly classified (**OK**) images and misclassified as *castle* (**KO**) images.

The results show that:

- In the initial layers (`ReLU` and `layer1`), the distributions of $\Delta\mu_n$ are centered around zero and show limited variation. Z-score values are generally low, although a few neurons exhibit slightly higher deviations.
- From `layer2` onwards, the dispersion of $\Delta\mu_n$ increases, and z-scores begin to highlight neurons with more pronounced behaviors. This could indicate the emergence of internal patterns differentiating OK and KO.
- The deeper layers (`layer3` and especially `layer4`) present a much wider distribution, with long tails and extreme values both in activation and inhibition. The z-score heatmaps show a higher density of neurons with elevated absolute values ($|z_n| > 2$), suggesting that high-level semantic encoding could play an important role in predictive confusion.

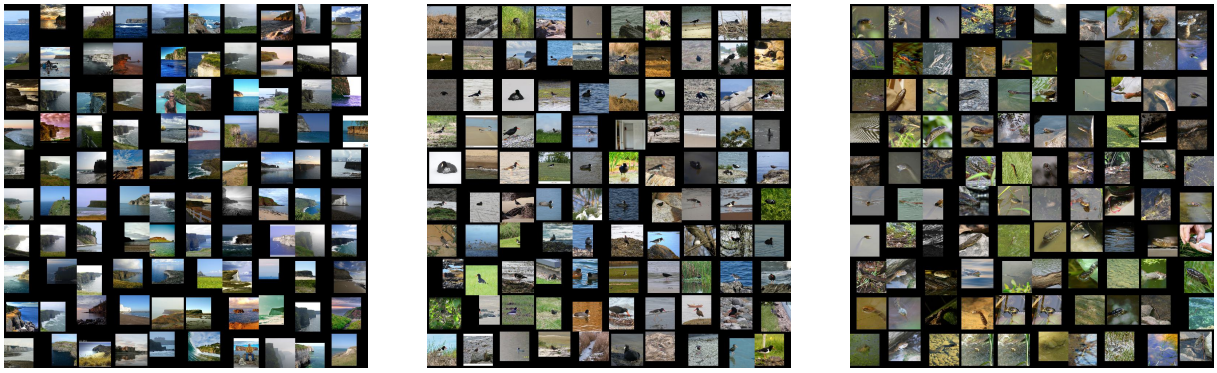
This statistical pattern provides an objective criterion to identify neurons with anomalous behaviors, i.e., with marked activation differences between correct and incorrect classifications. To qualitatively interpret these anomalies, a ranking has been generated of the neurons with the highest z-score values in the `layer4` layer, along with their semantic descriptions obtained from the visual stimuli that most strongly activate them.

Table 7 shows that many of these neurons are associated with landscape-related elements such as water, birds, or natural scenes. This reinforces the hypothesis presented in Figure 21, where several *palace* images misclassified as *castle* featured open or natural environments.

Table 7: Conflicting neurons with landscape-related semantics in the `layer4` layer.

#	**Neuron**	$\Delta\mu$	z	**Description**
1	1858	+2.209	+2.05	*"water and landscapes"*
2	1096	+0.936	+2.10	*"birds, water, and sand."*
3	744	+1.176	+2.30	*"a variety of animals, frogs and alligators, in their natural habitats."*

To complement the numerical analysis above, the following visual mosaics correspond to the neurons with the highest positive z-score deviations in the `layer4` layer, all of them related to landscapes, aquatic scenes, or natural environments. These images represent the visual stimuli that most strongly activate each of these neurons.



(a) Neuron 1858 (b) Neuron 1096 (c) Neuron 744

Figure 28: Visual mosaics (top-activations) of neurons related to landscapes.

Other Cases of Contextual Confusion. Beyond the *palace* → *castle* case, the contextual debugging system has also been applied to other classes that present systematic error patterns. Three particularly representative examples are described below:

- `power_drill` → `carpenter's_kit`: despite an *accuracy* of 77.99% (747 correct classifications out of 958), nearly 20% of the errors (43 images) occur in scenes where the drill appears embedded within toolkits. The neurons involved present descriptions such as *"suitcases and chairs"* or *"toolkits"*, indicating contextual dependency where the model associates the tool with its functional environment.
- `dining_table` → `restaurant`: with a high *accuracy* of 87.41% (1180 out of 1350), nearly 40% of errors (67 images) are directed toward `restaurant`. These confusions are frequent in scenes with food, tableware, or seated people, and the associated neurons show descriptions such as *"fruits and kitchen utensils"*, reinforcing the idea of a functional rather than object-focused reading.
- `microphone` → `stage`: with an overall *accuracy* of 75.87%, 326 errors are recorded, of which 73 (22.4%) correspond to `stage`. These images often include spotlights, stages, and the presence of performers. Highlighted neurons have descriptions such as *"chairs and decoration"* or *"people and objects"*, suggesting a scenic and contextual interpretation of the image.

These cases, shown in Figure 29, reinforce the validity and usefulness of contextual debugging as a tool to identify non-random predictive errors, especially in situations where the model relies excessively on the global context of the scene.

Figure 29: Summary of contextual confusions in three additional cases: `power_drill` → `carpenter's_kit`, `dining_table` → `restaurant`, and `microphone` → `stage`. The figure shows a target image, one of the predicted class, a misclassified image, the dominant neurons, and the highest z-score descriptions.

5.2.4 Conclusions of Use Case 2: Contextual Debugging

The detailed study of the *palace* → *castle* case reveals a non-random but systematic error pattern, strongly conditioned by specific visual features. Through statistical metrics and semantic analysis, several neurons in the `layer4` were identified with significantly higher activation in misclassified images, particularly associated with landscape environments, aquatic scenes, and the presence of birds or people in natural contexts.

These neurons appear to act as "confusing signals" that, under certain stimuli, divert the prediction toward the *castle* class. The error can be interpreted as an over-association of the *castle* concept with broad natural contexts, rather than focusing on distinctive architectural attributes. This suggests a semantic bias in the model's representation of historical buildings.

From this analysis, several practical proposals arise to improve the robustness of the model and reduce this type of error:

- **Focused fine-tuning** with hard examples that contrast *castle* and *palace* in different contexts, forcing the network to base its discrimination on more robust architectural attributes.
- **Use of visual attention** or techniques such as *Grad-CAM* to reinforce the model's focus on architectural regions and reduce its excessive reliance on global visual context.
- **Dataset enrichment** with samples where both classes share environments but are clearly distinguished by structure, reducing the spurious correlation between context and label.

This case highlights the potential of contextual debugging as a tool to identify systematic error patterns, analyze their semantic causes, and design concrete strategies for their mitigation. It also illustrates how the analysis of neuronal descriptions can provide valuable interpretive insights that complement traditional debugging methods.

6 Limitations

Despite the promising results obtained, the methodology presented in this work has several limitations that must be considered:

Dependence on VLMs. The proposed labeling system heavily depends on the quality and reliability of the vision-language models (VLMs) used. Despite recent advances, these models still show shortcomings in their ability to capture subtle shared concepts or abstract visual patterns. They often prioritize explicit objects present in the image rather than the common visual concept that activates the neuron. Thus, in cases where multiple objects share a texture, color, or structure, the VLM may generate inconsistent or biased descriptions. This limitation is exacerbated by the nature of their training, strongly conditioned by text associated with specific images, which may not accurately reflect higher-level visual correlations.

Evaluation of semantic quality. At present, we lack a robust mechanism to qualitatively assess whether the VLM-generated descriptions truly reflect the semantic content of the associated mosaic. One possible approach would be to use a second model to validate the coherence between image and description, but this introduces new risks: possible transfer of biases between models, circularity in validation, and the absence of an objective standard to define *semantic coherence*. This remains an open problem and an important challenge for VLM-based interpretation.

Dataset limitations. The ImageNet Fused dataset, while rich and diverse, contains overlapping classes and labels that are not always semantically disjoint. This can complicate error analysis, since in some cases the confusions are conceptually reasonable or even expected, making it difficult to define an *error* from a strictly semantic perspective.

Scalability and integration. The current pipeline operates in offline mode and with high computational cost, especially in the massive generation of descriptions for all neurons of a deep network. Scalability to larger architectures or production environments has not yet been validated. Moreover, the methodology does not integrate natively into the CNN prediction process, limiting its use in interactive or real-time applications.

Generalization. The experiments in this work were carried out on a specific architecture (ResNet-50) and dataset (ImageNet Fused). Although the results are consistent within this framework, generalization to other architectures (e.g., visual Transformers) or other domains (such as medical imaging or video) has not yet been explored.

7 Conclusions

This work has presented an innovative method for semantic labeling of internal neurons in convolutional networks using vision-language models (VLMs). By generating visual mosaics for neurons and applying adapted prompting strategies, open-text descriptions were obtained that expressively and precisely characterize the semantic behavior of each unit.

A complete method has been proposed that combines visual processing techniques, language models, and statistical analysis to explore and quantify the internal conceptual structure of a ResNet-50 trained on ImageNet Fused. By representing descriptions in semantic and multimodal embedding spaces, an exhaustive analysis of thematic diversity, hierarchical concept evolution, and semantic coherence across layers has been carried out.

Furthermore, the practical usefulness of this approach has been demonstrated in contextual debugging scenarios, allowing the identification of anomalous neurons associated with classification errors and the

analysis of their conceptual confusion sources. This application provides a powerful interpretive tool for diagnosing complex models.

The contributions made open new possibilities for neural network interpretability, offering a flexible, extensible, and language-based framework to explore the internal learning patterns of CNNs. Likewise, the system developed can be integrated into future research lines oriented toward model improvement in critical areas such as medical imaging, security, or autonomous driving.

The identified limitations and proposed future directions indicate great potential to further evolve this methodology, both technically and practically. The rapid progress of VLMs and interpretability techniques suggests that such approaches will become increasingly relevant in the design and analysis of transparent and reliable AI systems.

8 Future Work

This work opens several future lines of research and improvement, both methodological and practical:

Application to real problems. A natural direction is to implement the improvements identified through semantic debugging in concrete classification problems. Integrating this type of analysis into real pipelines could help improve the accuracy and reliability of CNN models, particularly in environments where interpretability is critical (e.g., medical, industrial, or security sectors).

Adaptation to specialized domains. The proposed approach can be extended beyond state-of-the-art general interpretability. Applying it to specialized areas such as medical imaging or pathology detection (e.g., cancer diagnosis) would constitute a highly relevant line of research. In this context, it would be essential to develop more rigorous and systematic prompting strategies, as well as independently validate the accuracy of the generated descriptions.

Constant model updates. The rapid evolution of vision-language models implies the need to regularly update the architectures used in the pipeline. Next-generation VLMs could provide substantial improvements in description quality and reliability, motivating periodic system revisions.

Advanced automatic evaluation. It would be interesting to explore new methods to automatically evaluate the quality of generated descriptions. For example, by comparison with predefined expert descriptors, or through visual reasoning systems capable of validating semantic correspondence between descriptions and activating visual patterns.

Generalization and robustness. Finally, an important future line is to study the generalization of the pipeline to other architectures (ViTs, Swin Transformers, deep CNNs for non-classification tasks), as well as validate the robustness of the proposed techniques on new datasets and in more complex or less structured scenarios.

Appendix

A Description of the ImageNet Fused dataset

To analyze the internal behavior of convolutional networks at the neuronal level, it is essential to have a dataset that combines high visual quality, semantic diversity, and reliable labeling. In this project, **ImageNet Fused** was used, a curated and standardized version of the well-known **ImageNet ILSVRC-2012** [7], widely recognized as a benchmark for training and evaluating deep vision architectures such as ResNet [9].

ImageNet contains more than 1.2 million training images with about 1,000 visual classes. Each class includes between 732 and 1,300 images, covering a wide semantic variety: from animals (e.g., "panda," "king penguin") and natural objects ("volcano," "sunflower") to artificial objects ("typewriter," "electric fan").

Unlike other datasets used in interpretability studies, such as the **Broden Dataset** employed in *Network Dissection* [2], which limits its annotations to a closed set of concepts such as object parts, textures, or scenes, ImageNet Fused retains the diversity and conceptual openness of its predecessor. This allows neurons to exhibit richer responses, less constrained by artificial labels.

Moreover, ImageNet is organized according to the lexical hierarchy of **WordNet** [18], a semantic database that groups words into synonym sets called *synsets*. This structure provides a precise definition of each category and facilitates hierarchical analysis of neuronal behavior (e.g., "animal" → "mammal" → "cat").

However, the original ImageNet version presents practical drawbacks for semantic neuron studies:

- Some images have inconsistent resolutions and proportions.
- Presence of visual noise and ambiguous labeling.
- Highly heterogeneous context conditions (background, orientation, scale).

For this reason, **ImageNet Fused** is used, a refined version that:

- Resizes and centers images uniformly.
- Filters out images with uninformative backgrounds or problematic composition.
- Maintains compatibility with architectures pretrained on ImageNet.

This optimized dataset provides an ideal setting to analyze neuronal activations robustly and to construct open semantic descriptions with vision-language models.

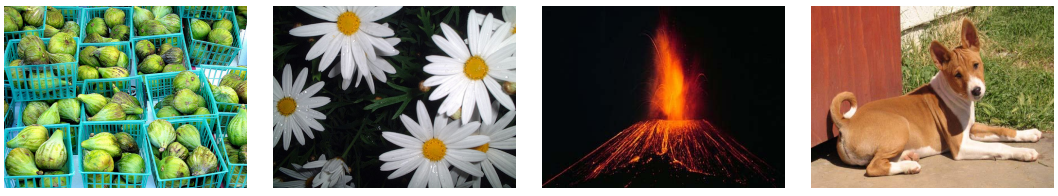


Fig Daisy Volcano Basenji

Figure 30: Examples from ImageNet Fused showing semantic diversity: fruit, flowers, natural scenes, and domestic animals.

B ResNet-50 Architecture

The *ResNet-50* (*50-layer Deep Residual Network*) [9] is a foundational architecture in computer vision, widely used for large-scale image classification tasks. It was introduced by He et al. to address the problem of *degradation* in deep networks, through the use of **residual connections** that allow direct information flow and prevent gradient vanishing.

General Structure

The ResNet-50 follows a modular and hierarchical architecture, composed of the following stages:

- An initial convolutional layer (`conv1`) with a 7×7 filter, followed by Batch Normalization and `ReLU` activation, producing **64 output channels**.
- A max pooling block that reduces the spatial dimensions of the image.
- Four main stages called `layer1` to `layer4`, each composed of several **residual blocks** with increasing channel depth:
  - `layer1`: 3 residual blocks – **256 neurons**
  - `layer2`: 4 residual blocks – **512 neurons**
  - `layer3`: 6 residual blocks – **1024 neurons**
  - `layer4`: 3 residual blocks – **2048 neurons**
- A global average pooling that aggregates spatial information from each feature map.
- A final fully connected layer, which transforms the representations into class predictions (1000 for ImageNet).

Residual Blocks

The basic element of this architecture is the **residual block**, which facilitates the learning of identity functions via *skip* connections:

- Each block consists of three convolutions with 1×1, 3×3, and 1×1 filters, followed by Batch Normalization and `ReLU` activation.
- The **residual connection** adds the original block input to the convolution outputs, enabling undistorted information flow and improving optimization.

This design allows the network to learn modifications to an identity function rather than absolute transformations, improving convergence and avoiding performance degradation in very deep architectures.

Representation Flow

ResNet-50 progressively builds semantic representations across its layers:

- In the initial layers (ReLU, `layer1`): low-level features such as colors, textures, and simple edges are captured.
- In intermediate layers (`layer2`, `layer3`): object parts, complex patterns, and abstract forms are detected.
- In the deeper layers (`layer4`): high-level semantic representations corresponding to whole objects or recognizable scenes are constructed.

This hierarchical process reflects a coherent functional and semantic organization, fundamental to the success of convolutional models in computer vision.

Key Properties

- 50-layer architecture with great depth, but trainable thanks to residual connections.
- Scalable and reusable structure, which has inspired multiple variants such as ResNet-101 or ResNeXt.
- Ability to learn robust and hierarchical representations, useful for multiple applications beyond classification.

For more technical details, the original paper [9] and official ResNet-50 documentation are recommended.

C Implementation Details. Generation of Textual Descriptors with LLaVA

The process of generating descriptions from neuronal visual mosaics was implemented using the Hugging Face `transformers` library in local mode, optimized for GPU.

Visual input and multimodal template Each neuronal mosaic, composed of 10×10 image patches, was converted into a square image of 640×640 pixels, which was provided as the visual input to the model. This input was combined with the layer-specific prompt through a multimodal conversation template in `chat` format with roles `<user>` and `<assistant>`.

Generation parameters To control the quality, diversity, and length of the responses, the following values were used:

- `temperature = 0.8`: encourages variability within the limits of the prompt.
- `top_p = 0.95`: restricts sampling to the accumulated top portion of the probability distribution.
- `max_new_tokens = 57`: limits the maximum length of the generated response to ensure conciseness and control.
- `batch_size = 64`: chosen to exploit GPU capacity for massive inference.

Zero-shot execution All generations were performed in *zero-shot* mode, i.e., without using training examples or additional context, only with the predefined prompt and the visual mosaic as input.

Automation and storage The complete process was automated through Python scripts that iterate over the selected layers and neurons, load the mosaics, generate the description, and save it in `.csv` files. Each entry in the file contains:

- neuron identifier
- layer name
- path to the visual mosaic
- generated textual description

Postprocessing Outputs were postprocessed to remove artifacts such as undesired prefixes (e.g., `"ASSISTANT:"`) and redundant spaces. This filtering ensures consistency in the textual content before being encoded into *embeddings*.

This system enabled the generation of descriptions for hundreds of neurons in an efficient and controlled manner, maintaining a clear correspondence between the visual content and the generated semantic response.

Final prompts used per layer The following prompts were ultimately employed for the generation of all neuronal descriptions used in this study:

ReLU

> You are a vision researcher specialized in low-level visual analysis. The image is composed of patches that strongly activate a single neuron from an early convolutional layer (ReLU). Do not mention the format or layout of the image.
>
> Focus strictly on low-level visual features: basic colors (such as red, green, blue, yellow, black, white, gray, orange, brown, purple, pink), edges, lines, textures, brightness, orientation, and contrast.
>
> Use simple and accurate language to describe the dominant visual pattern across all patches. For example: "horizontal black and white stripes", "diagonal red lines on a blue background", or "gray circles on a white surface".
>
> Do not mention any objects, scenes, or high-level concepts. Do not guess.
>
> Write exactly one short sentence describing the dominant low-level feature using only basic color names and orientation or shape descriptors.

Layer 1

> You are an expert in visual representation analysis. The image you see is a mosaic made of multiple image patches that strongly activate a single neuron from the first convolutional layer (Layer 1). Do not describe the format or layout of the mosaic. Focus only on the most common low-level visual features shared across the patches. These include things like color, contrasts, edges, textures, corners, curves, and geometric shapes or patterns. Describe the main visual feature using precise terms including the name of colors (black, white, red, blue, orange, purple, green, yellow...), types of orientations (vertical, horizontal, diagonal, curved...), shapes (lines, curves, corners, cross-shapes, triangles, circles, grids...), contrasts or brightness (high contrast, bright/dark background...), and repeated structures or arrangements. Do not guess or hallucinate objects, scenes, or concepts. Just describe the dominant low-level visual feature in one short, precise, and objective sentence.

Layer 2

> You are an expert in visual interpretability of convolutional neural networks.
>
> The image you will analyze is a mosaic composed of the 100 image patches that most strongly activate a single neuron from layer 2 of a CNN.
>
> Your goal is to provide a concise and accurate description of the visual concept this neuron appears to be sensitive to.
>
> Focus your description on: - Shared textures or visual patterns (e.g., dots, grids, fur, reflections, metallic surfaces) - Repeated shapes or geometrical structures (e.g., circular, linear,

angular) - Dominant colors or contrasts - If applicable, types of partial objects or visual elements (e.g., hair, scales, mesh) that appear often - Avoid naming full scenes or objects unless the same type appears in many patches.

Do not speculate or describe the image globally. Instead, abstract the visual commonality across the 100 patches.

Return your answer in a single paragraph using technical and precise language.

Layer 3

As an expert in visual representation analysis, briefly and precisely identify in one concise sentence the dominant visual element shared across these patches activating a layer-3 neuron, focusing specifically on geometric shapes, distinctive parts of recognizable objects or body parts, utensils, characteristic textures, recurring color combinations, or common materials the objects are made of (such as metallic, plastic, leather); if multiple animals, foods, or recognizable objects appear, avoid naming them individually and instead describe their common visual characteristic without describing the mosaic layout.

Layer 4

As an expert in visual representation analysis, briefly and precisely describe in one concise sentence what consistently appears across these patches activating a layer-4 neuron; directly state recognizable tangible objects, groups of related objects (e.g., kitchen utensils, sports equipment), distinctive actions or activities, dominant colors, materials, or common semantic categories, without introductory phrases or describing the mosaic layout.

D Analysis of the Internal Semantic Structure

To complement the global semantic analysis of internal neurons, this section studies how neuronal representations are organized within each layer through unsupervised clustering techniques. Specifically, the goal is to determine whether neurons encode differentiated concepts that can be grouped coherently, and to quantitatively evaluate the quality of this semantic organization.

To achieve this, the **KMeans** algorithm was applied to the semantic *embeddings* associated with each neuron. This process allows:

- Detecting internal groupings of neurons with similar semantic behavior.
- Quantifying cohesion and separation between captured concepts.
- Visualizing the conceptual structure of each layer through dimensionality reduction techniques.

Cluster computation To determine the optimal number of clusters k, three complementary metrics were used to assess the quality of the obtained groupings:

- **Inertia (Within-cluster Sum of Squares, WCSS):** quantifies the internal cohesion of clusters. It is defined as the sum of squared distances between each point $\vec{x}_i$ and its centroid $\vec{\mu}_j$:

$$\text{Inertia} = \sum_{j=1}^{k} \sum_{\vec{x}_i \in C_j} \|\vec{x}_i - \vec{\mu}_j\|^2$$

  The elbow method is used to detect the optimal point beyond which adding more clusters provides little gain in cohesion.

- **Silhouette Score:** a joint measure of cohesion and separation. For each point i, it is defined as:

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

  where $a(i)$ is the average distance between i and the other points in its cluster, and $b(i)$ is the average distance to the nearest cluster.

- **Intra-cluster cohesion:** measures cosine similarity between all pairs of *embeddings* within the same group, indicating the degree of internal semantic consistency.

The analyzed groupings are based on **multimodal *embeddings***, which have demonstrated better capacity to capture shared semantics at the neuronal level, integrating both visual and textual information. These clusters serve as the basis for subsequent analyses of the network's conceptual evolution.

Dimensionality reduction with UMAP To facilitate the visual interpretation of clusters, the **UMAP** technique (*Uniform Manifold Approximation and Projection*) was used, a non-linear dimensionality reduction method that projects representations into a two-dimensional space while preserving both local topology and global structure of the latent space.

Unlike t-SNE, UMAP is more efficient and preserves semantic continuity between regions more effectively. In this context, UMAP allows:

- Identification of well-separated or overlapping semantic clusters.
- Observation of the density and shape of neuronal groups.
- Comparative analysis between layers and their semantic complexity.

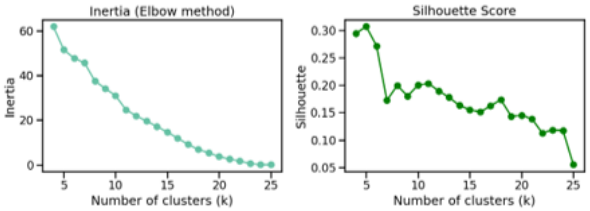
Cluster analysis The visual and numerical results obtained for each layer were analyzed according to:

- The **optimal number of clusters**, identified with the metrics above.
- **Internal cohesion** and **separation between groups**.
- The **semantic structure** revealed by UMAP projection.

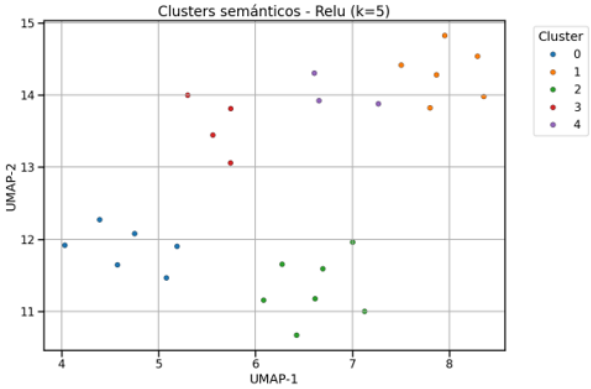
Figures for each layer are shown below, with their k-selection metrics and UMAP visualization. A specific qualitative interpretation is added for each case.

ReLU Layer

ReLU Layer



Metrics for selecting k

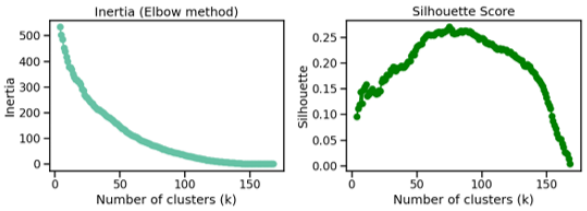


UMAP visualization with optimal k

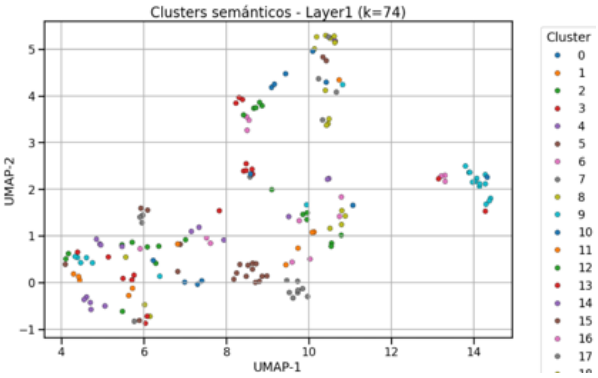
Figure 31: Results of semantic clustering for the `ReLU` layer: metrics for selecting the optimal number of clusters (left) and UMAP visualization of the obtained semantic clusters (right).

The optimal number of clusters selected was $k = 5$, indicating a simple and weakly differentiated semantic structure. The clusters correspond to basic visual patterns such as stripes, color contrasts, and simple shapes, reflecting the role of the ReLU layer in detecting elementary visual primitives.

Layer1



Metrics for selecting k

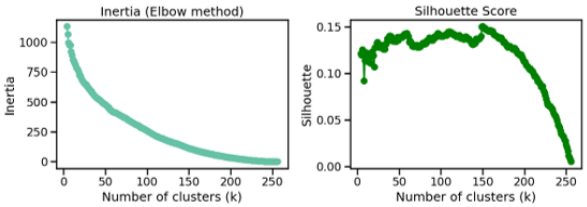


UMAP visualization with optimal k

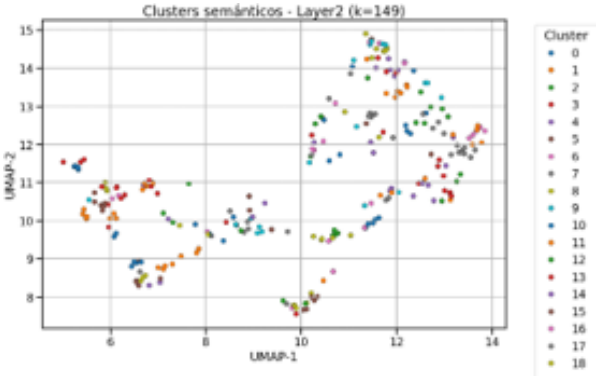
Figure 32: Results of semantic clustering for the `Layer1` layer: metrics for selecting the optimal number of clusters (left) and UMAP visualization of the obtained semantic clusters (right).

With $k = 74$, Layer1 shows greater semantic diversity, with well-defined clusters that capture textures, geometric repetitions, and structured contrasts. The organization of clusters in the UMAP map shows moderate separation and coherence with the integration of more abstract visual patterns.

Layer2



Metrics for selecting k

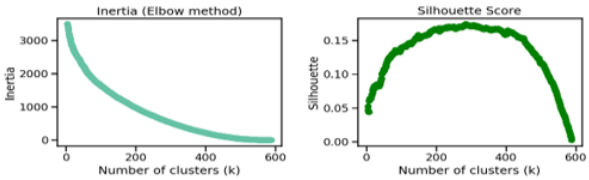


UMAP visualization with optimal k

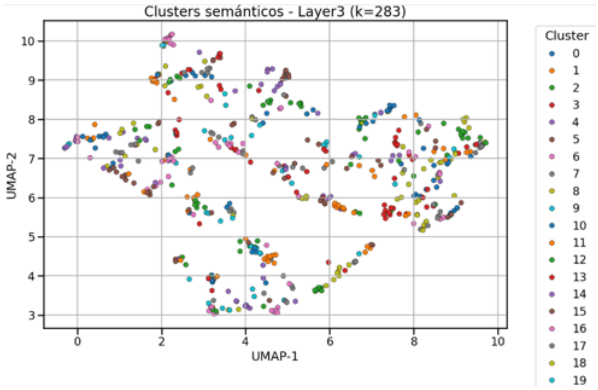
Figure 33: Results of semantic clustering for the `Layer2` layer: metrics for selecting the optimal number of clusters (left) and UMAP visualization of the obtained semantic clusters (right).

The optimal value $k = 149$ reflects a high semantic fragmentation. The clusters group descriptions related to object parts and complex visual structures. The UMAP visualization shows a continuous transition between groups, evidencing the hybrid and interpretatively rich nature of this intermediate layer.

Layer3



Metrics for selecting k

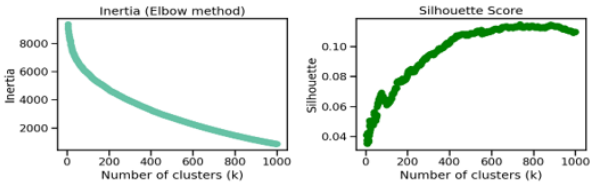


UMAP visualization with optimal k

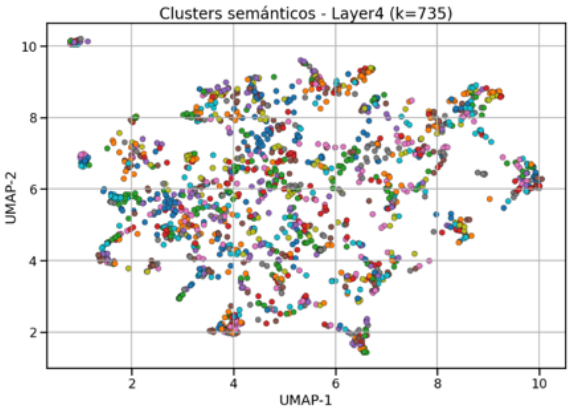
Figure 34: Results of semantic clustering for the `Layer3` layer: metrics for selecting the optimal number of clusters (left) and UMAP visualization of the obtained semantic clusters (right).

With $k = 283$, Layer3 shows high semantic specificity. The clusters display strong internal cohesion and correspond to concrete objects, recognizable categories, and highly specialized visual patterns. The grouping reflects the emergence of more defined visual meanings.

Layer4



Metrics for selecting k



UMAP visualization with optimal k

Figure 35: Results of semantic clustering for the `Layer4` layer: metrics for selecting the optimal number of clusters (left) and UMAP visualization of the obtained semantic clusters (right).

The high number of clusters ($k = 735$) reveals maximum semantic differentiation. The descriptions of the neurons in this layer refer to global scenes and complex interactions. The UMAP map shows a high density of small, differentiated groups, suggesting an exhaustive and refined semantic representation.

Conclusion

The layer-wise analysis highlights a clear evolution in the semantic organization of neurons: from an elemental representation in the `ReLU` layer, focused on basic visual attributes, to a rich and diverse encoding in `Layer4`, where complex scenes and objects emerge. This transition is evidenced both in the optimal number of clusters and in the density and separation of the UMAP projections. The results validate the hypothesis of an internal conceptual hierarchy within the convolutional network, and confirm the potential of the generated *embeddings* as a tool for open interpretation of neuronal behavior.

References

[1] Nicholas Bai et al. "Describe-and-Dissect: Interpreting Neurons in Vision Networks with Language Models". In: arXiv preprint arXiv:2403.13771 (2024). URL: https://arxiv.org/abs/2403.13771.

[2] David Bau et al. "Network Dissection: Quantifying Interpretability of Deep Visual Representations". In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017, pp. 6541–6549.

[3] Tom B Brown et al. "Language models are few-shot learners". In: Advances in neural information processing systems (2020).

[4] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations". In: International conference on machine learning. PMLR. 2020.

[5] Wei-Lin Chiang et al. Vicuna: An open-source chatbot impressing GPT-4 with 90%* ChatGPT quality. 2023.

[6] Yulin Dai et al. "InstructBLIP: Towards General-purpose Vision-Language Models with Instruction Tuning". In: arXiv preprint arXiv:2305.06500 (2023). URL: https://arxiv.org/abs/2305.06500.

[7] Jia Deng et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: Proceedings of the IEEEConference on Computer Vision and Pattern Recognition (CVPR). 2009.

[8] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: arXiv preprint arXiv:1810.04805 (2019).

[9] Kaiming He et al. "Deep Residual Learning for Image Recognition". In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016.

[10] P. He et al. DeBERTa: Decoding-enhanced BERT with Disentangled Attention. https://arxiv.org/abs/2006.03654. Microsoft Research. 2020. arXiv: 2006.03654.

[11] Junnan Li et al. BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Language Models. 2023. eprint: 2301.12597. URL: https://arxiv.org/abs/2301.12597.

[12] Haotian Liu et al. Improved Baselines with Visual Instruction Tuning. 2023. eprint: arXiv:2310.03744. URL: https://llava-vl.github.io.

[13] Haotian Liu et al. Visual Instruction Tuning. 2023. eprint: 2304.08485. URL: https://arxiv.org/abs/2304.08485.

[14] Haotian Liu et al. "Visual Instruction Tuning". In: Neural Information Processing Systems. 2023.

[15] Aravindh Mahendran and Andrea Vedaldi. "Understanding Deep Image Representations by Inverting Them". In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2015.

[16] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. Introduction to Information Retrieval. Cambridge University Press, 2008. URL: https://nlp.stanford.edu/IR-book/.

[17] Tomas Mikolov et al. "Efficient estimation of word representations in vector space". In: arXiv preprint arXiv:1301.3781 (2013). URL: https://arxiv.org/abs/1301.3781.

[18] George A Miller. "WordNet: A Lexical Database for English". In: Communications of the ACM (1995).

[19] Tuomas Oikarinen and Tsui-Wei Weng. "CLIP-Dissect: Automatic Description of Neuron Representations in Deep Vision Networks". In: OpenReview Preprint. 2023. URL: https://openreview.net/forum?id=DyV5ShqiAT.

[20] Jeffrey Pennington, Richard Socher, and Christopher D Manning. "GloVe: Global Vectors for Word Representation". In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). ACL. 2014, pp. 1532–1543.

[21] Alec Radford et al. "Learning Transferable Visual Models From Natural Language Supervision". In: Proceedings of the International Conference on Machine Learning (ICML). 2021. URL: https://arxiv.org/abs/2103.00020.

[22] Alec Radford et al. "Learning Transferable Visual Models From Natural Language Supervision". In: arXiv preprint arXiv:2103.00020 (2021). URL: https://arxiv.org/abs/2103.00020.

[23] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks". In: Proceedings of the Conference on Empirical Methods in Natural Language Processing. 2019.

[24] Nils Reimers and Iryna Gurevych. Sentence-Transformers: Multilingual CLIP-ViT-L-14 for Semantic Search. https://www.sbert.net/docs/pretrained_models.html#clip-models. 2021.

[25] Ramprasaath R Selvaraju et al. "Grad-CAM: Visual Explanations from Deep Networks via Gradient-based Localization". In: Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017.

[26] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. "Deep Inside Convolutional Networks: Visualising Image Classification Models and Saliency Maps". In: International Conference on Learning Representations (ICLR). 2014.

[27] Encord Team. Vision-Language Models (VLMs): A Comprehensive Guide. 2024. URL: https://encord.com/blog/vision-language-models-guide/.

[28] Hugo Touvron et al. LLaMA: Open and Efficient Foundation Language Models. 2023.

[29] Ashish Vaswani et al. "Attention Is All You Need". In: Advances in Neural Information Processing Systems (NeurIPS). 2017.

[30] Jason Yosinski et al. "Understanding Neural Networks Through Deep Visualization". In: Deep Learning Workshop, International Conference on Machine Learning (ICML). 2015.

[31] Matthew D Zeiler and Rob Fergus. "Visualizing and Understanding Convolutional Networks". In: European Conference on Computer Vision (ECCV). Springer, 2014.

[32] Linchao Zhu et al. "PromptBench: Towards Evaluating the Robustness of Prompts for Vision-Language Models". In: arXiv preprint arXiv:2309.08128 (2023). URL: https://arxiv.org/abs/2309.08128.