



POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Concept dataset extraction for Concept Activation Vectors: A study on open-vocabulary grounding and segmentation

Tesi di Laurea Magistrale in
Computer Engineering - Ingegneria Informatica

Alessandro Cogollo, 233022

Advisor:
Prof. Marco Brambilla

Co-advisors:
Antonio De Santis
Matteo Bianchi
Riccardo Campi

Academic year:
2025-2026

Abstract: Convolutional Neural Networks (CNNs) are black-box models that benefit from explainability frameworks; to this end, concept-based explainability methods such as Testing with CAV (TCAV) or Visual-TCAV can be helpful. TCAV is, in fact, a framework that exploits Concept Activation Vectors (CAVs), which must be trained on a set of concept images, a process that is expensive. Only a few automatic pipelines exist for extracting concept images from textual prompts; this work presents a solution: extracting concept crops from real images using open-vocabulary detection and segmentation models. The main research question investigates whether it is possible to automatically obtain a dataset that correctly represents a concept, using a textual prompt and a starting set of images as input. The proposed pipeline follows a modular architecture: an extractor, a cropper, and an augmenter. The extracted dataset has been evaluated along different axes: intrinsic quality, stability, and robustness of the CAVs, as well as the CAV similarity between the automatic and manual datasets at each layer. The proposed work presents a reproducible, documented pipeline as an alternative to the synthetic generation pipeline for the concept-based explainability field; it is intended as a principled exploratory study, providing a baseline for future investigations. Findings show an overall quality comparable to the manual baseline, measured across intrinsic metrics, CAV stability, and cross-source similarity, but still strongly dependent on the domain. The code of the pipeline and the experiments can be found here: github.com/AlessandroCogollo/Concept-dataset-extraction-for-Concept-Activation-Vectors.

Key-words: XAI, CAV, Open-Vocabulary Models

1. Introduction

1.1. Context and motivations

Artificial Intelligence (or AI) systems and models are now widely used, and as they develop and spread, more and more models, tools, and features are being integrated into everyday tasks to help and assist humans. AI refers to a broad research field encompassing subsets of models, such as Neural Networks (NNs), which can be applied to many tasks; one task in which neural networks have been found to be proficient is computer vision.

A set of models that perform well on computer vision tasks, such as classification or object detection, is the set of Convolutional Neural Networks (CNNs). CNNs’ architectures involve many hidden layers for feature extraction, non-linear transformations, and millions of parameters, making it hard to extract insights into the reasons behind a specific decision. In fact, CNNs are considered *black-box models*, as the input and output can be traced, but their behavior is opaque and hard to deduce. To overcome this limit, the research field of Explainability, or XAI, has become central to the adoption of artificial intelligence. Explainability investigates how an AI model reaches a conclusion, facilitating its adoption and providing many other benefits. As almost any research field in AI is data-driven, many explainability techniques, methods, and frameworks require substantial data for training and use; this is also the case for concept-based explainability methods, particularly the Testing with CAV (TCAV) method. The TCAV framework builds a Concept Activation Vector, or CAV, which is the orthogonal vector to the hyperplane that separates the images of a concept from random images, finding a vector dividing the random samples and the ones containing the concept; once the CAV is generated, a TCAV score is computed, permitting the evaluation of the conceptual sensitivity of a model toward a concept. One limitation to adopting this model is the need to gather or curate one or more sets of images to train the concept activation vectors, as concept-based explainability systems require a dataset of images of concepts. To simplify and accelerate CAVs training, an automated pipeline to aggregate images of concepts would be very helpful. To the best of our knowledge, there are only few automatic extraction pipelines that gather images of concepts from a textual prompt.

1.2. Proposed solution

The proposed solution aims to fill a gap in the literature by offering a different approach to the automatic extraction of concept datasets; it consists of a simple pipeline that generates a concept-centered dataset from a textual prompt and a domain of extraction. The purpose of this thesis is not only to extract images but also to assess the dataset quality and compare it with the manually curated dataset in terms of similarity. Since concepts might relate to many different aspects of a class, such as texture, patterns, materials, emotions, etc., the potential of Open-Vocabulary detection and segmentation models will be exploited, as they permit generalizing and expanding as much as possible the set of concepts that can be extracted. Indeed, these families of models enable the identification (segmentation and detection) of unseen concepts, without previous training or knowledge of the concept; at the time of writing, a state-of-the-art model belonging to open vocabulary models is the SAM3 [5], which is both a detection and segmentation model. In the proposed pipeline, after the segmentation step, image crops will be extracted using different strategies and optionally augmented with various transformations. The final part of this thesis has been dedicated to stress-testing the pipeline to ensure that the generated dataset is of high quality under different conditions and, eventually, to address the limitations of the proposed system. These evaluations have been conducted on the generated dataset itself, but also on the extracted CAVs, as the final goal of the pipeline is to extract a dataset for explainability purposes: ensuring that the obtained CAV is stable and similar enough to the baseline CAV, suggests agreement with the chosen manual dataset (containing the concept, i.e., Describable Textures Dataset) in a model’s geometry of activation.

1.3. Objectives and research questions

The following goal can briefly summarize the objective of the framework: given a textual prompt describing a concept, and a set of images from a class or, more generally, a domain, automatically generate a set of positive examples of the chosen concept, using open-vocabulary detection and segmentation models, followed by a cropping and augmentation step. Then, after training one or more Concept Activation Vectors (CAVs) on the embeddings extracted from a chosen model (or a set of models) and layer (or a set of layers), compare them with the CAVs obtained from a manually curated dataset. In other words, the main research question can be reformulated as follows: is it possible to automatically obtain a dataset effectively representing a concept, starting with a textual prompt and a set of images belonging to a starting domain?

In this work, the aim is to design and test a pipeline to extract images of a target concept automatically. The research questions below have been formulated:

1. **RQ1 - Intrinsic Evaluation:** To what extent the intrinsic quality of the manually curated dataset, compares with the one of the automatically extracted set? Also, how does it vary based on the starting domain? This measures the presence of the concept on the extracted dataset using a VLM-as-a-judge, a CLIP model, and other metrics. The hypothesis is that an automatically extracted dataset contains enough of the concept to mimic the manually curated one, but with a strong dependence on the starting class from which it is extracted.
2. **RQ2 - Stability and robustness:** To what extent can the CAVs generated from the automatically extracted dataset be considered to be stable and robust? This evaluates whether the dataset yields stable

CAVs across different positive and negative samples and across the model’s layers, as well as the number of samples required to train a stable CAV.

3. **RQ3 — Similarity:** How similar are the CAVs obtained from the automatically extracted dataset to those obtained from the manually curated dataset? Different similarity metrics are computed: cross-similarity (auto vs. manual, per layer) and intra-similarity (within a concept class).
4. **RQ4 — Ablation:** What are the components of the pipeline that most influence the results, and how does the proposed approach compare to alternative pipelines? Different ablation studies are conducted: over crop policy, augmentation, negatives, backbone, and prompt, and a direct comparison with the state-of-the-art. The aim is to identify the drivers of the intrinsic metrics and of the auto-vs-manual CAV similarity.

The above research questions are answered in an exploratory manner.

1.4. Contents

The goal of this thesis is to provide a complete, replicable pipeline to automatically extract a dataset of concepts, particularly useful for work in the XAI field. The functioning of this pipeline has been ensured by stressing and evaluating certain quality metrics. This work is divided into chapters and follows the structure below. To start, in this first chapter, the context and motivations that have led to the writing of this thesis have been listed and described; these reasons, in turn, led to the formulation of the problem, which has been investigated by answering the research questions listed above. The second chapter instead focuses on providing some background on the technologies enabling the proposed system, followed by a discussion on the state-of-the-art models for open-vocabulary object detection and segmentation, pipelines and frameworks, and some notes on the functioning of these systems; in particular, with a focus on concept-based explainability systems, such as the TCAV framework. Then, in the last section of this chapter, there will be details on the open-vocabulary detection and segmentation systems, delving deeper into the Segment Anything Model 3 (SAM3). Finally, a brief reference to another pipeline for concept images generation and dataset curation will be made. The third chapter will provide an overview of the implemented pipeline, describing in detail each experimental choice, the parameters, and their calibration. Then, the CAV training phase will be discussed, along with a brief comparison of the generated CAVs based on metrics and results. In the fourth chapter, the main experiments will be presented to answer the research questions mentioned above. In particular, the discussion will focus on the metrics and choices used to assess the quality of the automatically generated concept dataset, the similarities among the obtained CAVs, and their utility for the Visual-TCAV framework. Finally, in the last chapter, conclusions from the conducted work are reported, describing possible future work, limitations, and a brief commentary on the obtained results and contributions.

1.5. Contributions

In this section, a list of the contributions of this work is summarized as follows:

- **An automatic pipeline** using as input a natural-language prompt and a starting domain of real pictures, that extracts a set of images containing the target concept. This pipeline exploits open-vocabulary detection and segmentation, as an alternative to other techniques to curate a dataset (manually, or synthetically). The proposed architecture is composed by configurable off-the-shelf modules: an extractor (SAM3), a cropper with many different cropping policies, and an augmenter.
- **A set of methodological choices** for obtaining and comparing resulting CAVs: difference-of-means, GAP (Global Average Pooling), and shared-negative standardization. Together with the minimum number of samples required to obtain a stable CAV.
- **A protocol of evaluation** for the sets of extracted images, and the CAVs trained on them. This analysis is conducted under aspects of intrinsic quality, similarity and similarity between different samplings comparing them against a manual baseline.

2. Background and Related Works

2.1. Artificial Intelligence

Artificial intelligence, or AI, is an umbrella term that refers to a big family of models; an informal yet exhaustive definition of artificial intelligence describes it as "a branch of computer science trying to mimic the capabilities of human intelligence". Machine Learning (or ML), a subset of artificial intelligence, can be explained using a definition that well describes the functioning of AI in general; the following definition has been proposed by Tom

Mitchell, a famous computer scientist and university professor, in his book "Machine Learning": "A computer program is said to learn from experience E with respect to some class of tasks T and some performance measure P , if its performance on T , as measured by P , improves with experience E " [19].

In other words Mitchell describes machine learning systems as computer programs that are capable of learning from experiences and errors. This feature enables ML systems to perform a wide range of tasks without being explicitly programmed to; this can also justify its fast and huge diffusion. The rediscovery and diffusion of artificial intelligence models took place mainly in the last decade, when its models found wide application in many areas, mainly due to factors such as the invention and development of transformers, better computational units and increased commercial interest. Although AI is often associated to recent times, the idea of mechanical intelligence was formalized first in 1950 by Alan Turing in his article "Computing Machinery and Intelligence" [32]. Later on, a first concept, was theorized under the name of artificial intelligence by McCarthy et al. in 1955 with the paper "A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence" [18]. This first wave of enthusiasm remained temporarily frozen during what Minsky and Schank in 1984 called *AI winter*, due to limited developments in the field, linked to low computational power and low funds; in recent years, investments in the field have boomed, and new areas of research under the artificial intelligence umbrella are growing. In addition to machine learning, there are many other subcategories of artificial intelligence, the most relevant ones are shown in figure 1.

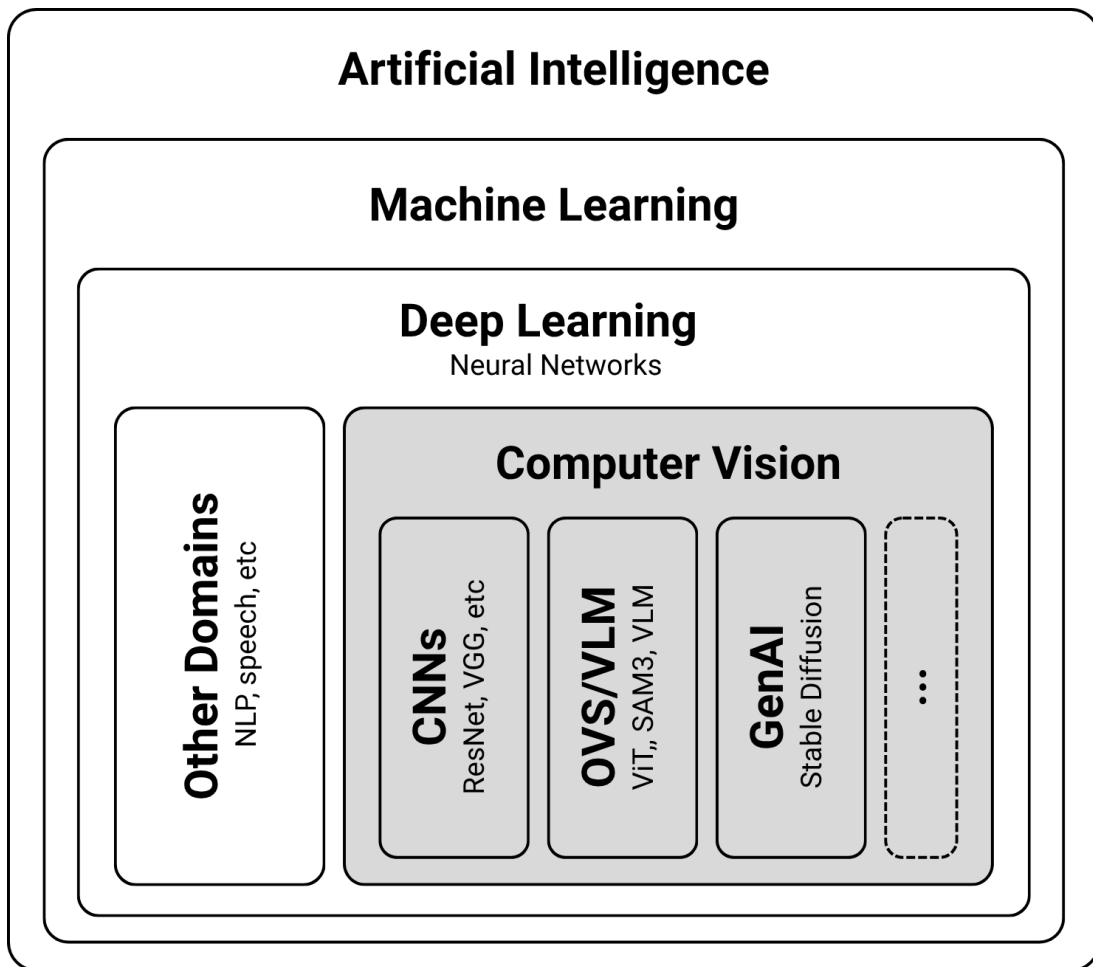


Figure 1: A simple AI Taxonomy

As shown in figure 1, one of the specializations of artificial intelligence is deep learning, which has become more and more popular starting from the 2010s. Deep learning models are called neural networks, and are now adopted in many areas; these models are inspired by the functioning of human brains, and are composed of multiple sets of layers of different kinds (input, at least one hidden layers, and output layers). Layers are made of multiple linked neurons, which are the atomic element of a neural networks, with a very similar functioning to human neurons. These neurons receive input signals that are summed together with a bias, and the result of the sum is used as an input for an activation function, deciding whether to pass a signal to the next neuron. The quick diffusion of Deep Neural Networks (DNNs) can be associated also to its evidently enormous potential. DNNs are in fact able to approximate and solve many real-world problems with sufficient levels of accuracy and

precision. The potential of DNNs is quite clear, and led to development of numerous models, categorized into different families each dedicated to performing different tasks; among them, some of the most relevant and widely adopted models in terms of diffusion and reliability are those processing images, which enable computer vision. These models are able to perform many tasks, such as recognition, grounding, segmentation, classification, and many others. Among computer visions models, the Convolutional Neural Networks (CNNs), are considered to be the standard models to be adopted when dealing with image processing, even though other promising models are gaining popularity, such as Vision Transformers [11].

CNNs, being deep neural networks themselves, are composed of multiple layers with different functions: activations layers, pooling layers, input layers, convolutional layers, flattening, fully connected and output layers. Each of these layers transforms the input volume to another volume, processing it to extract features, therefore spatial relationships between pixels. A simple architecture of a CNN is shown in the figure 2.

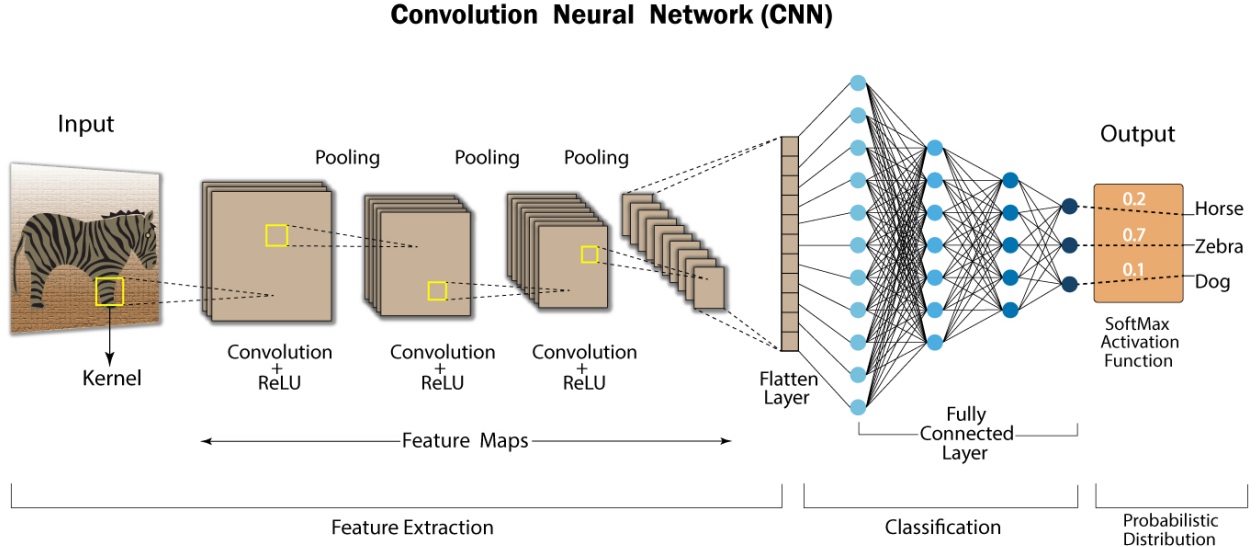


Figure 2: A simple representation of the architecture of a generic convolutional neural network, clearly showing the feature extraction layers, and the classification steps. Taken from a [LinkedIn](#) article [14]

As mentioned before, the functioning of CNNs reminds us of the way in which the human brain works, as they are designed to mimic it. Currently, neurosciences are able to understand that certain areas of the human brain are dedicated certain functions, but scientists are only partially aware of its full functioning; the same way, we know that certain areas of a neural network help with certain reasoning, and that the calibration of certain parameters may change their functioning, but we’re still not able to fully understand how a neural network learns and make decisions. In fact, even though convolutional neural networks are widely studied and adopted, they are considered to be black-boxes, making them difficult to be trusted, especially in critical decisions.

Another family of artificial intelligence models adopted in this work are the Vision-Language Models (or VLMs) which allows multimodal interactions between text and images or videos. This is achieved by mapping relationships between images and textual prompts, allowing the generation of images from text or text from images. This approach consequently presents the limitations of a VLM, since OVDs and OVSs share a similar functioning and evaluate the proposed pictures similarly, introducing a circular dependency.

2.2. An overview on Explainable AI

Explainable AI (or XAI) is a field of computer science that has been explored first at the end of the 70s, to explain how an intelligent system works and to justify predictions and decisions taken by AI models. Explainable AI, has grown in popularity after the wide adoption of deep neural networks in many fields, to assess how Artificial Intelligent models work, often with a focus on neural networks and deep learning models. Frequently, many taxonomies (eg. F. Xu et al, in “Explainable AI: A Brief Survey” [35]) show two main subcategories of techniques for XAI: transparency design and post-hoc explanation. Transparency design is a subset of XAI techniques documenting and helping inspecting the design and build choices of an AI model, so that the stakeholders can be able to understand and evaluate its accuracy but also fairness, tendency to biases and other factors relevant to evaluate its functioning. Transparency design has become relevant with the spread of AI models, and the introduction of new laws, regulations and policies ensuring the “right of explanation” [13, 22].

Post-hoc explanation techniques (also referred to with *post-modeling explainability*) instead focus on explaining how an AI model made a decision, after it has been taken (*post-hoc*). These sets of frameworks, techniques and models provide the user with many different types of explanations, such as textual, visual, local, etc, but also explanations by examples, simplifications, etc. There are also different types of post-hoc explanation frameworks focusing on different aspects; Arrieta et al [1] provide a good taxonomy of explainability methods in general, and lists some post-hoc explanations techniques, as there are post-hoc methods model agnostic and model specific, models focusing on deep models and others on shallow models. From another point of view, explainability methods can be also distinguished into the research field of traditional XAI and concept-based XAI (C-XAI), which will be explained and expanded in the next section.

2.3. Concept-Based Explainability

2.3.1 Definitions and taxonomies

Before going into detail exploring methods and techniques for concept-based explainability, it is fundamental to give a clear definition of a *concept*. The following definition, as with many others in this thesis, can be quite different; a possible definition of a concept, as suggested by Molnar in his book "Interpretable Machine Learning A Guide for Making Black Box Models Explainable" [20], can be the following:

Proposition 2.1. *"A concept can be any abstraction, such as a color, an object, or even an idea".*

Beyond this good yet simple definition, concepts need also to be understandable by humans, and to satisfy certain requirements, they must have certain properties, a list of them can be found in the work of A. Ghorbani et al [12], and is reported below:

- *Meaningfulness*: a group of individuals will locate the same concept in the same area of pixels.
- *Coherency*: samples containing the same concept should be similar between them, but different from other samples containing a different concept.
- *Importance*: concepts should influence the prediction or not of a class.

Starting from the definition above and these properties, the reader can now go further and into the details of the concept-based explainability. Concept-based explanation (C-XAI) methods are becoming more relevant as they are capable of providing human-friendly explanations. As explained by Poeta et al [23], C-XAI frameworks are capable of explaining, as an example, how CNNs classify images using concepts, and adhering to specific criteria such as being meaningful, coherent and relevant to the final class, as well as explicit and faithful.

Concept-based explainability methods can be divided themselves into two categories, as shown in the taxonomy proposed by Ciravegna [9], these are:

- *Concept-based Models* incorporate concept prediction in the model architecture. This forces the NN to suggest a set of predefined concepts or attributes, beyond performing the primary task. These concepts can be treated in different ways: annotating them, automatically extracting them, automatically generated.
- *Post-hoc Concept-based Explanations* these methods are focusing on providing explanations for already existing methods, without the need of modifying the network architecture. Used concepts can have various origins (external datasets with attributes' notation, such as the Broden dataset, or DTD, as explained later), or can be represented by input prototypes that encapsulate common patterns or areas of interest within the network's learned representations. These frameworks are used to gather insights on many CNN classes.

The frameworks explained in the next pages fall into the category of supervised post-hoc methods. A simple and intuitive definition of supervised and unsupervised post-hoc methods follows the traditional definition of supervised and unsupervised models in AI, and is briefly given below:

- *Unsupervised*: these sets of methods have the goal of identifying clusters of samples that have an influence on a prediction or on a class. These techniques provide explanations in terms of Class-concept relations. They don't simply select features solely based on their saliency or internal similarity but rather for their capability to represent other input samples. Some examples falling under this XAI methods' category are: ACE, On Completeness aware.
- *Supervised*: Supervised explainability methods based on concepts take a set of concepts and analyze how a neural network responds to these concepts. This evaluation is done to evaluate which concepts were learned by the CNN, at which stage, and how the network is influenced by them. These methods provide different insights, such as how a specific prediction, or an entire class weight correlates with the projection of a concept in the latent space of the model. Another explanations provided by other methods is about the study of the activation of a single hidden node over the entire dataset to associate them with a concept. TCAV belongs to this framework family.

Before finally moving on to explaining the specific techniques inherent for this work, it is worth mentioning another classification criteria, as these definitions are used later while presenting a few explanation frameworks: global and local explanation techniques.

- *Local explanation*: Local explanations methods provide insights focusing on single cases. These techniques can be used in many fields (finance, banks, transactions, etc), and focus on a particular prediction.
- *Global explanation*: Global explanations focuses instead on the global model, analyzing which factors are relevant for a whole class or model. These methods permits us to make a rank of variables that influence the model the most.

2.3.2 CAV and Testing with CAV

Now that the necessary clarifications on the definitions, categories of methods, and properties have been made, the relevant frameworks can be introduced. Here, the focus is on Testing with CAV (or TCAV) [16], which is a supervised post-hoc concept-based explanation method, inherently providing global explanations. In some taxonomies, TCAV is also categorized as a meta-explanation technique [29], as it explains the internal logic of a model by exploiting a secondary mathematical structure (CAVs). In the next sections, there will be also a mention on the Visual-TCAV [27] framework, which in some ways expands TCAV functionalities, among the various features, providing local explanations. The TCAV technique is based on the notion of a Concept Activation Vector (CAV), as it represents "a vector in the direction of the values of that concept's set of examples" [16]. Basically it consists of a linear classifier, which learns to separate concept's sample from other random samples; the orthogonal vector represents the CAV in question. More precisely, CAVs are, as defined by Xu et al [21] "a vector representation of a concept found in the activation space of a layer of a neural network". In order to produce a concept's CAV, two sets of samples need to be curated: D_c^- and D_c^+ ; the first one, D_c^- represents a set of random (or negative) samples, meanwhile the second one D_c^+ represents a set of positive samples, thus images containing the concept. When these sets are fed to the CNN, activations can be extracted from a layer l for the images used as input. Activations are used to train the CAVs on a concept, at a specific layer; CAVs are specifically used as a separator between the activations of the positives and negatives images, with the following formula:

$$\mathbf{v}_c^l \cdot \mathbf{f}_l(x) + b = 0, \quad (1)$$

where $\mathbf{v}_c^l$ is the normal vector of the hyperplane separating the positive activations $A_{c,l}^+$ from the negative ones $A_{c,l}^-$, and b is the intercept. Once a CAV is obtained, the sensitivity of the model to the concept represented by the CAV needs to be assessed. In other words, the goal is to understand whether that model made a decision since a certain concept was present or absent in the input image. Testing with CAV, exploits CAVs to assess to what extent certain concepts are taken into account by a model. These concepts are human-friendly concepts (eg. striped, dotted, etc) which are used to assess whether, as an example, a model classifying a zebra as such is taking into account the stripes on the skin of the animal.

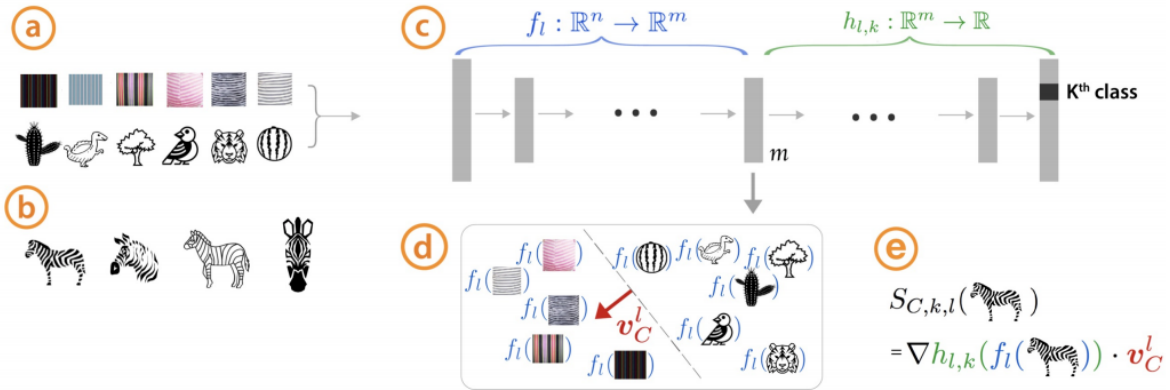


Figure 3: Overview of the TCAV pipeline: a set of concept images are fed to the CNN, up to layer l , where the activations of the layer are extracted; there, a linear classifier is trained to separate the activations of the layer when fed with a concept's sample and those of negative images. The CAV $\mathbf{v}_c^l$ is the orthogonal to the boundary separating the two concepts. Then, directional derivatives are used to compute the TCAV score. Taken from Kim et al [16].

In the TCAV framework, as shown in the figure 3, the CAVs generation follows the same steps described above. Once the CAV has been trained, the sensitivity of the model to a certain concept (*conceptual sensitivity*) is assessed using CAVs and their directional derivatives. Practically, the framework gauges the "sensitivity of the predictions to changes in inputs, towards the direction of a concept, at neural activation layer l " [16]. If $f_l(x)$

represents the activations for input x at layer l , the sensitivity of a layer l , on a class k , to a certain concept c can be computed as follows:

$$S_{c,k,l}(x) = \lim_{\epsilon \rightarrow 0} \frac{h_{l,k}(f_l(x) + \epsilon \mathbf{v}_c^l) - h_{l,k}(f_l(x))}{\epsilon} = \nabla h_{l,k}(f_l(x)) \cdot \mathbf{v}_c^l, \quad (2)$$

The TCAV paper then proposes a TCAV Score, which consists on the conceptual sensitivity across entire classes of inputs, which is computed as follows:

$$\text{TCAV}_{Q_{c,k,l}} = \frac{|\{x \in X_k : S_{c,k,l}(x) > 0\}|}{|X_k|}, \quad \text{TCAV}_{Q_{c,k,l}} \in [0, 1]. \quad (3)$$

The explanations provided by TCAV fall into the category of *global explanations* as this model is providing explanations for the whole model and not on a specific prediction; it is indeed providing information on the sensitivity of a model to a certain concept but cannot assess the sensitivity of the model on a specific decision, nor the actual contribution of the concept to the decision, or some kind of contribution map.

To extend the capabilities of TCAV and overcome the intrinsic limitations of global systems, an extension of the TCAV system has been proposed by De Santis et al with the work of Visual-TCAV. The functioning of the Visual-TCAV is quite similar to the one of the TCAV, but introduces some improvements; Visual-TCAV can in fact be seen as a sort of expansion of TCAV, as it joins both local and global explanation into a single hybrid framework. More precisely, Visual-TCAV is providing as an output:

- *Local explanations*: a concept map, which is a saliency map of where the model identified the chosen concept, and the concept attribution, which is a numerical index of the importance of the concept for a selected class.
- *Global explanations*: is obtained by the averaged attributions of multiple images, and represents the extent to which a certain concept is able to influence the decision of the model.

The pooled-CAV is used as input to compute the concept map. A ReLU function is applied to exclude the regions of the picture where the activations are negatives, thus considering only the regions that actually correlate with the concept; a **min-max** normalization is used following the policies described in the original paper. To provide further local explanation, Visual-TCAV introduces attribution for a concept at each prediction. To compute local explanations, activation maps are taken from each level we are interested in, without averaging them, but preserving its dimensions; obviously, dimensions depend on the specific backbones. In the case of CAVs computed for the Visual-TCAV framework, instead of logistic regression, difference of means is adopted to improve stability, as better discussed in the next sections. At this point, local features are projected onto the previously computed CAV. Now the concept map is generated; alignment scores are transformed in a visual map. Spatial scores are finally put together: the projected activations are weighted by the gradients of the predicted class and summed over the spatial dimensions, resulting in a single scalar quantifying the relevance of that concept for that specific prediction.

Instead of computing the TCAV score, an average concept attribution score is computed. Visual-TCAV represents an improvement with respect to the traditional TCAV, as it evaluates the weight of a concept on the whole model or on a whole class. The full pipeline of the Visual-TCAV framework can be inspected in figure 4. Other extensions and changes in the Visual-TCAV framework, with respect to the TCAV framework can be further explored in the original work, as they're explained in detail in the original paper, and are not central to this thesis.

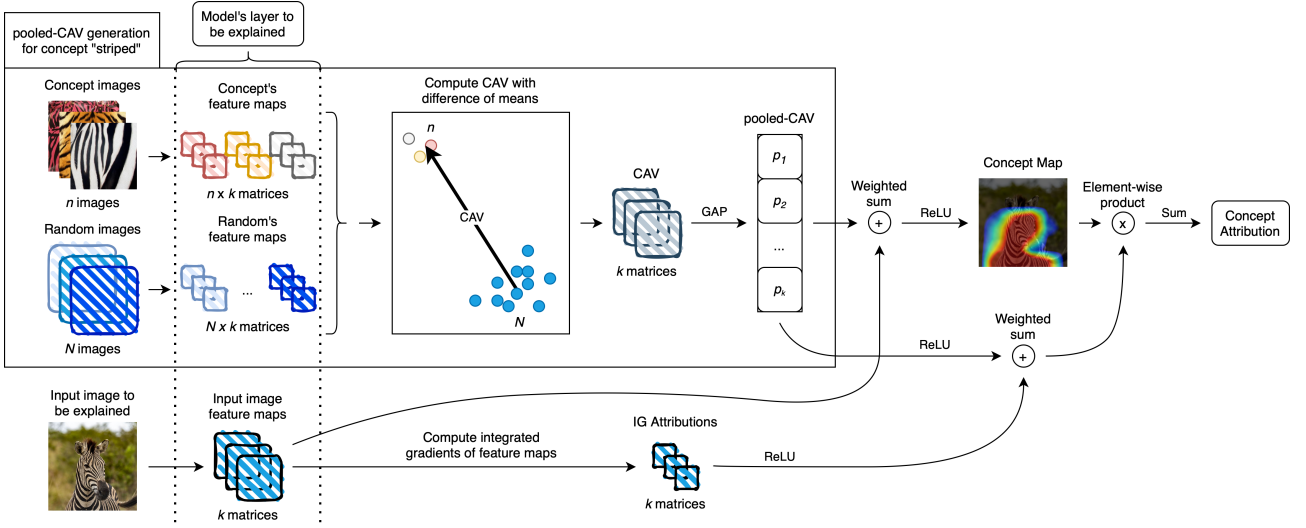


Figure 4: Overview of the Visual-TCAV pipeline. Taken from De Santis et al [27].

2.4. Open-Vocabulary Detection and Segmentation

As mentioned in one of the previous sections, the reason that justifies the work presented in this thesis is the necessity of establishing an automatic pipeline to extract concepts datasets; to obtain such a goal, the proposed framework aims at locating concepts from a starting domain, and extract them to automatically curate a dataset of concepts. For this reason, this section will present the problem of automatically identifying and segmenting unseen objects; the characteristics and functioning of the typology of models used here will be presented, focusing on two of these specific models. The focus will be on open-vocabulary object detection and segmentation models. In the work of Zhu et al [36], the target of Open-Vocabulary Detection and Open-Vocabulary Segmentation models is described as the goal of detecting or segmenting "unseen or novel classes that occupy semantically coherent regions or volumes within an image, video, or a set of point clouds". As just stated, Open-Vocabulary Detection (OVD or OVOD) and Open-Vocabulary Segmentation systems (OVS) are moreover systems capable of detecting unseen features; they're relevant to the case of this work for this specific reason, as they're able of expanding the capabilities traditional object detectors and object segmentation models capacities, being trained on a fixed set of classes (closed set). Notably, closed set object detection (ie. Faster R-CNN, YOLO, [25, 26]) and segmentation systems often perform well on the chosen closed set of objects, patterns, concepts to locate and segment, but are not able to generalize the same performances on unseen objects, posing clear limits to the wide adoption of these models to more general areas; meanwhile, open-vocabulary systems enable models to train without labeled objects, enabling the capacity of detecting unseen objects, providing great flexibility in finding and locating semantic regions, objects, textures, etc in a picture.

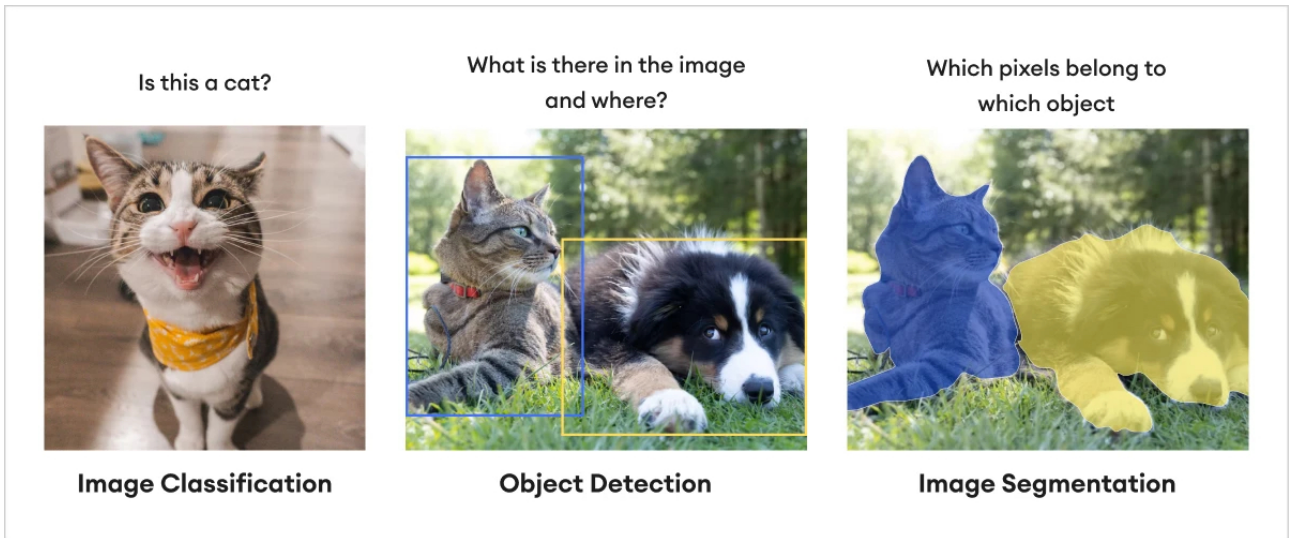


Figure 5: Differences in three computer vision tasks: classification, detection and segmentation. Taken from SuperAnnotate.com [30]

2.4.1 Open-Vocabulary Detection models

Open-Vocabulary Detection and Open-Vocabulary Segmentation models, are dedicated to two very different tasks. Open-Vocabulary Detection systems represent an extension of the closed-set object detection models, which, as their name suggests, are systems able to identify and detect multiple objects within images. Concretely, this task consists in detecting one or more objects or concepts in an image, by returning one or more bounding boxes, often associated to a score assessing the probability of that bounding box of containing the object or concept. The Open-Vocabulary Detection task begins by receiving a textual input, usually written in natural language, also referred to as prompt, or query, and one or more images in which we're trying to find the bounding box containing the object described with the prompt. A bounding box is the smallest rectangle, described by a set of coordinates, containing the object or the concept searched. The functioning of OVDs is quite different from the closed-set object detection; the idea is to compare the textual embedding of the prompt, and the embedding of the input image. Basically, a shared embedding space is used to find similarities between the embedding of the textual prompt and the embedding of the image.

2.4.2 Open-Vocabulary Segmentations models

Open-Vocabulary Segmentation systems are models dedicated to similar tasks with respect to the Open-Vocabulary Detection systems, but with the specific goal of assigning precise masks (instead of bounding boxes), starting from a textual prompt. Their functioning is quite similar to the one of the open-vocabulary detectors, as they are capable of working with unseen concepts, but with a different output; a mask consists of a pixel-precise set of coordinates describing the boundaries of the object (or concept). The state-of-the-art of OVS models is represented by the Segment-Anything-Model (or SAM), proposed in 2023 by Kirillov, et al [17]. SAM has been later improved in newer and better-performing versions, called SAM2 and SAM3. SAM3 is particularly suited to this setting, as it is built to work well with generic descriptions of concepts. Its functioning is presented in the next subsection.

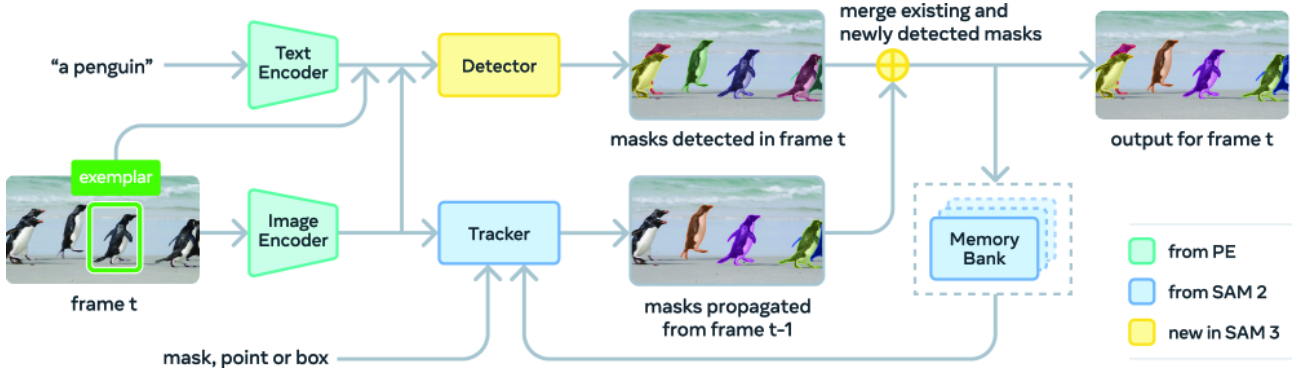


Figure 6: Segment Anything Model (SAM) 3 pipeline. Taken from Carion et al [5].

2.4.3 SAM3

As explained in the paper by Carion et al [5], SAM3 introduces the Promptable Concept Segmentation (PCS) task, which is formalized as "taking text and/or image exemplars as input, and predicting instance and semantic masks for every single object matching the concept, while preserving object identities across video frames". The PCS task works well with "simple noun phrases" (NPs), such as "red apple", or "striped cat", enabling the recognition of atomic visual concepts, meanwhile "it is not designed for long referring expressions or queries requiring reasoning", even though it can be combined with Multimodal Large Language Model (MLLM) to further improve prompts.

SAM3 represents the state-of-the-art model for object segmentation and detection as it joins both these tasks in a single model, permitting the extraction of bounding boxes and segmentation mask without the need to implement a complex pipeline; also, SAM3 outperforms all the other models in the same task's category, as shown in the benchmarks reported in the original paper [5]. The architecture of this model is shown in the figure 6, and is composed of different components; the core modules are a detector and a tracker, that are decoupled but working using the same Perception Encoder (PE) vision backbone. In this work, the focus will be on the functioning model using a textual input. First, using a text encoder, the textual input is converted into an embedding representing the query; the same process is performed using an image encoder when using images as inputs. The detector module is based on DETR (DEtection TRansformer) [6], and, after the detector module, a presence head is implemented to guarantee the presence of the concept in the image before generating bounding boxes and masks.

2.5. Other works

Another work worth mentioning is the work of Campi et al [4], which proposes a different solution to the same problem introduced in the previous sections. Their proposed solution, specifically, follows another approach: the generation of synthetic images by text-to-image pre-trained models. They are generating concept images from natural language prompts using models such as Stable Diffusion (SD) 1.5 and others. Once the pictures have been generated, a filterer based on a CLIP model (Contrastive Language-Image Pre-training model) [7] is used to select only images that actually contain the chosen concept; CLIP models, in fact, are capable of recognizing subjects in a picture by comparing in the same vectorial space, the embeddings of both the image and the textual prompt. Once the pool of concept images is available, CAVs are trained using the difference of means method, and GAP is trained using a pool of 500 random images from random classes of the **ImageNet** dataset [10] (300 for the RQ4). These CAVs are extracted from different layers of three backbones and then compared using cosine similarity with CAVs obtained from manually curated datasets. The proposed pipeline achieves high similarity scores, with peaks around textured concepts.

Another related work is the framework proposed by Astolfi et al. [2], where, instead of proposing a new generation method, an evaluation on the zero-shot text-to-image (T2I) generative models is conducted; the goal here is to evaluate their capability of generating synthetic concept datasets, and assessing their faithfulness to real concepts under different point of views. Their findings show that zero-shot generation, without fine-tuning, does not produce realistic concepts, emphasizing open challenges in using generated data. This result is one of the reasons that lead to the proposal of this pipeline, as purely generative pipelines does not seem to be able to produce realistic concepts. This makes the extraction of real crops an interesting alternative to be evaluated.

3. Methodology

This chapter presents the methodology adopted to implement the pipeline and explains the chosen approach for conducting the experiments. An overview of the pipeline will be provided, along with the techniques used to train and compare CAVs and any other experimental choices.

3.1. Problem formulation and notation

The generation of a concept dataset using the proposed pipeline requires an input and generates an output. Formally, inputs and outputs for the task in question are listed in this subsection.

- **Input:** consists of a pair $I_{C,c} = [S_C, p_c]$:
 - S_C , a set of images from a chosen starting domain (dataset and class, or classes), where C is the set of classes of the starting domain $\{e.g., \text{zebras}, \text{ladybugs}, \text{etc}\}$.
 - p_c , a textual prompt describing a concept in natural language c .
- **Output:** a set of images containing the concept $D_c^{automatic}$;

Below is a list of notations used in the next pages:

- $f(\cdot)$ the model that is being evaluated, in the case of the baseline of this thesis, a **ResNet50**. Given an input image x , the notation $f(x)$ indicates the forward pass; $f_l(x)$ represents the activations at layer l .
- L The set of layers, e.g., the collection of layers $l \in L$ where CAVs are trained and evaluated.
- When referring to sets of images, CAVs, or any other element automatically extracted, or manually curated, the following notation will be adopted $D_c^{automatic}$, or D_c^{manual} , representing the concept image set for c , produced *automatically* or curated *manually*.
- D^- : The set of *negative* images used to train a CAV against a concept set; in the proposed setup, the negative set is obtained by randomly picking samples from **ImageNet**.

To evaluate whether and how much the quality of the extracted dataset is dependent on the starting domain, every concept has been mapped to a single starting class, multiple starting classes, and a set of random classes; the sets of images extracted using a single class, multiple classes, and random classes are referred to, respectively, as **single-class**, **multi-class**, and **random-class** datasets. The full mappings of these classes have been reported in the table 4. The classes of the starting domain are intentionally chosen letting **ChatGPT** propose a set of **ImageNet** classes to extract from for each concept class; this is an experimental choice, simplifying the proposal of the classes when dealing with a large (1000) amount of classes to choose from, even though the results are naturally depending to the model of **ChatGPT**; for reproducibility purposes, the classes suggested by the model have been kept fixed for the various experiments. The implementation of an automatic class selector, based on LLMs, will also be discussed as a possible expansion of the proposed pipeline in the "future works" chapter. The **random-class** starting domain has been introduced to evaluate whether this architecture can extract images without strongly depending on a single class or a set of classes that clearly contain that concept, thereby overcoming the limitation of needing to manually choose the starting classes.

To assess the quality of the generated dataset, a ground-truth dataset is needed for comparison; this requirement is not strictly related to the extraction pipeline but is necessary to evaluate its quality. The ground truth, or manually curated dataset, can be referenced as D_c^{manual} , and the associated CAV as v^{manual} .

3.2. Pipeline Overview

As explained before, the goal is to transform a concept (described in natural language) and a set of images into an automatically generated dataset of images that clearly and predominantly depict the chosen concept. As inputs, the proposed pipeline requires:

- A sufficiently large set of pictures composed by one or more classes (e.g., *zebras*).
- A clear concept, describable with a few words.

As outputs, the pipeline returns:

- A set of pictures containing clearly and predominantly the concept input.

An overview of the pipeline that will be described in this section is presented in this picture 7.

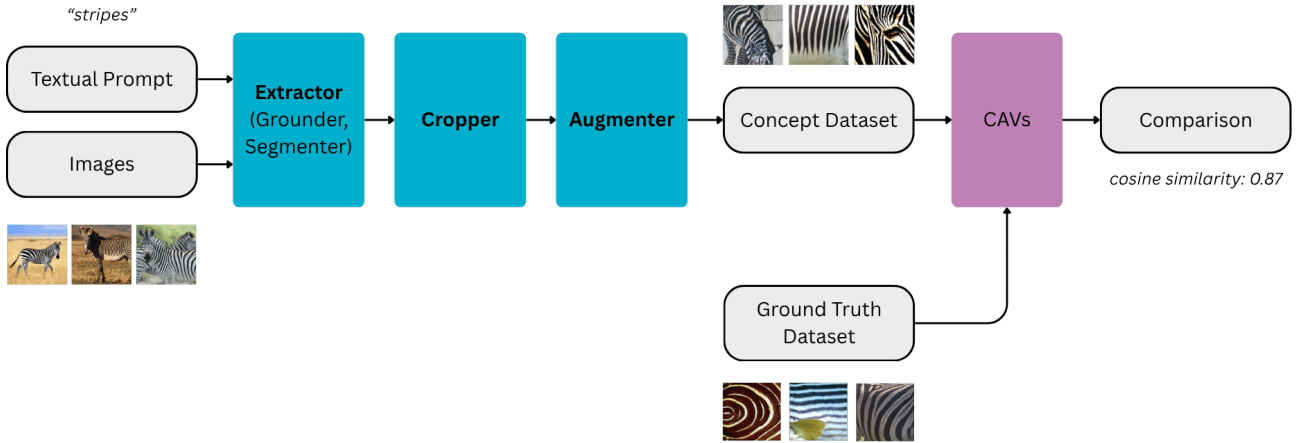


Figure 7: Concept dataset extraction pipeline: a textual prompt and a set of images from a proposed starting domain are fed to a set of modules (extractor, cropper, augmenter) that, with a chosen set of policies, generate a dataset of concepts. Starting from a dataset of concepts, a set of CAVs can be easily computed and compared with CAVs trained on manually curated image sets.

To better explain its functioning, the pipeline can be divided into modules, as shown in figure 7.

3.2.1 Extractor

The extractor module is tasked with a prompt and attempts to extract a concept from an image, returning a bounding box or a mask containing the concept. In this work, the extractor is implemented using the aforementioned SAM3 model (Segment Anything Model) [5], an open-set object segmentation model developed by Meta¹. As explained in the previous chapter, SAM3 natively integrates a grounding model, making the implementation of two distinct grounding and segmentation modules unnecessary. This module takes as input one or more images and a textual prompt, and examines them to identify the concept described in the prompt. The output is one or more masks for each picture.

3.2.2 Cropper

This module receives as input one or more masks that, if overlapped with the image processed by the extractor, bound the concept. In turn, one or more crops are produced in the interested area. The aim of this module is to extract the concept and obtain a new picture that prominently features it. The cropper can crop the original image using different policies, such as:

- **Sliding Window:** looks for the biggest squared pixel area in the segmentation mask proposed by the extractor module. It first calculates the size of one side of the square by combining the minimum limit (the smallest size possible to consider the picture valid) and the maximum one, with a random component weighted on the parameter `size_bias`; then, the mask is transformed into a zero-one matrix, and applies

¹To generate the automatically extracted concept dataset, the SAM3 implementation is downloaded directly from the official [facebookresearch/sam3](https://github.com/facebookresearch/sam3) public repository using the 3.1 checkpoint.

a two-dimensional cumsum. This technique dynamically calculates the sum of internal pixels in any sub-window in real time. By moving this window over the image, it can identify the maximum area of pixels belonging to the mask. If the extractor outputs more than one mask, the cropper chooses one.

- **Bbox**: Crops the area of the bounding box provided by SAM3, forcing the ratio to be a square. It starts by identifying the geometric center of the bounding box (c_x, c_y), then choosing the larger of width and height from the original bounding box. This ensures the object will not be cropped. Now, the side of the square is constrained to be no smaller than the minimum window size or no larger than the original picture size. Using the formula: $(c_x - side//2, c_y - side//2)$, it applies a limit (**min/max**) in order for the crop not to go out of the picture.
- **Center mask**: it creates a square crop of a fixed dimension centered on the segmentation mask's center of mass (center of gravity), ignoring the rectangle's borders. It first chooses the side dimension based on the required window, ensuring it does not exceed the picture's dimensions. It then extracts all indices of pixels where the mask is active (True) and computes the arithmetic average along the axes X and Y. The provided results correspond to the mask's center of gravity. If the mask is empty for any reason, it automatically uses the bounding box center. This technique centers the square on the center of gravity and clamps it to the image margins so it does not extend beyond the image's borders.
- **Largest bbox**: this represents a purely geometric policy that ignores the segmentation masks and simply scans every bounding box returned by the extractor and chooses the one rectangle with the biggest area.

In the figure 8, an example of different cropping policies is shown.

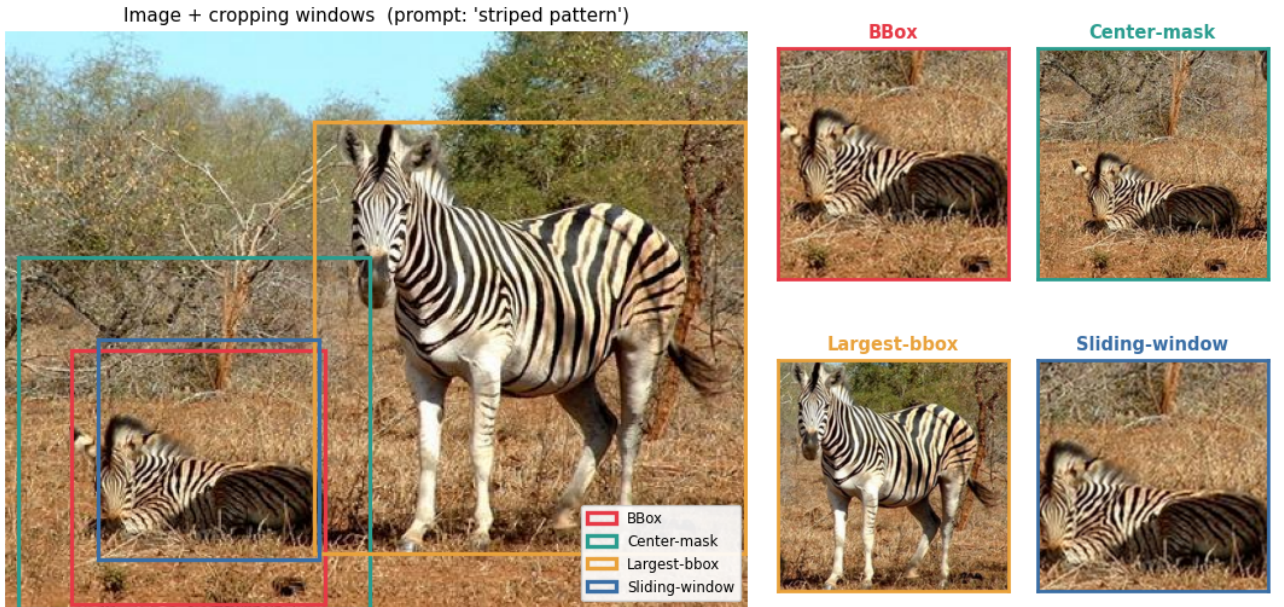


Figure 8: An example of the four cropping policies available, applied over the picture of a zebra

3.2.3 Augmenter

This module augments cropped images by randomly combining standard augmentation techniques. The augmentation module can be deactivated and is proposed for two reasons: either to increase the number of images when the number of crops extracted does not reach the minimum, or to augment crop diversity when extracting images from only one or a few classes. The augmentation techniques proposed are the following:

- **Rotation**: if enabled, rotates the image orthogonally, choosing randomly between a set of four rigid angles: 90°, 180°, -90°, or 0°. Choosing to fix the rotation angles helps avoid introducing a black background when rotating the picture at intermediate angles (e.g., 45°).
- **Shearing**: applies a distortion on the X axis. A random angle within a chosen range is picked, and the image is laterally deformed, rotating the Y axis of that angle to simulate a change in point of view.
- **Crop and resize**: extracts a central crop and zooms it by 25%. In other words, it simulates a 25% zoom-in, forcing the CNN to focus on the image's internal details.
- **Flipping**: randomly selects one of the four flipping directions: horizontal flip (**hflip**), vertical flip (**vflip**), a simultaneous flip on both axes (**hv**), or no flips (**none**).
- **Color Augmentation**:
 - **Brightness**: multiplies the light channels by a random uniform factor (e.g., 0.6 to make it darker, 1.4 to make it brighter).

- **Contrast:** increases the gap between light and dark regions by extracting a factor from a range (e.g., 0.5 - 1.5), where lower values make the image appear closer to a uniform grayscale, while higher values yield greater contrast.
- **Saturation:** alters the intensity of vivid colors. A factor close to 0.5 makes the image look like a black-and-white picture; meanwhile, a factor close to 1.5 makes it more vivid.
- **Hue:** applies a shift in the chromatic spectrum, extracting a delta value (e.g., between -0.3 and 0.3). This represents the most aggressive alteration of the images, as it can result in different colors in the texture or surface (e.g., a red texture becoming a blue one).

Before saving the image, the algorithm performs a sanity check to determine whether the final pixel matrix is identical to the input one; if so, a horizontal flip is applied to ensure a difference from the vanilla one. The augmentation hasn't been performed on the fly; after generating the vanilla dataset, it is saved for visualization and various experiments. Some random augmentation examples are shown in figure 9.

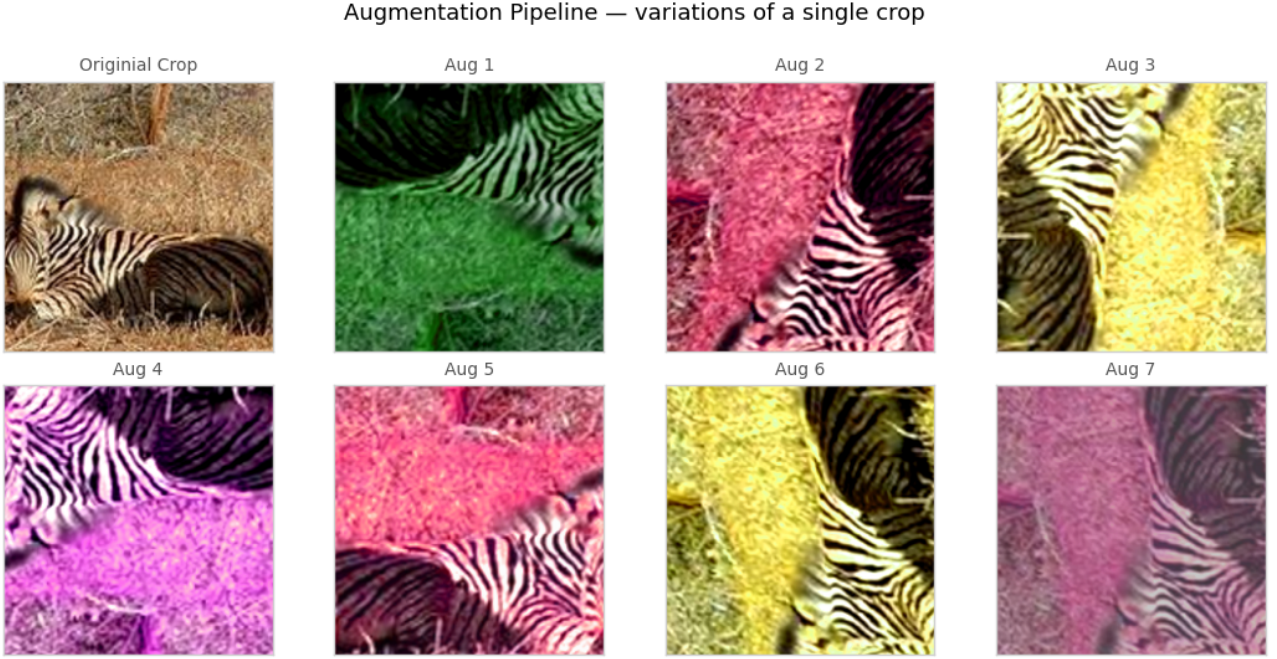


Figure 9: Seven random augmentations of a sample image extracted from the picture of a zebra

3.3. Experimental choices and design

This section presents the experimental choices, clarifies the tests conducted, and lists the research questions. The proposed experiments are organized to answer the four research questions, whose mapping is reported in detail in Table 1, and the results are discussed in Chapter 4.2. All the tests share the same experimental setup; the baseline backbone is **ResNet50** [15] using the PyTorch implementation and the weights of **IMAGENET1K_V1**. To conduct this experiment, the following ResNet50 layers, L , are selected to extract activations.

$$\mathcal{L}_{\text{ResNet50}} = \{\text{layer1.2, layer2.3, layer3.5, layer4.2}\} \quad (4)$$

The architecture of a ResNet50 implemented in **torchvision** is structured in four sequential stages (**layer1**, **layer2**, etc). Each sequential stage is composed by many bottleneck blocks, and are referred to using the notation **layerX.Y**, where **X** denotes the **X**-sequential block, and **Y** denotes the **Y**-bottleneck layer. This means that, as shown in the list below 4, hooks are set to the last step of each sequential step. These layers are chosen at increasing depths (one early, two in the middle, one late), so that the CAV's quality and other metrics could be evaluated at different stages of the feature-extraction pipeline. These hooks are chosen because they are the last bottleneck steps before forward propagation to the next level, thus representing the tensor that is propagated to the next sequential stack. Consistently, the hook is chosen on the block's module output: after batch normalization, the residual sum, and the ReLU activation.

Experiments are explored under different aspects, such as subset sizes, correlation coefficients, and comparison against a reference baseline (manually curated datasets or random vectors).

Another experimental choice that needs clarification is the Vision Language Model (VLM) used in the first experiment to assess the presence of the concept in the dataset. The reason for implementing a VLM is

to assess whether a concept is present or not in a picture. In the particular case of this experiment, the **Qwen/Qwen2-VL-2B-Instruct** [34] model is adopted. To perform a similar task, in particular to assess the similarity between pictures in the same class, a pretrained **ViT-B-32** [11] CLIP model is chosen. Because CLIP models are subject to the same biases as VLMs, this approach is also at risk of circular dependency. Both implementations are based on PyTorch.

The chosen parameters, cited for reproducibility purposes, are listed below. To extract the concept dataset, no limits are placed on the original dataset. Limiting the number of pictures per concept would accelerate the extraction process, but could result in the extraction of non-representative samples. The expected number of generated images is 100-120 per class and domain; this value is chosen after a test to determine the minimum number of samples required to obtain a stable CAV, consistently with the tests conducted in the papers of Astolfi et al. [2], where the minimum number of samples required for the representation to stabilize is evaluated. After reaching the desired number of concept images, the extraction process steps to the next class. The number of samples is chosen after testing the CAV stability. All experiments have been run on **Kaggle** using GPU **T4 x2** and on **Colab** using **A100** GPU. The threshold for the extraction module is 0.25, which represents the confidence required by **SAM3** to consider the concept present in the picture.

All the experiments, except the ablation studies, are conducted with a baseline setup; this setup consists of using the **sliding_window** cropping policy, the **vanilla** dataset, and the **multi-class** set of images.

The textual input used to extract the dataset of concept images is proposed by **ChatGPT** and is structured to be as complete as possible, while maintaining a short proposition structure as suggested by the **SAM3** specifics. An example of a prompt for a concept, can be the one of the **striped** concept, which is: "**striped pattern**", "**stripes**", "**striped texture**"; for more complex concepts, such as **veined**, the proposed prompt is "**veined pattern**", "**veins**", "**veined texture**", "**veined object**", "**marble texture**", "**leaf veins**", "**veined marble**", "**organic vein pattern**".

3.4. Concepts and datasets

In this section, the concepts used in the experiments will be explained, along with a brief description of the datasets used. Right below the set of concepts chosen to be extracted with this framework will be presented, and then compared with the ground truth dataset: **striped**, **chequered**, **dotted**, **wood**, **water**, **braided**, **bubbly**, **fibrous**, **veined**. These concepts are selected as they belong to different categories, enabling the exploration of behaviors of different types of concepts. In particular, the following categories are considered to be of interest and are listed here: base concepts, such as **striped**, **chequered** and **dotted**, are chosen consistent with other works, as they are considered to be easy concepts to be found in many contexts; concepts of materials, with **wood** and **water**, to assess the capacity of our pipeline to automatically extract crops based on their materials; concepts with characteristic attributes and shapes, such as **braided**, **fibrous**, **veined** and **bubbly**. There already exist manually curated datasets that will serve as ground truth for these experiments and can be used to compare the extracted datasets, as well as artificially generated datasets for further ablation studies.

As mentioned before, the choice of the classes in the starting domain to look for the concept is made using **ChatGPT-5**; this approach leaves room for future work, as well as for the automatic selection of concepts to extract using LLMs.

The following datasets are used to extract the concept classes considered ground truth for comparison with automatically extracted concept datasets.

- **Zero-shot Text-to-Image (T2I) Dataset**: "a dataset of synthetic visual concepts generated using zero-shot Text-to-Image (T2I) models, designed to support research in concept-based Explainable Artificial Intelligence (XAI)". [2]
- **Describable Textures Dataset (DTD)**: as stated in the official website and in the paper [8], the Describable Textures Dataset is: "an evolving collection of textural images in the wild, annotated with a series of human-centric attributes, inspired by the perceptual properties of textures".

As a pool of classes from which to extract the dataset from **ImageNet** [10] is chosen; as stated on the official website and in the original paper, it "is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images".

Finally, obviously, the automatically extracted dataset is used for the experiments assessing its quality. Examples of images extracted for the concept classes of **braided** can be seen in the pictures below 10, as well as those extracted for the class **striped** 11.

braided / multi-class [sliding_window]



Figure 10: Random samples of pictures containing the concept "braided".

striped / multi-class [sliding_window]

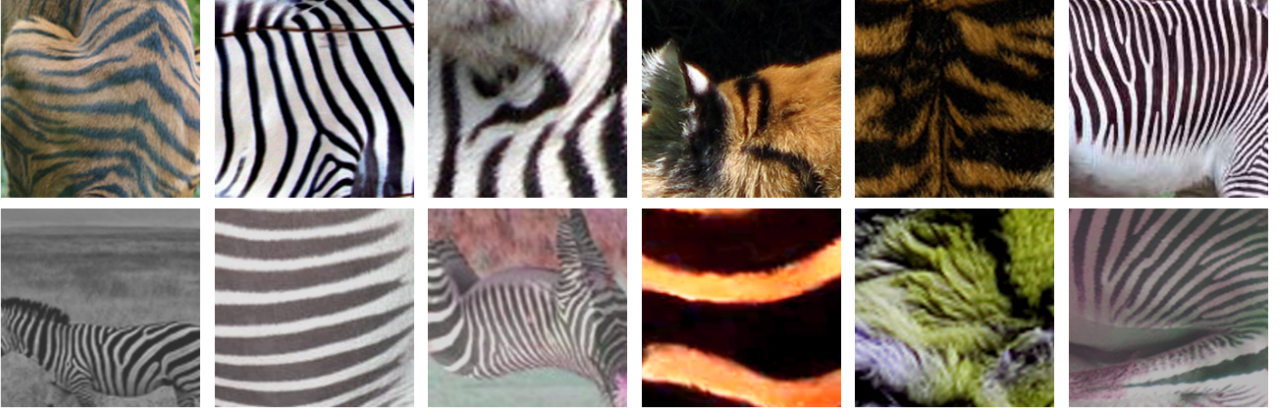


Figure 11: Random samples of pictures containing the concept "striped".

3.5. CAV Training

In the previous sections, Concept Activation Vector (CAV) has been presented as "a vector representation of a concept found in the activation space of a layer of a Neural Network" [21]. Here, the chosen algorithm adopted to train the CAV is the Difference of Means (DoM), unlike the one proposed by the original paper, logistic regression; in fact, as shown by Martin [33], the difference of means training algorithm has been proven to produce more consistent CAVs; Difference of Means has been adopted also by De Santis et al. in their works on Visual TCAV [27]. With regard to the proposed pipeline, the activations at layer l are referred to with the notation $\phi_l(x) = \text{StandardScaler}(\text{GAP}(f_l(x)))$ after the application of the Global Average Pooling (GAP) and the standardization. Centroids of the concept set D_c^+ , and of the negatives D^- are referenced with the following notation:

$$\mu_c^l = \frac{1}{|D_c|} \sum_{x \in D_c} \phi_l(x) \quad (5)$$

$$\mu_-^l = \frac{1}{|D^-|} \sum_{x \in D^-} \phi_l(x). \quad (6)$$

The CAV resulting from these centroids is obtained using the difference of means for the concept c at layer l as the unitary vector linking the two centroids, as shown in the following equation.

$$v_c^l = \frac{\mu_c^l - \mu_-^l}{\|\mu_c^l - \mu_-^l\|_2}. \quad (7)$$

Also, consistent with the Visual-TCAV paper and as anticipated above, a Global Average Pooling (GAP) is applied to the activations, as the main focus is not the localization of activations but the features activated by a given concept at a chosen layer. GAP is defined as follows:

$$\text{GAP}_k = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W X_{i,j,k} \quad (8)$$

where H and W are the height and width of the layer, meanwhile X represents the activation at the coordinates (i, j) , for channel k . For the experiments conducted for the third research question (RQ3), layers are summarized using two spatial poolings methods instead of only adopting GAP, which is used together with global max pooling.

Different backbone models are tested to conduct ablation studies and extract activations at different layers. First, **ResNet50** is considered the baseline model, followed by **VGG16** [28] and **InceptionV3** [31], all of them using the **PyTorch** implementations.

Before any CAV computation, activations are standardized by features using a **StandardScaler**. The scaler is fit on the shared negative pool and then applied to the CAVs.

To train the CAVs, negative examples are drawn from a random pool of images from **ImageNet**.

3.6. CAV Comparison and metrics

To compare the extracted CAVs $v_c^{l,\text{automatic}}$ with the ones computed from the manually curated dataset $v_c^{l,\text{manual}}$, the canonical cosine similarity score is adopted. Cosine similarity is a metric used to compare two vectors in terms of direction, regardless of their magnitude, measuring their similarity in a space; mathematically, its formula is described below:

$$\cos(u, v) = \frac{u \cdot v}{\|u\| \|v\|} \quad (9)$$

To test the extraction pipeline, cosine similarity is used in two ways: first, to measure intra-similarity within the same concept class; this metric helps evaluate the similarities between samples within the same concept class. High intra-similarity means that different subsets of the same concept class yield the same interpretations. The second way cosine similarity is used is to compare CAVs across datasets or concept classes, thus gauging the similarity between vectors of the same concept trained on different datasets (automatic vs. manual) or between different concepts.

When testing stability and other properties of the CAVs, each CAV is computed on a sample of the pool (bootstrap). This process is performed B times, without replacement, both for the positive set D_c and for the negative set D^- , forming an ensemble of CAVs. Stability is computed as the cosine similarity between each CAV.

4. Experimental Analysis

In this section, the experiments conducted to evaluate the automatically extracted dataset will be discussed, presenting the chosen approach, the main results, and takeaways.

4.1. Main experiments

This chapter briefly explains the main experiments conducted and maps them to the research questions they aim to answer. The table below 1 shows the mapping between the experiments and the research questions.

The main experiments, except for the ablation studies, are run using the same base pipeline explained in the section 3.3. Ablation studies are distinguished from the main experiments because they isolate the behavior of individual components to understand which components of the automatically generated concept dataset pipeline most influence performance and quality.

4.2. Results

4.2.1 Intrinsic dataset evaluation

This first experiment is conducted to assess the presence of the concept in the extracted dataset, along with other metrics; its goal is, in fact, to determine whether the generated dataset contains valid images that actually

Table 1: Mapping between research questions, experiments, and metrics. Unless they target the component under ablation (RQ4), all experiments share the baseline setup described in Section 3.3.

RQ	Experiment	Metric
RQ1	1. Intrinsic dataset evaluation	VLM-as-a-judge alignment (mean, std, P@0.5, P@0.7, share <0.25); intra-class CLIP cosine coherence; repetition rate, near-duplicate pairs, participation ratio, dataset size N .
RQ2	2. CAV stability and robustness	Pairwise cosine between CAVs re-trained on B bootstrap resamples; accuracy of the classifier; knee-elbow test.
RQ3	3. Auto-vs-manual similarity	Cosine similarity per layer, between the CAVs of the extracted set of images, and the baseline set of images; intra-class cosine similarity.
RQ4	4. Ablation and SOTA comparison	Intrinsic metrics and in auto-vs-manual CAV similarity after the variation of various components (crop policy, domain, augmentation, backbone, prompt); comparison with the synthetic generated dataset.

show the concept. After a first visual inspection, the quality of the concept dataset is measured using various metrics. First, chosen metrics are computed with the aid of a VLM, using the **Qwen/Qwen2-VL-2B-Instruct** model. More specifically, the metrics computed are the mean alignment and the standard deviation, using a prompt to identify the dominant pattern/texture in the image. To assess the presence of the concept in the images, the VLM is prompted using the following query:

Look at this image. Is '{concept}' the dominant visible pattern? Answer 'yes' only if the conditions hold; otherwise answer 'no'.

and uses a threshold of 0.25; below this, the concept is considered clearly absent in the picture. The VLM judge is prompted in logit-scoring mode, thus it does not produce a textual free-form answer, but returns the probability of the image of containing the target concept; mean alignment and standard deviation are then computed over the extracted images.

Also, the VLM alignment below 0.25 is measured. This metric is interesting when we want to check if augmentation is producing weak images with the concept; also, it helps understanding the confidence in the presence of the concept in the extracted set. Other metrics measured are the precision thresholds of 0.5 and 0.7. The repetition rate percentage is computed as well to evaluate whether the model is extracting many pictures from the same starting image, or, more generally, images that are too similar; the number of near-duplicate pairs is computed for the same reasons. Finally, the participation ratio is computed, providing a sense of the active degrees of freedom and a simple count of the dataset dimension, giving a sense of the dataset's volume. All these metrics have been computed using a CLIP model, specifically a **ViT-B-32** model.

In practice, this experiment is conducted, obtaining all the metrics described above for all concepts, starting domains, augmentations, and vanilla extractions, and all cropping policies. The obtained metrics can be inspected in the table 2; due to space constraints, only a sample class is reported. The chosen concept class is the **striped** class, consistent with the rest of the experiments.

Beyond evaluating the dataset, a comparison of the different extraction policies (augmentation vs. vanilla, starting domain, etc.) is conducted to assess which one is most promising in terms of intrinsic evaluation.

Some conclusions that can be drawn from these results are that, for the single and multi-class extractions, the VLM-as-a-judge alignment of the automatically extracted datasets is comparable to the manual baseline (DTD), with also a comparable standard deviation. Also, it is very clear that the properties of the extracted dataset strongly depend on the starting domain; as it can be observed, the concept-classes extracted from the random classes are the worst performing ones, with mean alignment below chance and standard deviations higher than those of the single or multi-classes. The multi-class also introduces greater variation between images, as it draws from a more diverse starting domain. This does not necessarily mean the pipeline can't extract crops from random images, but rather that the starting classes need to contain a sufficient number of crops with that concept. One important note about the concept classes is that to extract the target number of crops, the pipeline has to inspect many more pictures; this results in higher processing time, and leads to a weaker and less consistent presence of the concept. Some more findings on this ablation will be reported in the ablation studies section.

Apparently, for some concepts, the automatically extracted dataset yields higher mean VLM alignment values than the manual baseline. This metric does not imply that the extracted dataset is of higher quality than the ground truth dataset; rather, it indicates that the VLM model has identified the concept more frequently as predominant in the processed images. This is probably related to two factors: the first is the potential

Class	Crop policy	Align. Mean	Align. Std	%< 0.25	P@0.5 (%)	P@0.7 (%)	Rep. (%)	Dup. pairs	PR	N
Manual baseline (DTD)										
Manual		0.872	0.077	0	100.0	95.0	0.574	41	15.4	120
Vanilla										
Single	bbox	0.914	0.036	0	100.0	100.0	0.202	10	23.9	100
	center_mask	0.915	0.025	0	100.0	100.0	0.020	1	18.2	100
	largest_bbox	0.917	0.034	0	100.0	99.0	0.343	17	24.9	100
	sliding_window	0.892	0.042	0	100.0	100.0	0.020	1	9.9	100
Multi	bbox	0.878	0.123	1	97.0	95.0	0.182	9	11.5	100
	center_mask	0.898	0.061	0	99.0	98.0	0.000	0	13.4	100
	largest_bbox	0.895	0.094	1	98.0	97.0	0.384	19	10.3	100
	sliding_window	0.845	0.123	0	95.0	93.0	0.141	7	10.6	100
Random	bbox	0.566	0.208	10	65.0	30.0	0.000	0	26.5	100
	center_mask	0.516	0.226	13	55.0	25.0	0.000	0	20.4	100
	largest_bbox	0.562	0.216	10	64.0	31.0	0.000	0	26.8	100
	sliding_window	0.576	0.203	8	68.0	32.0	0.000	0	24.0	100
Augmented										
Single	bbox	0.883	0.057	0	100.0	98.0	0.061	3	17.2	100
	center_mask	0.889	0.039	0	100.0	100.0	0.061	3	19.3	100
	largest_bbox	0.896	0.047	0	100.0	99.0	0.040	2	20.1	100
	sliding_window	0.844	0.095	0	97.0	96.0	0.121	6	14.9	100
Multi	bbox	0.826	0.144	1	92.0	86.0	0.020	1	16.9	100
	center_mask	0.838	0.115	0	96.0	90.0	0.061	3	18.1	100
	largest_bbox	0.832	0.145	0	92.0	88.0	0.020	1	15.9	100
	sliding_window	0.769	0.163	0	89.0	78.0	0.081	4	15.4	100
Random	bbox	0.590	0.194	6	67.0	35.0	0.040	2	28.7	100
	center_mask	0.470	0.191	14	42.0	13.0	0.000	0	32.5	100
	largest_bbox	0.551	0.203	8	56.0	28.0	0.081	4	28.9	100
	sliding_window	0.537	0.205	11	60.0	24.0	0.061	3	27.2	100

Table 2: Intrinsic evaluation of the concept "striped"

circularity between the extractor and the VLM, that might introduce a bias in the evaluation, as discussed in the limitations section. The second reason could be the discarding of the background and surrounding context by the crop performed during the extraction process. The manual DTD set contains deliberately broad intra-class variation. Results from the intrinsic evaluation experiment, such as the VLM-as-judge alignment value should not be interpreted as a metric of a higher quality of the dataset with respect to the baseline, but rather as the evidence of the target concept’s presence in the extracted images.

4.2.2 CAV stability and robustness

The goal of this experiment is to assess the stability and robustness of the CAVs generated from the dataset extracted using the proposed pipeline; these tests are also performed to calibrate the parameters, volumes, and thresholds to obtain a CAV well capable of correctly describing a concept. Various experiments are conducted to address the macro questions posed in the first subsection of this chapter. The first test assesses whether the concept dataset enables the generation of stable CAVs independently of the negative/random dataset. Between all the chosen concepts, a single concept is selected, then, for 20 times, the experiment is repeated, regenerating the CAV the same number of times; each CAV is then compared with the others, using the `cosine` similarity pair-wise with all the other CAVs obtained. This process is repeated for all concepts in the pool, and the results are then averaged. The method used to compute CAVs is the difference-of-means method. In the figure 12, the results for the striped class are reported. For this and other concepts, CAVs are stable on the four layers; the average mean on the layers is 0.960, meanwhile the mean of the third layer (`layer3.5`) is 0.975.

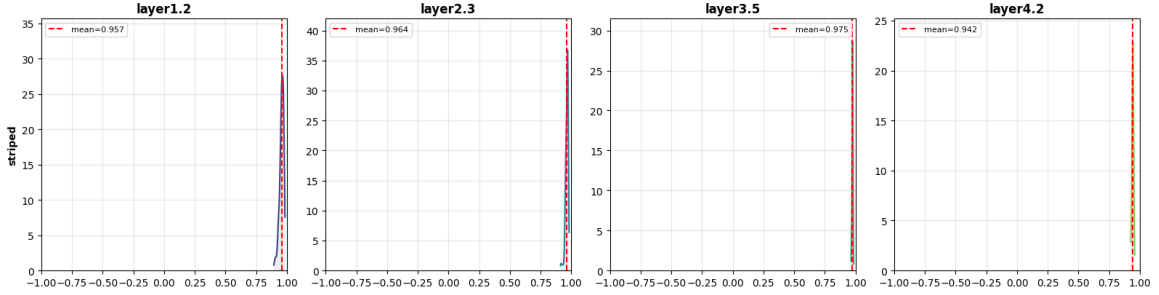


Figure 12: Plots of the average cosine similarities, and their distribution, computed pair-wise over different sampling of the negative images. The chosen layers are those described in the previous sections for the ResNet50 network.

Another test conducted is the analysis of stability with respect to the dataset’s dimensions, or, in other words, how much a reliable CAV depends on the number of positive samples. This test measures two dimensions: the intra-trial cosine similarity between CAVs trained on different samples, and the classifier accuracy, or the CAV’s ability to correctly separate positive from negative samples. As shown by the aggregated results in the figure 13, the CAVs generated for the aggregated concept start to reach a plateau in the similarity metric depending on the sampling of negatives, around 75 samples.

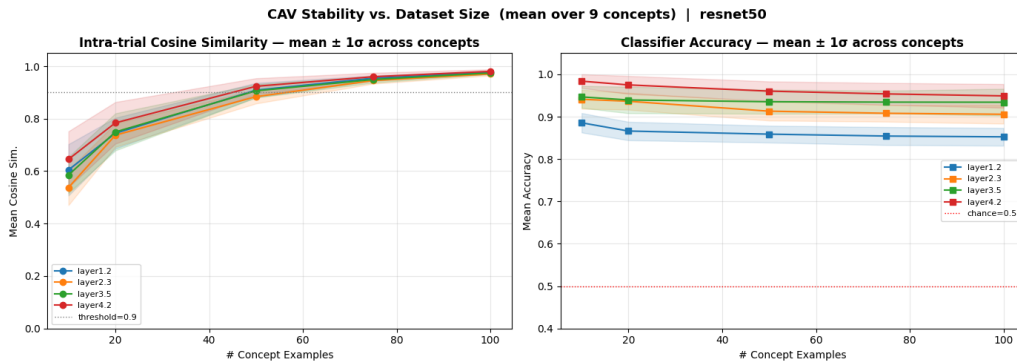


Figure 13: These plots show metrics aggregated from all the concepts considered. Intra-trial cosine similarity between different samples picked from the same concept image set, showing the values of the CAV similarities computed starting from the activations of different layers of a ResNet50 network. On the right, the graph shows the classifiers’ accuracy.

To assess the number of samples to consider when training the CAVs, the knee-elbow test, is performed. The knee-elbow method consists of a technique to find the "knee" (or "elbow") of a curve, or the number of samples, clusters, or any other parameters, above which the relative costs of increasing that parameter are no longer worth the increase in performance. This technique is adopted in this case to suggest the number of samples to successfully train a stable CAV. This technique suggests that, for most concepts, after 75 samples, the CAVs reach a plateau, meaning they are stable enough. This result can be observed in figure 14. CAVs generated from manually extracted image sets reach stability around 75 samples, too. The experiments on CAV stability across all concepts show that the minimum number of samples required to guarantee a stable, consistent CAV is 100.

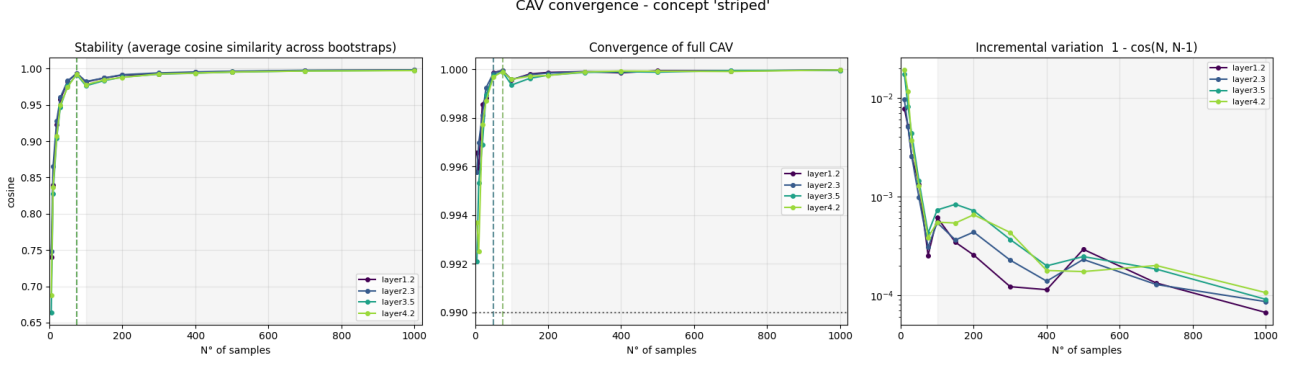


Figure 14: Convergence of CAVs for the striped class

A gauge on the evolution of CAVs intra-similarities can also be assessed in the figure 15; in this case, 30 independent trainings of CAVs for the same concept are performed for each layer.

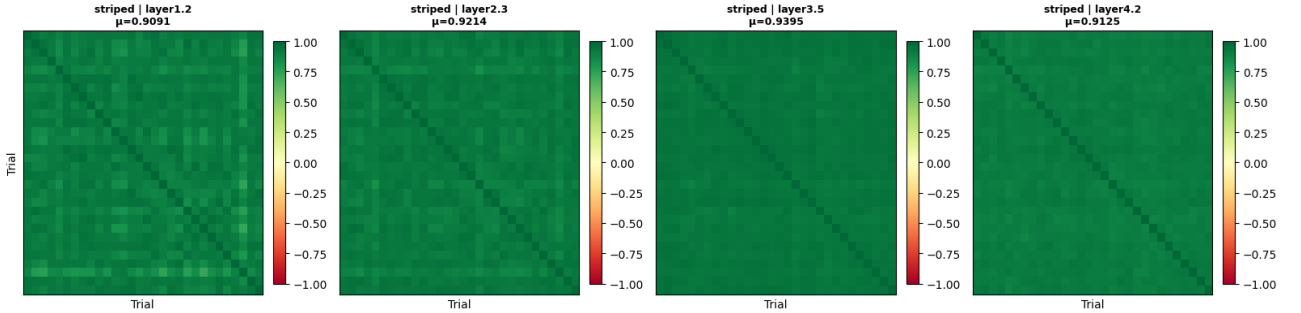


Figure 15: Intra-similarities between 30 independent trainings per concept and layer.

4.2.3 Auto-vs-manual CAV similarity

This experiment assesses the similarity between CAVs obtained from the automatically extracted dataset and those from the manually curated dataset at different levels of the CNN pipeline. The aggregated results from the CAVs for the selected concepts can be inspected in figure 16.

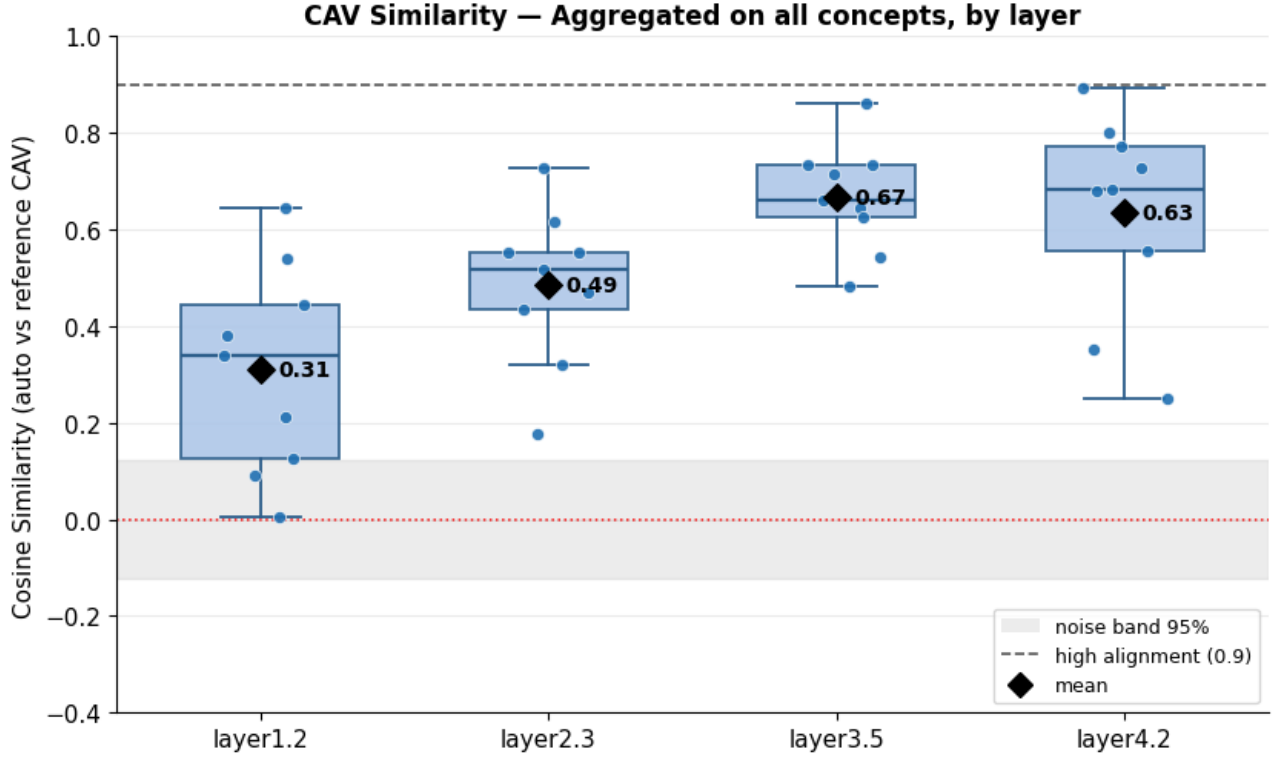


Figure 16: Cosine similarity evolution per layer, between CAVs trained on automatically extracted concept images, and CAVs trained on manually curated sets of images. The high-alignment serves as a visual reference, while the noise band can be derived from the cosine similarity of two random vectors for layer1.2, which represents the biggest noise band, thus the most conservative also for the other layers.

Figure 17 shows the evolution of the cosine similarities of the CAVs divided per layer; the best performing layers are the last ones (third and fourth), with textures being overall well extracted in the third and fourth layers. Materials concepts, water and wood concepts, are performing worse in the fourth layer with respect to the third; this behavior might be different from other patterns, as it is not associated with a visual pattern but with a material. Another interesting key point of this experiment is that the augmented dataset performs worse when evaluated, but better when comparing the CAVs. This result will be further discussed in the dedicated ablation study.

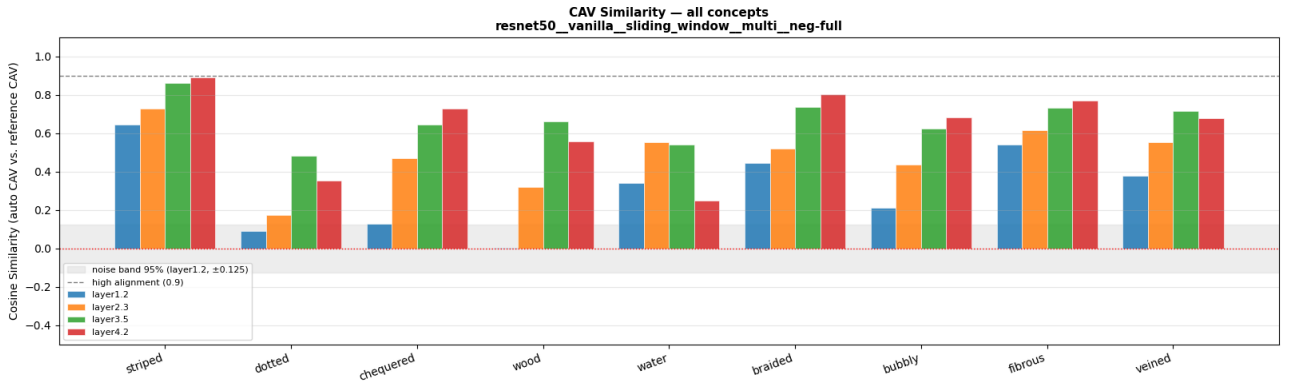


Figure 17: Evolution of the cosine similarity metric, per layer and concepts. Noise band and high alignment follow the same reasons as Figure 16

4.3. Ablation Studies

To conclude the chapter on the experiments, some ablation studies will be presented; the goal is to test the pipeline with different backbones, parameters, and policies.

4.3.1 Starting domain

This ablation study investigates the dependence of the extracted dataset on the starting domain. The pipeline permits the extraction of real crops, to avoid hallucinations, but requires that the concept is in the starting pictures. As observed in this subsection, this approach can be preferred when the starting domain is rich or when the concept images need to be realistic. The choice of the domain is the factor that most influences the results, compared to the other ablation studies. The coherence and quality of the extracted dataset depend on the presence of the concept in the source domain; this can be assessed by evaluating the extraction on random classes. Moreover, random-classes are chosen to assess the capability of the extraction pipeline to extract the concept from randomly chosen classes, with, in many cases, poor presence of the concept; results show that the extracted dataset often does not reach the required amount of pictures before running out of samples in the starting pool, as well as suggesting worse performances in all the metrics: the number of generated samples is very low for certain classes, as can be observed in the **asparagus** concept class, extracted to conduct the comparison with the synthetic dataset (figure 25). The VLM mean alignment is always lower too; its value, averaged across all concepts, has a $\Delta = 0.27$ with respect to the averaged single-class VLM mean alignment, likely also due to the high variation and lower presence of the target concept in the starting domain. In terms of intrinsic evaluation, Figure 18 shows the performances of the concepts.

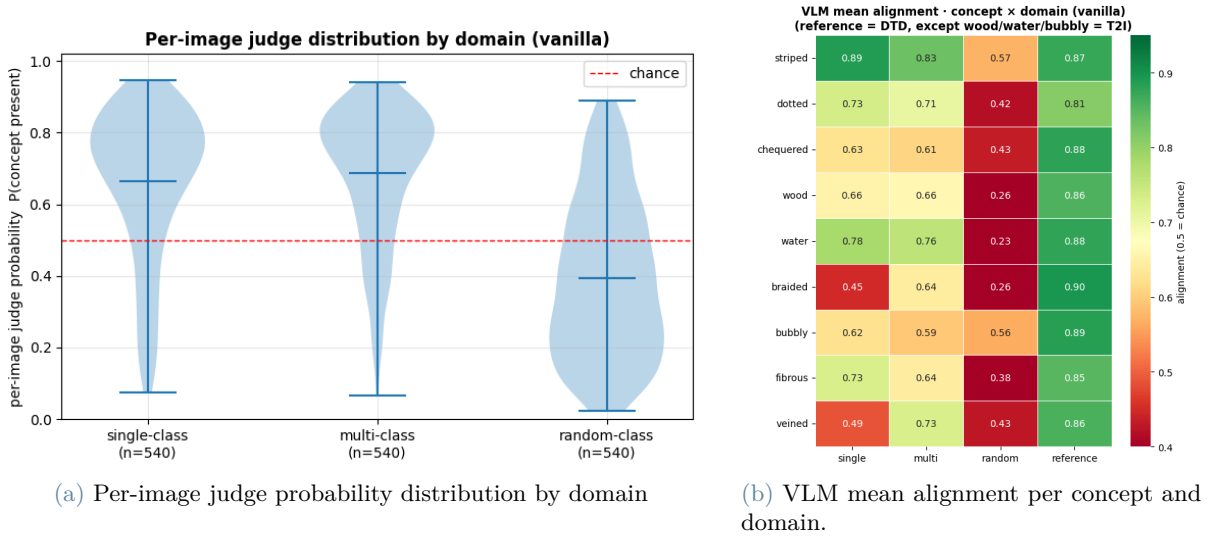


Figure 18: Intrinsic evaluation of the extracted datasets.

4.3.2 Cropping strategies

This experiment compares the metrics of datasets obtained by cropping the same concept from different images using different cropping policies. To this extent, intrinsic evaluation and CAVs comparisons' experiments are conducted. By inspecting the results of the intrinsic evaluation experiment, it is clear that the **largest_bbox** and **center_mask** are maximizing the alignment metrics, while **sliding_window** is showing lower performances and higher variance, meaning that, by an intrinsic evaluation point of view, the VLM could need some context, background and padding around the object or concept, to better classify the concept as present or absent. In the case of the comparison between the extracted CAVs, the **sliding_window** cropping policy appear to systematically produce CAVs that are the most similar to that of the original manually curated dataset; in the whole experiment, the maximum similarities values have been obtained by the **sliding_window** policy, with high similarity metrics on every layer, and lower dependence on the selected starting domain. This ablation study shows that tight crops better capture the concept, whereas padded or larger crops are better recognized by VLMs probably due to the presence of more context and background. The measured gap between cropping strategies in terms of VLM alignment is presented below. The values of the VLM alignment for each cropping strategy are **bbox**: 0.66, **center_mask**: 0.70, **largest_bbox**: 0.67, **sliding_window**: 0.65. In all the cases, the variance in the results is minimal.

4.3.3 Augmented vs. Vanilla

This experiment compares performance between **augmented** and **vanilla** datasets. To this extent, the dataset is evaluated intrinsically, and CAVs' similarities are tested. Regarding the results of the intrinsic evaluation,

augmentation modifications increase the number of samples and the dataset’s diversity, but often reduce the mean VLM alignment, though this effect is only slight. This can be observed systematically across all the concepts, with a decrease in **precision@0.5** and **precision@0.7**, and an increase in the participation ratio for all the concepts in the multi-class starting domain. The decrease in mean alignment is mainly observed in the concepts of materials, where a shift in the hue distribution may influence the VLM’s decision. The measured gap between augmentation against vanilla, in terms of VLM alignment, is: 0.70 for the vanilla datasets, against 0.65 for the augmented dataset, when comparing the whole vanilla extractions against the whole augmented extractions, resulting in a $\Delta = 0.05$ gap. In terms of CAVs’ similarities, CAVs trained on augmented concepts are systematically more similar to those in the manually curated dataset; this is shown in Figure 19. In the Figure 19, a summary of the ablations between augmentation and vanilla sets, as well as the one between cropping policies, is presented.

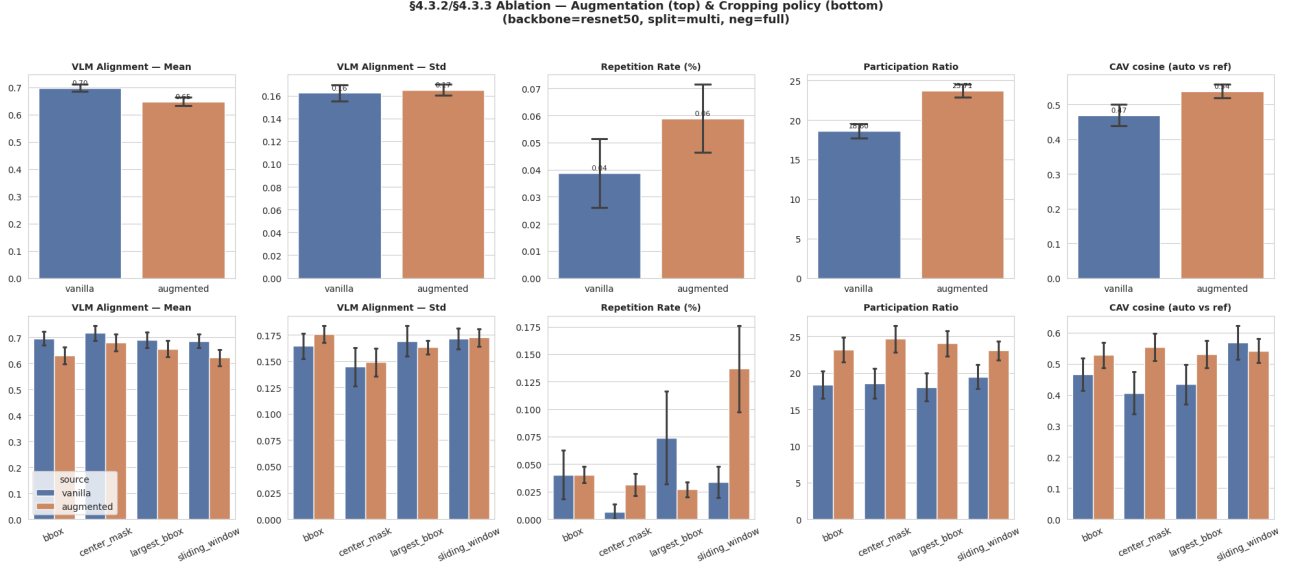


Figure 19: Brief summary showing changes in performances between the two ablation studies presented above: cropping strategies and vanilla dataset VS augmented dataset

4.3.4 Backbones

This experiment is conducted to compare three backbones: **Resnet50**, **VGG16**, **InceptionV3**; these models are chosen in line with other works [4]. The purpose of this ablation study is obviously not to compare models one with the others but to gauge the evolution of the activations along the feature extraction pipeline, and to evaluate the behavior of the proposed pipeline more generally on different models; a strict comparison between the models can’t be done as the models don’t share the same architecture, and, even though the extraction layers are chosen to extract activations at the beginning, at the end, and at two hooks in the middle, the pipeline works very differently between these models. Coherence between the models can be observed in the reported experiments, meaning that the behavior observed on the CAVs generated using **ResNet50** is not necessarily dependent on the architecture; in fact, we observe that similarities between CAVs grow closer to deeper layers, with the exception of **InceptionV3** which lowers its performance on the last layer. Even though high similarities can be observed in the intermediate layers of InceptionV3 and ResNet50, this last model seems to be the most stable backbone among those tested; this ablation study is conducted simply to prove robustness. Figure 20 and 21 show the metrics of different networks on results aggregated by concept.

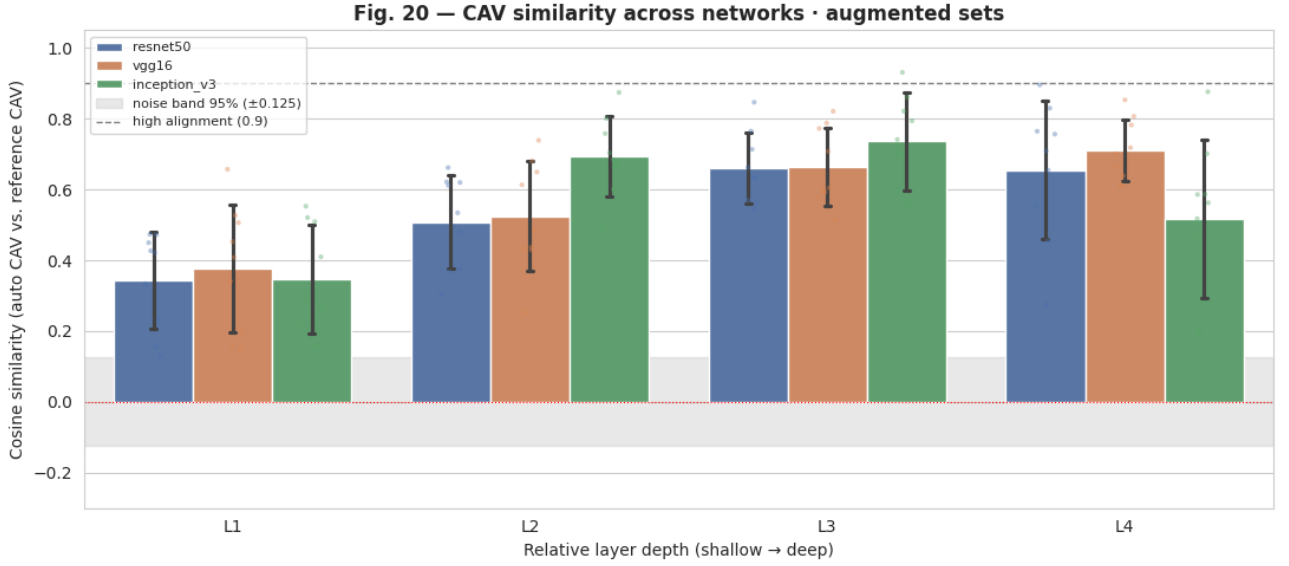


Figure 20: Similarities between CAVs computed using activations from different networks, at different layers, between CAVs generated on automatically extracted sets and manually curated ones. Metrics are aggregated across all concepts and tested on augmented image sets.

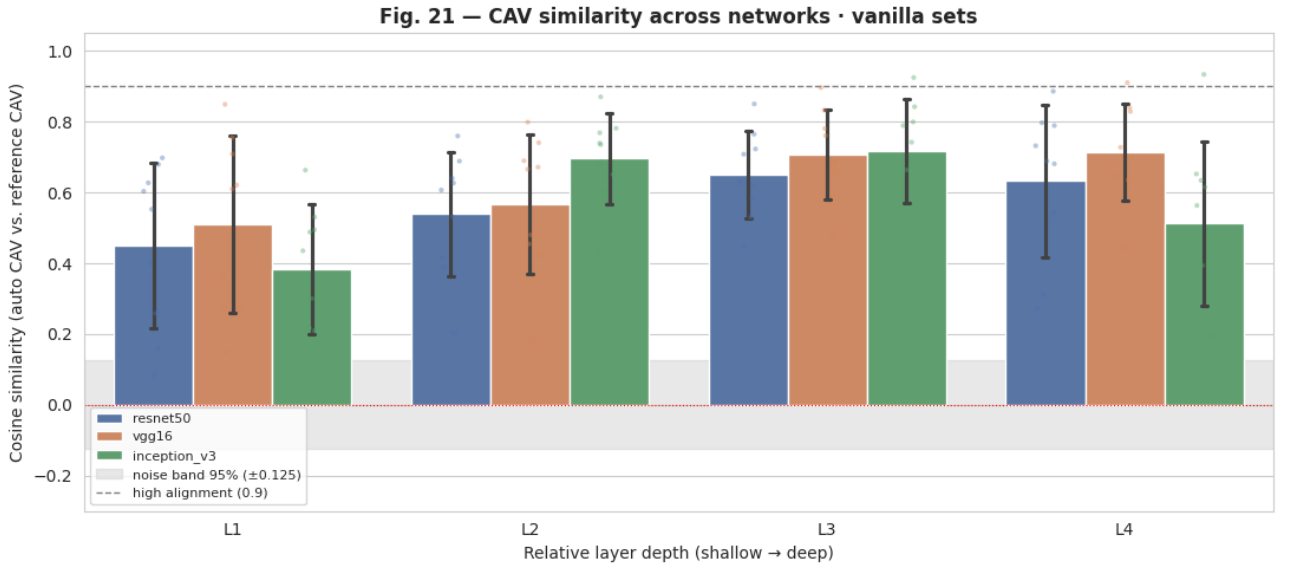


Figure 21: Similarities between CAVs computed using activations from different networks, at different layers, between CAVs generated on automatically extracted sets and manually curated ones. Metrics are aggregated across all concepts and tested on the vanilla image sets.

4.3.5 Prompt Variations

To gauge the relevance of the prompts entered into SAM3, the proposed extraction pipeline is tested with various textual inputs. Three variations of the original prompt are tested, following different approaches, trying to investigate different behaviors of the extraction pipeline depending on the prompted input:

- **Bare_term**: the prompt consists of a single word, describing the searched concept; it consists of the minimal bare keyword, setting the least limits. Examples of prompts are: **stripes**, **dots**, **checkerboard**, **wood**, etc.
- **Surface_phrase**: the prompt consists of a brief phrase looking for the surface concept. Examples are: **a striped surface**, **a dotted surface**, **a chequered surface**, etc.
- **Texture_phrase**: the prompt consists of a simple phrase looking for the texture concept. Examples are: **striped texture**, **dotted texture**, **chequered texture**, etc.

The dataset obtained using these prompt variations is extracted using the **multi-class** options, without augmentation, and using the **sliding-window** cropping policy. These variations are tested against the base dataset by evaluating the intrinsic quality of the image set, and the cosine similarity of the CAVs against the baseline and the dataset extracted with the base prompt. Before moving to the intrinsic evaluation of the extracted dataset, the number of generated images needs to be analyzed: an observation is that in some cases, switching from one prompt to the other strongly influences the quantity of generated crops; specifically, in terms of the number of extracted images, the amount of crops is appreciably below the expected number for all the prompt variations. Results of the intrinsic evaluation, and comparison with manually curated datasets and baseline are reported in the table 3 and figure below 22.

Table 3: Metrics averaged on all concepts, per concept variations, compared with the baseline prompt described in the previous sections. N stands for the average number of samples per concept

Metrics	Bare-term	Surface-phrase	Texture-phrase	Baseline
VLM Alignment – Mean	0.700	0.669	0.717	0.687
VLM Alignment – % below 0.25	4.431	0.635	2.755	3.333
VLM Precision @ 0.5 (%)	87.992	73.333	85.782	82.222
VLM Precision @ 0.7 (%)	55.352	53.016	64.218	58.333
Repetition Rate (%)	0.058	0.138	0.092	0.034
Num. Near-Duplicate Pairs	2.857	5.857	4.571	1.667
Participation Ratio	15.535	16.730	15.780	19.470
Dataset Size (N)	47	41	48	60

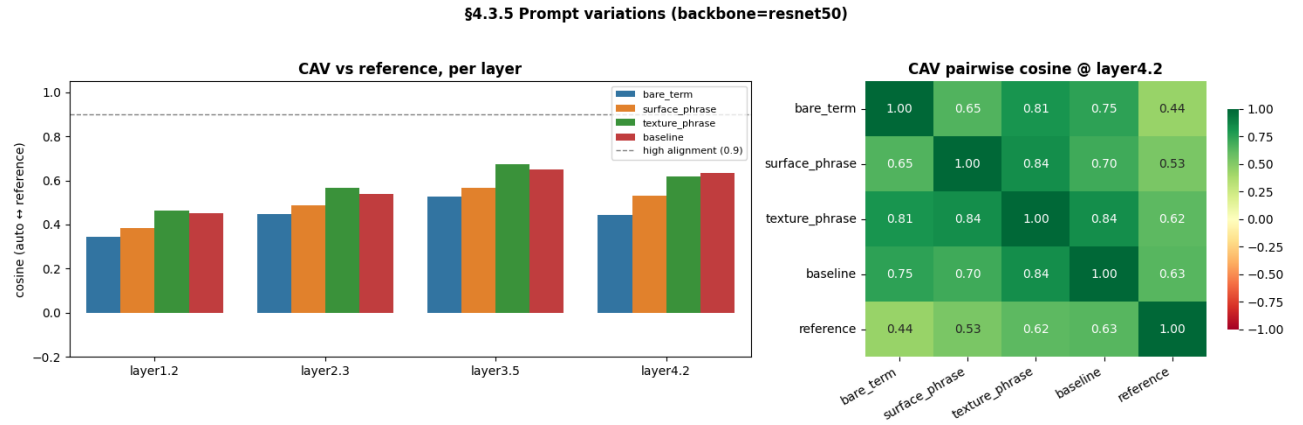


Figure 22: Comparison between different prompts variations (bare_term, surface_phrase, texture_phrase and manual baseline).

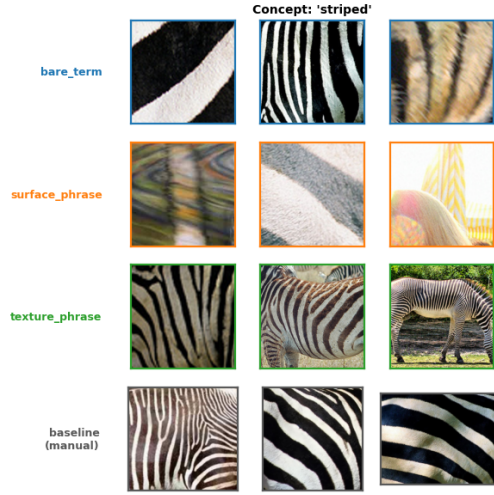
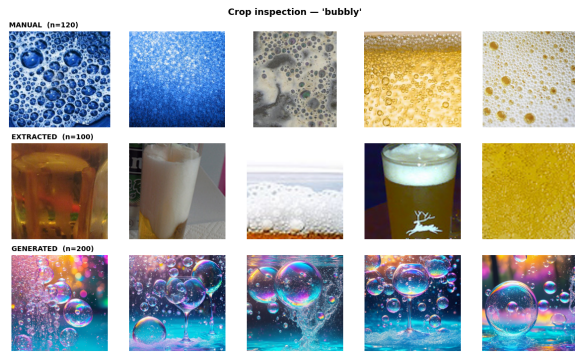


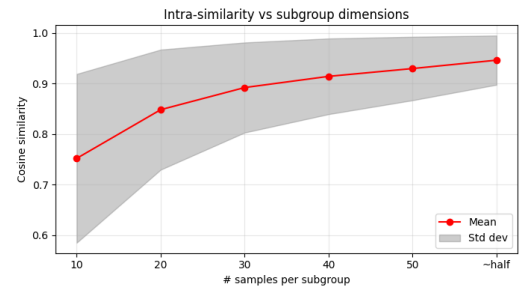
Figure 23: Samples of the different datasets extracted using different prompts, for the striped class.

4.3.6 Comparison with generated concepts

This experiment is conducted to compare two pipelines for automatically extracting a dataset of concepts. To compare these two pipelines, an extended dataset of 35 concepts is automatically extracted. In detail, the same three backbones are chosen, and centroids to define CAVs have been computed on the activations of the last three layers of the backbone models; differently from the previous experiments, to be coherent with the synthetic generation experiment, the same backbones are used (ResNet50, InceptionV3, VGG16), but using the **Keras** implementation. This ablation study measures intra-class cosine similarity across bootstrap samples and cross-class cosine similarity against manually curated image sets. The **ImageNet** classes to be used as starting domain have been chosen in the same way as it has been done to select the starting classes in the baseline pipeline, thus letting **ChatGPT** select classes that are likely to contain the concept. In many cases, choosing a class to extract crops from is non-trivial, as many concept classes appear only in specific classes; in some cases, **ChatGPT** suggests classes that aren't in the provided class list or don't match the prompt. One example of this limitation, discussed also in the limitations section, is the **asparagus** class, which does not explicitly exist in the **ImageNet** dataset. Extracting the **asparagus** concept from the classes of other vegetables in the **ImageNet** dataset yields to the extraction of only 17 crops after scanning over 4000 samples, with a success rate of only 0.425%, whereas for a few classes, no crops have been extracted. In the figure below, a glimpse of is shown 24



(a) Comparison between manually curated dataset, automatically extracted ones, and generated ones.



(b) Performances of the intra-class cosine similarity, computed on the last three hooks of the ResNet50 model, varying on the dimensions of the subset, as done in [4].

Figure 24: Comparison between text-to-image generation pipeline, and the proposed extraction pipeline

The experiment yields a mean cosine similarity of 0.61, but lower than the 0.77 obtained with the generated images. One difference between the two methods concerns the implementation of the CLIP model after extraction to filter out images that do not contain the concept. This framework does not implement that filter, thus the comparison is between an unfiltered dataset and a filtered one (Campi et al. [4]). It is reasonable to hypothesize

that a filtered dataset would lead to better results; this experiment is left to the implementation of the filter, as explained in the future works section. Results of the proposed pipeline, in terms of cosine similarities averaged on the last three layers of the three proposed backbones, for all the concepts, are the following: **all**: 0.61 ± 0.18 ; results on the datasets extracted belonging to the following subsets are the following: **Search** 0.65, **DTD** 0.61, **ImageNet** 0.59. Results in terms of subjects are the following: **Textures** 0.64, **Objects** 0.59. In general, objects and textures are performing almost identically, with a difference in the CAVs' similarities that can be considered noise ($\Delta = 0.05$).

In general, our approach generates CAVs that underperform with respect to the ones trained on the synthetic generated dataset by ~ 0.16 (~ 0.20 when using **DreamBooth**). The hypothesis here could be, that the gap could be partially reduced by the usage of CLIP to filter the domain, or by the choice of a better starting domain both to generate better and more crops (eg. not **ImageNet**, which has been used as the baseline); these tests have not been performed, and are to be considered as simple hypothesis to be evaluated in future works. The automatic extraction process could be preferable when the goal is to avoid hallucinations by extracting real textures, or in cases in which generation is not feasible. In general, the generation pipeline is more useful when dealing with small starting datasets, as the proposed extraction pipeline is strongly dependent on the starting domain.

5. Conclusions and Future Works

5.1. Summary of findings

This section reevaluates the contributions after the experiments have been conducted. The sets of images extracted using this pipeline can be compared to the baseline, even though they have been proven to have a strong dependence on the starting domain (RQ1). The CAVs trained on this dataset are stable with respect to the size of positives and negatives samples (RQ2). Regarding research question 3 (RQ3), results show that the similarity between CAVs for automatically extracted and manually curated datasets increases in the intermediate and deeper layers; also, textures and materials are better captured by different layers. For RQ4, the components driving quality and stability (crop, augmentation, negatives, backbone, prompt) are isolated, and the pipeline is directly compared with the generative pipeline of Campi et al. [4], on 35 concepts (objects and textures) and three backbones. Also, the proposed framework, and thus its contribution, has been studied to be reproducible, as parameters are documented and the experimental setup has been made explicit; in this sense, the complete mappings between concepts and starting domain classes have been reported in the appendix, for reproducibility purposes, avoiding dependence on a specific model of ChatGPT.

5.2. Limitations

A clear limitation of the proposed pipeline is associated with the need for a big dataset to extract the images from, to maximize the probabilities of extracting enough images to train the CAV; inherently, this leads to an increase in the time required to run the experiments, especially when looking for a concept in random classes. Another clear limitation of this pipeline is linked to false positives, or the extraction of images that do not evidently contain the concept, or that are cropped in a way that does not make the concept distinguishable or prominent in the picture. As already mentioned, a limitation introduced by VLMs and CLIP models is the risk of circularity in evaluating the content of extracted images, as VLMs and CLIPs share a similar approach to SAM3. This circularity should be further tested in future work, as the low VLM alignment in the random-classes extracted dataset could be due to a small sample size or to the VLM actually judging the concept as absent. To avoid relying solely on intrinsic evaluation, the CAV's evaluation can provide a more complete overview. Finally, **Qwen2-VL-2B** is a capacity-limited judge; its adoption is justified by its low computational requirements and the exploration results it provides; meanwhile, larger models could be tested in future work. The figure 25 below shows some examples of images that have been extracted without correctly containing the concept, in this case, of **asparagus**.



Figure 25: Some extracted pictures for the **asparagus** concept class, that are not containing any asparagus.

Also, only a few concepts have been tested, mainly concepts oriented to textures and visual patterns; more complex and abstract concepts, such as emotions or statuses, have been left out. Given the low number of concepts and target classes, this work is intended as an exploratory, descriptive study, as testing the null hypothesis would be underpowered with so few classes. Another intrinsic limit of the pipeline is that it is designed to work primarily with generic datasets, excluding domain-specific datasets such as medical or other technical images; this limitation stems from the segmentation model SAM3’s limited ability to extract concepts in technical domains. A final limitation concerns the use of ChatGPT as the class-starting domain selector, since the model can’t be fixed and is subject to newer checkpoints, retraining, etc. Reproducibility is preserved by fixing the classes and listing them in the appendix.

5.3. Future works

This work is intended as an exploratory study of the possible adoption of an automatic pipeline to extract concept images, and some possible expansions of the system have already been considered. A first approach could be to implement a vision-language model [3] or a CLIP (Contrastive Language-Image Pre-training) [24] as a filter; this could help limit the presence of out-of-concept images, as shown in Figure 25 and explained in the previous subsection. Another possible extension could be the implementation of a prompt expansion or refinement system based on a large language model to automatically propose the best prompt. Also, a possible expansion of the proposed pipeline could be an automatic class selector that defines the classes of the starting domain; a class selector based on LLMs could automatically select an initial set of classes to extract concepts from, further accelerating the extraction of the image set. Other concept classes could be tested too, especially those belonging to different categories, such as complex concepts and emotions; to further test the extension of the prompt formulation, it would also be interesting to automatically extract relevant concepts to test. Linked to the automatic extraction pipeline, another interesting possible implementation could be the addition of an automatic concept suggester module. It would be interesting to test the extraction pipeline on other models with different backbones, such as Vision Transformers. Furthermore, to better evaluate the proposed pipeline’s functionality, an interesting expansion could be to replace the VLM as the judge and implement a crowd-based system to assess the presence of a concept in an image. Expanding the framework to Vision Transformer or other non-CNNs architectures could also be interesting. In this work, CAVs are trained sampling random negatives from **ImageNet**; these images are mostly full-scene pictures. An interesting variation on this approach could be to test CAV training on cropped images, extracted using the same strategy as the extracted positives (e.g., a leave-one-out approach). This methodology could be useful for evaluating whether CAVs learn to distinguish between different resolutions and cropping strategies, or whether they are actually capable of learning from the concept sets. To complete this work, another extension could be to analyze the dataset for explainability purposes, comparing TCAV and Visual-TCAV scores consistencies with the scores obtained using the baseline sets.

References

- [1] Alejandro Barredo Arrieta, Natalia Díaz-Rodríguez, Javier Del Ser, Adrien Bennetot, Siham Tabik, Alberto Barbado, Salvador García, Sergio Gil-López, Daniel Molina, Richard Benjamins, Raja Chatila, and Francisco Herrera. Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI, December 2019.
- [2] Giacomo Astolfi, Matteo Bianchi, Riccardo Campi, Antonio De Santis, and Marco Brambilla. A Framework for Evaluating Zero-Shot Image Generation in Concept-Based Explainability. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3303–3311, 2026.

- [3] Florian Bordes, Richard Yuanzhe Pang, Anurag Ajay, Alexander C. Li, Adrien Bardes, Suzanne Petryk, Oscar Mañas, Zhiqiu Lin, Anas Mahmoud, Bargav Jayaraman, Mark Ibrahim, Melissa Hall, Yunyang Xiong, Jonathan Lebensold, Candace Ross, Srihari Jayakumar, Chuan Guo, Diane Bouchacourt, Haider Al-Tahan, Karthik Padthe, Vasu Sharma, Hu Xu, Xiaoqing Ellen Tan, Megan Richards, Samuel Lavoie, Pietro Astolfi, Reyhane Askari Hemmat, Jun Chen, Kushal Tirumala, Rim Assouel, Mazda Moayeri, Arjang Talattof, Kamalika Chaudhuri, Zechun Liu, Xilun Chen, Quentin Garrido, Karen Ullrich, Aishwarya Agrawal, Kate Saenko, Asli Celikyilmaz, and Vikas Chandra. An Introduction to Vision-Language Modeling, May 2024.
- [4] Riccardo Campi, Santiago Borrego, Antonio De Santis, Matteo Bianchi, Andrea Tocchetti, and Marco Brambilla. Towards Synthetic Concept Activation Vectors via Generative Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2745–2753, 2025.
- [5] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, Jie Lei, Tengyu Ma, Baishan Guo, Arpit Kalla, Markus Marks, Joseph Greer, Meng Wang, Peize Sun, Roman Rädle, Triantafyllos Afouras, Effrosyni Mavroudi, Katherine Xu, Tsung-Han Wu, Yu Zhou, Liliane Momeni, Rishi Hazra, Shuangrui Ding, Sagar Vaze, Francois Porcher, Feng Li, Siyuan Li, Aishwarya Kamath, Ho Kei Cheng, Piotr Dollár, Nikhila Ravi, Kate Saenko, Pengchuan Zhang, and Christoph Feichtenhofer. SAM 3: Segment Anything with Concepts, March 2026.
- [6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers, May 2020.
- [7] Hong-You Chen, Zhengfeng Lai, Haotian Zhang, Xinze Wang, Marcin Eichner, Keen You, Meng Cao, Bowen Zhang, Yinfei Yang, and Zhe Gan. Contrastive Localized Language-Image Pre-Training, February 2025.
- [8] Mircea Cimpoi, Subhransu Maji, Iasonas Kokkinos, Sammy Mohamed, and Andrea Vedaldi. Describing Textures in the Wild. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3606–3613, Columbus, OH, USA, June 2014. IEEE.
- [9] Gabriele Ciravegna. C-XAI: Concept-based Explainable AI, May 2024.
- [10] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.
- [11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale, June 2021.
- [12] Amirata Ghorbani, James Wexler, James Zou, and Been Kim. Towards Automatic Concept-based Explanations, October 2019.
- [13] Bryce Goodman and Seth Flaxman. European Union regulations on algorithmic decision-making and a "right to explanation". *AI Magazine*, 38(3):50–57, September 2017.
- [14] Kh Nafizul Haque. What is Convolutional Neural Network — CNN (Deep Learning). <https://www.linkedin.com/pulse/what-convolutional-neural-network-cnn-deep-learning-nafiz-shahriar>.
- [15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition, December 2015.
- [16] Been Kim, Martin Wattenberg, Justin Gilmer, Carrie Cai, James Wexler, Fernanda Viegas, and Rory Sayres. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV), June 2018.
- [17] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment Anything, April 2023.
- [18] John McCarthy, Marvin L. Minsky, Nathaniel Rochester, and Claude E. Shannon. A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence: August 31, 1955. *AI Magazine*, 27(4):12–14, December 2006.

- [19] Tom M. Mitchell. *Machine Learning*. McGraw-Hill Series in Computer Science. McGraw-Hill, New York, nachdr. edition, 2013.
- [20] Christoph Molnar. *Interpretable Machine Learning - A Guide for Making Black Box Models Explainable*. Leanpub, 2019.
- [21] Angus Nicolson, Lisa Schut, J. Alison Noble, and Yarin Gal. Explaining Explainability: Recommendations for Effective Use of Concept Activation Vectors, February 2025.
- [22] Council of the European Union. Council recommendation of 22 May 2017 on the European Qualifications Framework for lifelong learning and repealing the recommendation of the European Parliament and of the Council of 23 April 2008 on the establishment of the European Qualifications Framework for lifelong learning. *Official Journal of the European Union*, 2017.
- [23] Eleonora Poeta, Gabriele Ciravegna, Eliana Pastor, Tania Cerquitelli, and Elena Baralis. Concept-based Explainable Artificial Intelligence: A Survey. *ACM Computing Surveys*, 2025.
- [24] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision, February 2021.
- [25] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detection, May 2016.
- [26] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, January 2016.
- [27] Antonio De Santis, Riccardo Campi, Matteo Bianchi, and Marco Brambilla. Visual-TCAV: Concept-based Attribution and Saliency Maps for Post-hoc Explainability in Image Classification. *Transactions on Machine Learning Research*, December 2025.
- [28] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, April 2015.
- [29] Timo Speith. A Review of Taxonomies of Explainable Artificial Intelligence (XAI) Methods. In *Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '22, pages 2239–2250, New York, NY, USA, June 2022. Association for Computing Machinery.
- [30] Superannotate. Image segmentation detailed overview [Updated 2024]. <https://www.superannotate.com/blog/image-segmentation-for-machine-learning>.
- [31] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the Inception Architecture for Computer Vision, December 2015.
- [32] A. M. Turing. Computing Machinery and Intelligence. *Mind, New Series*, 59(236):433–460, 1950.
- [33] Tyler Martin. *Interpretable Machine Learning*. PhD thesis, University of Cambridge, Department of Engineering (Darwin College), 2019.
- [34] Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. Qwen2-VL: Enhancing Vision-Language Model’s Perception of the World at Any Resolution, October 2024.
- [35] Feiyu Xu, Hans Uszkoreit, Yangzhou Du, Wei Fan, Dongyan Zhao, and Jun Zhu. Explainable AI: A Brief Survey on History, Research Areas, Approaches and Challenges. In Jie Tang, Min-Yen Kan, Dongyan Zhao, Sujian Li, and Hongying Zan, editors, *Natural Language Processing and Chinese Computing*, volume 11839, pages 563–574. Springer International Publishing, Cham, 2019.
- [36] Chaoyang Zhu and Long Chen. A Survey on Open-Vocabulary Detection and Segmentation: Past, Present, and Future, April 2024.

A. Appendix A

Table 4: Mapping between each probed concept and the ImageNet source classes used to build its concept set. Parenthetical notes report the contrast rationale for non-exhibiting classes

#	Concept	Configuration	ImageNet source classes
Base — object textures			
1	striped	Single-class	zebra
		Multi-class	zebra, tiger
		Random-class	polar bear (uniform surface), ping-pong ball, iPod, hammer
2	chequered	Single-class	crossword puzzle
		Multi-class	crossword puzzle, quilt
		Random-class	valley (irregular natural shapes), jellyfish, reflex camera, mountain bike
3	dotted	Single-class	dalmatian
		Multi-class	dalmatian, ladybug
		Random-class	striped skunk (linear pattern, not dotted), picket fence, electric fan, violin
Materials			
4	wood	Single-class	picket fence
		Multi-class	wooden spoon, park bench
		Random-class	steel arch bridge, window screen, jellyfish, volcano
5	water	Single-class	lakeside, fountain
		Multi-class	coral reef, lakeside, fountain
		Random-class	fire screen, library, screwdriver
Attributes			
6	braided	Single-class	hamper
		Multi-class	hamper, knot, pretzel
		Random-class	billiard table, bloodhound, cauliflower
7	bubbly	Single-class	bubble
		Multi-class	bubble, beer glass, seashore
		Random-class	stone wall, tricycle, cabbage butterfly
8	fibrous	Single-class	hay
		Multi-class	hay, wig, broom
		Random-class	soccer ball, digital clock, traffic light, fire engine, volcano
9	veined	Single-class	head cabbage
		Multi-class	head cabbage, brain coral
		Random-class	basketball, school bus, coffee mug, toaster, jellyfish

Table 5: Mapping between each concept or object proposed by Campi et al. [4], and the ImageNet source classes used to build its concept set

#	Concept	ImageNet-1K source classes
<i>Object- and scene-level concepts</i>		

Continua nella pagina seguente

#	Concept	ImageNet-1K source classes
1	asparagus	broccoli, cauliflower, artichoke, cabbage
2	cloister	monastery, church, palace, mosque, dome, bell cote
3	feline	tabby cat, tiger cat, Persian cat, Siamese cat, Egyptian cat, cougar, lynx, leopard, snow leopard, jaguar, lion, tiger, cheetah
4	grandfather	groom, academic gown
5	guitarist	acoustic guitar, electric guitar, banjo, cello, violin
6	kitchen	refrigerator, microwave, dishwasher, toaster, coffeemaker, espresso maker, wok, frying pan, mixing bowl, spatula
7	lichen	cliff, stone wall, lakeside, seashore, valley
8	rodent	hamster, guinea pig, fox squirrel, marmot, beaver, porcupine
9	tattoo	scuba diver, baseball player, groom
10	taxi sign	taxicab, minibus, police van, streetcar
11	newspaper	newspaper, crossword puzzle, book jacket
<i>Textures and patterns</i>		
12	bubbly	coral reef, fountain, beer glass, soap dispenser
13	cracked	stone wall, cliff
14	crystalline	alp
15	dotted	ladybug, Dalmatian
16	frilly	jellyfish, sea anemone
17	honeycombed	honeycomb
18	meshed	chainlink fence, sieve
19	perforated	colander, grating
20	spiralled	snail, nautilus, corkscrew
21	striped	zebra, tiger, tiger cat, tabby cat
22	waffled	waffle iron
23	woven	rug
24	wrinkled	African elephant, Indian elephant
25	spotted	dalmatian, leopard, jaguar, cheetah, ladybug
26	chequered	crossword puzzle, quilt, tile roof
<i>Materials and parts</i>		
27	cast iron	frying pan, wok, dutch oven
28	fin	great white shark, tiger shark, hammerhead shark, stingray
29	glass	wine bottle, beer bottle, goblet, vase, cup
30	grass	lawn mower, ox, bison, ram, bighorn sheep
31	leopard pattern	leopard, jaguar, cheetah
32	pen	fountain pen, ballpoint pen
33	sphere	soccer ball, tennis ball, ping-pong ball
34	steeple	church, monastery
35	wheel	mountain bike, tandem bicycle, unicycle, motorcycle, wagon

Abstract in lingua italiana

Le reti neurali convoluzionali (CNNs) sono modelli black-box che beneficiano dei framework di *explainability*; a questo scopo, metodi *concept-based* di *explainability*, tra cui Testing with CAV (TCAV), o Visual-TCAV, possono essere molto utili. TCAV è un framework che sfrutta i Concept Activation Vectors (CAVs), che devono essere allenati su un set di immagini di concetto. Solo poche pipeline esistono per estrarre automaticamente immagini da prompt testuali; in questo lavoro viene presentata una pipeline alternativa di estrazione: ritagliare porzioni di immagini reali, contenenti il concetto, usando modelli di detection e segmentation open-vocabulary. La principale domanda di ricerca investiga la possibilità di ottenere automaticamente un dataset che rappresenti correttamente il concetto, usando un prompt testuale e partendo da un set di immagini di input. La pipeline proposta è composta da moduli, tra cui: un estrattore, un cropper, ed un augmenter. Il dataset estratto è stato valutato sotto diversi aspetti: qualità intrinseca, stabilità tra diversi resamplings, e robustezza dei CAV, così come la similarità di CAV tra il dataset automatico e manuale, su ciascun layer. Questo lavoro presenta una riproducibile e documentata pipeline, alternativa alla generazione sintetica, da impiegare nel campo dell'*explainability* basata sui concetti; è da intendersi come uno studio esploratorio iniziale, che getti le basi per investigazioni future. I risultati mostrano una qualità complessiva comparabile alla baseline manuale, misurata in termini di metriche intrinseche, stabilità dei CAV, e similarità *cross-source*, ma con una forte dipendenza dal dominio di partenza.

Parole chiave: XAI, CAV, Open-Vocabulary Models

Acknowledgements

Grazie al Professor Marco Brambilla, ad Antonio De Santis, Matteo Bianchi e Riccardo Campi per avermi sapientemente consigliato durante questo (lungo) percorso; grazie per la vostra pazienza, professionalità e fiducia. Grazie ai miei colleghi di MegaHub, che sono anche amici ed amiche; sono orgoglioso di lavorare con voi, per lo spazio e l'accoglienza che mi avete fatto trovare e per quello che insieme stiamo costruendo.

Grazie ai compagni e compagne del Collettivo Rotte Balcaniche,

Grazie alla mia famiglia: alla mia mamma Luisa ed a mio papà Claudio, per aver “aspettato” in silenzio questo momento, pur non sempre capendo le scelte che ho fatto; grazie a mio fratello Riccardo, per aver provato a spiegarglielo. Grazie ai miei nonni ed alle mie zie.

Grazie ad Alberto, a Marco, a Leonardo, a Costa, a Pier.

Grazie a Francesco e Valeria, a Daniele e Sara, Emma, Gigi, Sofia, Luca; ad Ombra ed a Zeno.

Grazie al marablock ed a KRAP. Grazie Graziella e buona festa del grazie. Il gaso è tutto.