Politecnico di Torino

Polytechnic University of Turin

M.sc. Computer Engineering (AI and Data Analytics)

A.Y. 2024/2025

Graduation Session July 2025

Exploring Odd-One-Out Anomaly Detection

Supervisors

Prof. ssa. Tatiana TOMMASI

Dr. Paolo RABINO

Candidate

Silvio CHITO

Table of Contents

Chapter 1

Abstract

The Odd-One-Out anomaly detection problem is an emerging research direction aimed at identifying visually distinct instances within multi-object 3D scenes. Unlike traditional anomaly detection tasks, which rely on predefined notions of normality or anomaly, the Odd-One-Out task is inherently contextual. An object's status as "anomalous" is determined not by global priors but by its dissimilarity to co-occurring objects in the same scene. This contextual framing introduces significant modeling challenges, requiring both spatial understanding of individual objects and relational reasoning to capture how objects relate to each other.

This thesis explores how to learn context-aware object representations for 3D anomaly detection. We begin by reconstructing 3D voxel grids for each scene using multi-view 2D feature maps, which are backprojected using known camera intrinsics and extrinsics. We evaluate feature quality under different encoding strategies, including a baseline with ResNet50 and a distillation pipeline using DINOv2. Our experiments reveal that direct use of DINOv2 features outperforms distillation-based approaches and produces richer embeddings for downstream tasks, even if naive clustering on these features results insufficient for separating anomalies from normal objects.

To enhance the discriminativeness of object features, we apply contrastive learning on compact voxel grid representations, exploring strategies such as synthetic hard negative generation, positive samples extrapolation and memory banks. Despite limited gains, these experiments highlight the complexity of scene-dependent anomaly separation.

We then propose a more efficient alternative: compressing 3D object features into fixed-size embeddings and refining them with a Transformer-based architecture. Attention mechanisms help recalibrate these features within the context, while a novel Residual Anomaly Module introduces a learnable "normality centroid". This module allows the system to measure deviation from contextually-defined normality, effectively modeling scene-specific feature embeddings.

We evaluate our framework on the ToysAD8K and Parts15K datasets, which handle multi-view 3D scenes. The proposed approach demonstrates strong potential for real-world applicability in automated inspection pipelines, in particular where objects are numerous, similar and subtly different when anomalous.

Chapter 2

Introduction

2.1 Anomalies and the Odd-One-Out problem

In general terms, anomalies could be defined as something that does not match what we expect. Some examples can be a data point that does not fit into a particular pattern, a faulty piece inside a factory line, or an unusual shape in a medial scan. All these examples share the the same idea that an anomaly is represented by an aspect of what we're evaluating, that deviates from its standard behavior.

Usually, when dealing with single objects, anomalies are inferred based on the deviation computed over a predetermined data distribution in order to evaluate it quantitatively. In the particular case in which anomalies have to be spot among other objects, this becomes not trivial, since there could me many scenarios where the observed object is part of the majority of the objects while not in others. This defines the Odd-One-Out problem, which emphasizes that what does not fit into the known standard behavior of what we're observing, relies heavily on the context.

This makes the definition of anomalies really broad and flexible, since it is linked directly to the environment in which the observation has taken place. For instance, a red toys would be considered an outlier inside a box full of green toys, but not if red toys would have composed the majority of the objects inside the box. Therefore, anomalies are not defined by predefined rules but rather by comparison with the context, represented by everything around the observed sample.

This contextual dependency means that adaptive systems are required, in order to understand and model the surrounding environment or reference group. This leads to the context-aware anomaly detection methods, where the system is supposed to learn firstly the typical structure or behavior within a certain setting, and then infer deviations related to the learned context. Consequently an object or a pattern can be flagged as an anomaly in one scenario and entirely normal

in another, highlighting the importance of relational understanding over absolute criteria. This dynamic nature of anomalies open up to new challenges by aiming to design systems that are sensitive to context and able to compute responses by reasoning based on situational awareness.

2.2 Anomalies in Computer Vision

Anomaly detection in computer vision relates to the task of identifying unexpected pattern, objects or events within visual data. Traditionally, the field has focused mainly on 2D images in order to detect semantic novelty, like new objects or categories that have not been seen during training, or localized defects like structural or appearance irregularities. These methods rely on visual clues such as color, shape, texture or surface inconsistency to recognize between normal and anomalous regions.

In many practical scenarios, anomaly detection systems are applied in industrial quality control, where the goal is to identify defects on manufactured parts, or in surveillance, where unusual behavior or objects must be identified automatically. Medical imaging also benefits from anomaly detection by highlighting irregular patterns related to human body parts, which may indicate a certain disease.

Recently, the scope of anomaly detection has expanded to incorporate richer and more complex representations of the visual world. Instead of relying purely on 2D data, modern approaches manage to integrate multi-modal data sources such as depth maps, point clouds and multi-view images. This shift is motivated by the fact that 2D images alone may be not enough to address structural anomalies that can be identifies in three dimensions or are only visible from certain view angles.

Moreover, a growing trend of pointing out logical inconsistencies and integrating auxiliary information like language descriptions are rising, in order to provide a more comprehensive understanding of anomalies. These developments reflect a broader interest toward multi-modal and context-aware anomaly detection that aims approximate human perception and reasoning capability as much as possible.

Despite recent progress, many anomaly detection methods still face significant challenges when applied outside simple or controlled environments. Real-world scenarios are often represented by far more complex scenes, which can present multiple overlapping objects, inconsistent lighting, and varying viewpoints. These factors introduce noise and ambiguity that models trained in simplified settings may not handle well enough. To be truly effective, anomaly detection systems must be robust, efficient, and able to generalize beyond the specific conditions they were trained on.

Anomaly detection through visual data can be handled in many ways, depending

on the nature of the task. In certain cases, the goal is to detect global anomalies without necessarily pointing out where the anomaly occurs, therefore employing an holistic evaluation. This is common in settings like medical diagnostics or scene-level quality control, where a model should simply determine whether an input is normal or not, without localizing the cause.

Anyway, in many real-world applications, anomalies are tied to specific instances of regions within a scene. In this case object detection become a valuable component, since it let the downstream algorithm analyze visual data with finer granularity by inspecting each detected instance in its own context. For example, in manufacturing, each detected product can be analyzed for defects, and in surveillance, object detection can isolate people or vehicles before evaluating whether their behavior is unusual or not. Thus, object detection is often used as a first step in order to enable the anomaly detection module to pay more attention and make more targeted and focused decisions.

2.3 Current work limitation

Most existing methods mainly focus on simplified settings involving isolated objects or single-view images, which limits their usage in real-world settings. Consequently they struggle with generalization when they're exposed to unseen environments, rare objects configuration or domain shifts between training and test data distributions, since they often rely on implicit assumptions or predefined patterns that do not always hold in practice.

When dealing with multi-object scenes, in particular with multi-view data where multiple instances are arranged in complex and cluttered layouts, occlusions and overlapping elements pose significant challenges. The anomaly detection task in this scenario requires models that can fuse information from various view points, in order to build a coherent 3D understanding of the scene geometry and appearance. Additionally, shaping the notion of normalcy is not universal, but rather dependent on the context, whose meaning is given by all its components and their relative relationships.

Such models must generalize well to new, unseen object categories and instances, which is inherently a difficult task due to the diversity and variability of real-world environments. All these challenges, such as handling multi-object clutter, leveraging multi-view data, reasoning within complex scenes and ensuring robust generalization, brings out the current limitations of anomaly detection in multi-view multi-object centric scenarios and motivate the development of more sophisticated and adaptable frameworks.

2.4 Core contributions

In this thesis work, we propose a framework for anomaly detection in multi-object centric scenes using an odd-one-out formulation. We explore how to handle multi-view data for obtaining a 3D semantically rich representation of a scene and how to actually detect objects inside it, in order to assess the 'anomaliness' of the detected instances. On top of these modules, we formulate a new lightweight and fast framework for anomaly detection based on a Transformer Encoder, potentially being useful in industrial or manufacturing scenarios where speed and efficiency play a crucial role. In this part of the work, many strategies have been explored to boost the context-awareness of the model along with its generalization capabilities.

To improve context-awareness in our Transformer-based anomaly detection framework, we explored strategies that let the model reason about each object in relation to the entire scene (thought as the totality of objects inside it). We introduced a normality token into the transformer sequence to represent the global scene context, guiding the interpretation of each object embedding. A contrastive loss approach was also examined to promote better generalization by clustering normal instances and separating potential anomalies. Finally, we added a residual anomaly head that measures how much each object deviates from the learned norm, enabling more precise and interpretable anomaly scores.

Chapter 3

Literature Review

3.1 Visual Anomaly Detection

Visual anomaly detection is a challenging task that has raised attention thanks to its relevance in many real-world domains like manufacturing, industrial inspection or quality control. The main difficulty lies in the definition of anomaly, since anomalies are inherently rare and unpredictable and therefore collecting enough labeled samples still remain a challenge. Normally these kind of tasks have really imbalanced dataset, with many normal samples, but very few anomalous ones. However, researchers have explored alternative strategies including supervised and self-supervised [1],[2], unsupervised [3] [4] [5], few-shot [6] [7], and zero-shot learning methods [8] [9] [10] that can operate with limited or no anomaly-specific supervision. In order to solve these challenges, many different models have been proposed, ranging from traditional computer vision techniques to deep learning-based architectures. These models differ in their ability to capture semantic and spatial patterns, which are relevant to detect deviations from normality.

3.1.1 Types of Anomalies

Visual data can be organized in the following categories: data point, entity, relation, and frame [11]. A data point is the smallest indivisible element captured by imaging device (e.g. a pixel for an image) and a cohesive set of data points compose an entity, which represents a real-world object. A relation function takes multiple entities as input and combines them to form a visual frame, which encapsulates the relationships between different entities, capturing contextual information within the visual scene. This leads to the formation of hierarchical anomalies [11], which can propagate from the lower level, like data points, until higher ones, like relationships among objects inside a frame.

Based on the visual data category we're analyzing, we can distinguish among two main anomaly types: structural and semantic anomalies [11]. Structural anomalies refer to deviations of data points regarding geometry, shape or spatial representation of an object by focusing on local structural irregularities. This means that a more fine-grained analysis is exploited for detecting deviations on a single object.

Semantic anomalies relate to the entity, relation, and frame levels of visual data, where deviations emerge from contextual inconsistencies rather than localized defects. Such anomalies arise when an object appears in an unexpected context, when relationships between objects violate learned patterns or when the overall scene composition differ from expected semantics. Detecting these kind of anomalies requires a higher understanding of object roles, spatial arrangements and inter-object relationships. This often involves models capable of context-aware reasoning, leveraging tools such as attention mechanisms, scene graphs, and relational learning frameworks to capture and interpret the semantics of the visual scene.

This distinctions have brought to the birth of many different datasets for anomaly detection regarding visual data like MVTec AD [12], which contains high resolution images of industrial objects and textures with structural defects like scratches, dents and contaminations, focusing on localize defects. Semantic anomalies instead are covered in datasets like MVTec-LOCO [13], which includes logical anomalies where objects appear in incorrect locations, are missing entirely, or violate contextual rules.

3.1.2 Traditional Methods

Traditional methods for anomaly detection cover statistical, clustering and classic machine learning models, which have been reliable tools for many applications thanks to their simplicity, interpretability and low-cost computation. Statistical techniques try to reconstruct the normal samples probability distribution and classify anomalies based on whether the samples fall inside the reconstructed distribution. Typical models that belong to this category are parameter distribution estimation, such as Gaussian [14] or Gaussian Mixture Models [15], kernel density estimation method [15] and non-parametric estimation methods like Nearest-Neighbour.

Clustering methods focus on grouping similar object embeddings into clusters within the feature space [16] [17]. Anomalies are then detected as data points that do not fit well into any cluster [18] [19]. This approach works well when anomalies follow known patterns distinct from normal data. However, their performance degrades in scenarios where anomalies are context dependent, meaning that their anomalous nature is defined relative to the surrounding environment. In this cases, the decision boundary between normal and anomalous instances becomes not absolute anymore,but rather scene-specific. Additionally, these methods perform

better with low-dimensional feature representations, which in most real-world scenario is not the case.

Classical machine learning based methods includes one-class classifiers, whose task is to classify a single class in feature space. The traditional approach is represented by one-class support vector machines (OCSVM) [20] and support vector data description (SVDD) [21], which aim at modeling the boundaries for normal samples, by flagging everything that that not reside inside it.

However many limitations are met by these models. Statistical methods rely on strong assumptions on the target distribution, which is not always met due to the visual data complexity in the modern datasets and large amounts of training samples for shaping the distribution. Moreover it is hard to model distributions when dealing with high-dimensional features like images and draw clear separable decision boundaries for normal and anomalous samples.

3.1.3 Deep Learning-Based Anomaly Detection

Thanks to the advent of deep learning models it has been possible to handle complex pattern in high dimensional data, therefore a new class of models has raised in order to deal with these challenges by exploiting the richness of the feature outputted by a neural network.

Thanks to this kind of feature richness, it has been possible to represent visual data like images or videos through compact and meaningful feature vectors, useful for downstream tasks like anomaly detection task. All these qualities, obtained by the neural network based feature extractor, are already suitable for classical machine learning algorithms. This fills the gap of interpretability and transparent decision-making for more complex models like neural networks while still delivering expressive and adaptable features representations [22]. Clustering-based models exploit the features coming out of a neural network for clustering data points based on their proximity in feature space, improving detection accuracy by retaining its cluster nature [23]. Normalizing flows [24] estimate the probability distribution of the data and can be used in conjunction with neural feature extractor as probabilistic estimators [25] [26] Support Vector Data Description (SVDD) [21] instead is a classical machine learning method and aims to find the boundaries that separate normal data from anomalous ones. When it is used on top of deep neural network features it manages to create precise boundaries for high dimensional data.

Reconstruction based models are trained in order to learn the distribution of the normal data by reconstructing it [27] [28] . The distribution of normal samples is learned when the reconstruction error for normal instances is relatively low compared to the anomalous case, which should lead to a big reconstruction error. Most reconstruction methods have employed GANs, VAEs and Diffusion Model for the reconstruction task and each of them have advantages and pitfalls, being

discusses in details in the following studies [29], [30], [31], [32]. This strategy has been extended to feature reconstruction [33] [34] and knowledge distillation [35] [36], under the hypothesis that a student network trained to mimic a teacher exposed only to normal data will underperform on anomalies.

With the availability of large-scale pre-trained vision models, more recent works have leveraged their rich representations by computing feature distances [37] [38], though these methods often involve computationally expensive memorization and matching steps. Another line of research revisits the supervised setting by synthesizing anomalies through noise injection, image mixing [39], or generative models [40] [41].

3.2 Foundational Models as Feature Extractor

Recent research in self-supervised learning introduced foundational models capable of learning general-purpose visual representations from large-scale image datasets without explicit manual annotations. In particular, DINO [42] (Self-Distillation with No Labels) and its improved version DINOv2 [43] have demonstrated extraordinary performance as feature extractors across various downstream tasks, including object detection, segmentation, and 3D understanding. DINO leverages a self-distillation framework using Vision Transformers (ViTs), where a student network is trained to match the outputs of a momentum teacher on different image augmentations. This approach allows DINO to learn semantically rich and spatially coherent representations without labels. DINOv2 improves its predecessor by scaling up the training data, model capacity, and training duration, resulting in more robust and transferable features. It introduces improvements such as more diverse training data, stronger regularization, and better optimization techniques. Both DINO and DINOv2 have proven especially effective in scenarios where high-quality features are required for tasks involving limited supervision or domain adaptation, making them ideal as backbone feature extractors in complex pipelines, including 3D scene understanding and anomaly detection.

3.3 Contrastive Learning for Anomaly Detection

In the context of visual representations, Contrastive Learning represents a learning framework whose focus is to distinguish between similar (positive) or dissimilar image pairs (negative), in order to obtain a discriminative visual representation of input data. It is self-supervised learning paradigm, meaning that no labeled data is required for training. Among the most popular frameworks there's SIMCLR [44], which exploits image augmentations of the same image (positives) against all the other samples in the batch for learning (negatives). MoCo (Momentum Contrast)

[45] maintains a dynamic memory bank of negatives and uses a momentum encoder for stability while BYOL and DINO avoid explicit negatives by using asymmetric networks to align positive pairs. Contrastive Learning has shown robust performance in representation learning for downstream tasks like classification, detection, and segmentation by improving consistency for intra-class samples while spreading out outliers [46]. This benefits both one-class classifiers and clustering-based detectors. In the anomaly detection setting, CL has been used widely for learning meaningful representation in the embedding space [SPADE, CutPaste, DRAEM] for images, patches or synthetically generated samples, since the main goal in these scenarios is to pull all the positive samples together while pushing all the negatives apart.

Although CL has gained attention as a self-supervised framework, its supervised version SupCon [47] has also shown promising results. SupCon builds upon the same core idea but incorporates class label information during training. Instead of treating only augmented views of the same image as positives, SupCon pulls all examples of the same class closer in the embedding space while pushing apart those from different classes explicitly.

3.4 3D Anomaly Detection

Thanks to the advances in 3D sensing technologies, many other data modalities have been explored (like point clouds, depth maps and RGB-D images) in order to represent real-world objects and scenes. This richer representations provide 3D-aware information that is often missing from traditional 2D color images, making them useful in scenarios where subtle or complex anomalies must be detected.

Recent research extends 2D methods to leverage additional structural information [48] [49], focusing on improving detection through feature fusion and robust anomaly simulation. Other techniques combining RGB and depth information are showing interesting results [50] [51]. These hybrid data fusion techniques demonstrate that multimodal approaches outperform RGB pure based methods on MVTec 3D-AD dataset [52], allowing for more precise anomaly localization by leveraging complementary information from both modalities.

When we face settings with limited sensor data, diffusion-based reconstruction models like R3D-AD offer promising solutions [53]. By transforming data distributions during the diffusion process to obscure input anomalies, these models achieve high performance on diverse datasets, including Real3D-AD [54] and Anomaly-ShapeNet [55].

In our work, we focus on multi-view images of object-centric scenes, coming from the ToysAD and PartsAD datasets [56]. We backproject the image features for each view into a common 3D voxel grid reference, representing an entire scene, where features are accumulated and then further analyzed for the Odd-One-Out

task.

3.4.1 3D Object Detection

The 3D object detection task's main goal is to localize and identify objects in the real-world three-dimensional space, by providing information about their position, size and orientation. Instead of working on images that live in the 2D plane, as the totality of 2D detection algorithm do, 3D object detectors deal with depth and spatial data from sensors like LiDAR, stereo cameras or RGB-D devices, in order to build a richer representation of the environment. These type of data are really important when dealing with scenarios that require in-depth spatial awareness, such as in autonomous systems and robotics applications. For example, in self-driving cars, these capabilities are essential for identifying pedestrians, vehicles and obstacles, ensuring safe navigation. In a similar way, robotic systems rely on 3D object detection to interact with dynamic environments like manufacturing production lines and warehouses. These examples highlight the importance of detecting individual objects in 3D space, since it allows to perform a fine-grained analysis based on each object's spatial attributes, rather than analyzing the scene (composed of many instances) directly as a whole. 3D object detectors can be classified based on the data they operate on, like image-based, point-cloud based and hybrid approaches [57]. Among these methods, recent multi-view image-based models exploit multiple camera perspectives to predict 3D object locations by learning 3D position embeddings directly, while aligning them to multi-view features [58] [59].

Voxel-based methods instead, convert raw 3D point clouds into structured voxel grids, allowing the use of 3D convolutional neural networks to extract spatial features. VoxelNet [60] employs this approach by learning end-to-end from voxelized point clouds, while other frameworks like PointPillars [61] and PV-RCNN [62] improved computational efficiency and accuracy by combining voxel and point-based features.

Hybrid methods that fuse multi-view image inputs with voxel or point cloud representations are also emerging, aiming to combine the rich semantic information from images with the precise spatial attributes from point clouds [63] [64]. Their goal is to combine the strenghts resulting from both modalities and enhance object representations.

3.5 Multimodal Large Language Models for Visual Tasks

MultiModal Large Language Models (MM-LLMs) have made significant advancements by enabling standard LLMs to handle multimodal (MM) inputs and outputs through cost-effective training methods. These enhanced models keep the strong reasoning and decision-making abilities of traditional LLMs, while also supporting a wide variety of multimodal tasks [65]. Image-focused MM-LLMs, often called Large Vision Language Models, typically include a vision encoder, a language encoder, and a cross-modal alignment network [66]. MM-LLMs use LLMs as their core "cognitive engine" bringing valuable features such as robust language generation, the ability to transfer knowledge to new tasks without retraining, and in-context learning (ICL) [65]. Early research in this area mainly targeted understanding multimodal content and generating text, with tasks like image-text comprehension seen in systems such as LLaVA [67], MiniGPT4 [68], and OpenFlamingo [69]. More recently, MM-LLMs have been further developed to support outputs in specific modalities, including tasks that generate both images and text, as demonstrated by models like Emu [70], MiniGPT-5 [71] and Gemini Flash 2.0 [72] .

Chapter 4

Methods

4.1 The Odd-One-Out existing architecture

The method we took inspiration from is the one presented in [56], a transformer-based architecture designed for object-centric anomaly detection in 3D scenes. Built on voxelized DINOv2 features, OddOneOut architecture encodes object-level representations while leveraging scene-level context to identify anomalies relative to their surroundings.
It considers a scene containing N instances of an object category. In particular, it takes M multi-view images, along with their camera parameters $\{(\mathbf{I}_t, \mathbf{P}_t)\}_{t=1}^{M}$ and aims at learning a mapping from these multi-view input to object-centric anomaly labels and their corresponding 3D bounding boxes

$$\psi : \{(\mathbf{I}_t, \mathbf{P}_t)\}_{t=1}^{M} \mapsto \{(y_n, \mathbf{b}_n)\}_{n=1}^{N}.$$

The entire architecture can be divided in four main components:

- **The 3D feature volume construction module** computes a 3D feature volume with the multi-view images. They're processed through a shared 2D CNN-based encoder that outputs 2D feature maps $\mathbf{F}_t = \mathcal{E}_{2D}(I_t) \in \mathbb{R}^{d \times h \times w}$, which are projected in voxel space through the camera parameters. The projected features are accumulated inside a voxel grid and then averaged. The final feature volume is then fed to a 3D CNN-based network in order to refine the projected features

$$\mathbf{F}_v = \mathcal{E}_{3D}\left(\text{aggr}\left(\{\Pi_{\text{proj}}(\mathbf{F}_t, \mathbf{P}_t)\}_{t=1}^{M}\right)\right)$$

- **The feature enhancemenet module** is needed for reconstructing the appearence and the geometry of the scene and is actually used only at training time. It makes use of two-layered 1x1x1 convolution blocks α_c and α_σ, which are employed for obtaining color and density volumes. Then, the pixel-wise color and density maps are composed by integrating along a camera ray using volume renderer $\mathcal{R}$.
  Additionally, in order to enrich the reconstructured features, a distillation process, with DINOv2 as the teacher network, is exploited. A similar 1x1x1 convolution block β is employed in order to compute a neural feature field starting from the initial 3D feature volume. These feature volumes are then fed to the volume renderer (just like with color and density heads outputs) for computing a view-specific render. The rendered view is then compared directly with the teacher's features. The image rendering and its corresponding loss function for a single viewpoint P_t are shown below:

$$\mathcal{L}_t^{\text{im}} = \|I_t - \hat{I}_t\|^2 + \|I_{t\sigma} - \hat{I}_{t\sigma}\|^2, \tag{3}$$

  where $[\hat{I}_t, \hat{I}_{t\sigma}] = \mathcal{R}([V_c, V_\sigma], P_t)$, $\hat{I}_t$ and $\hat{I}_{t\sigma}$ are the rendered image and mask.

- **The object-centric feature extraction module** aims at localizing the objects inside the scene in order to compare all the instances among each other. In this case the density volume obtained in the feature enhancement block is used for obtaining a coarse point-cloud through a pre-defined threshold. Then, a DBSCAN algorithm is employed for distinguish among foreground objects and producing the related bounding boxes. The predicted bounding boxes are then brought all to the same 8x8x8 dimension thanks to RoI pooling, in order to be comparable. This method specifically, does not need ground truth bounding boxes.

- **The Cross-instance matching module** aims at computing local similarities among instance volumes pairs (z_n and z_m). After retrieving the projected feature volumes through β, instances are compared by means of voxel-wise similarity and a topk operation takes only the most similar ones:

$$\mathcal{C}_k^{nm}[i] = \text{top}_k \left[\beta(\mathbf{z}_n[i])^\top \beta(\mathbf{z}_m[i])\right]$$

  The result of this operation is employed for performing a sparse-attention based comparing between two volumes. Finally, the updated feature volume is passed through 3D CNN blocks to downsample by a factor of 1/8, which is finally reshaped into a vector and fed to a 2-layer MLP outputting the final prediction label y_n. A binary cross entropy loss is used for computing classification loss.

A schematic illustration of the above method is displayed in 4.1.

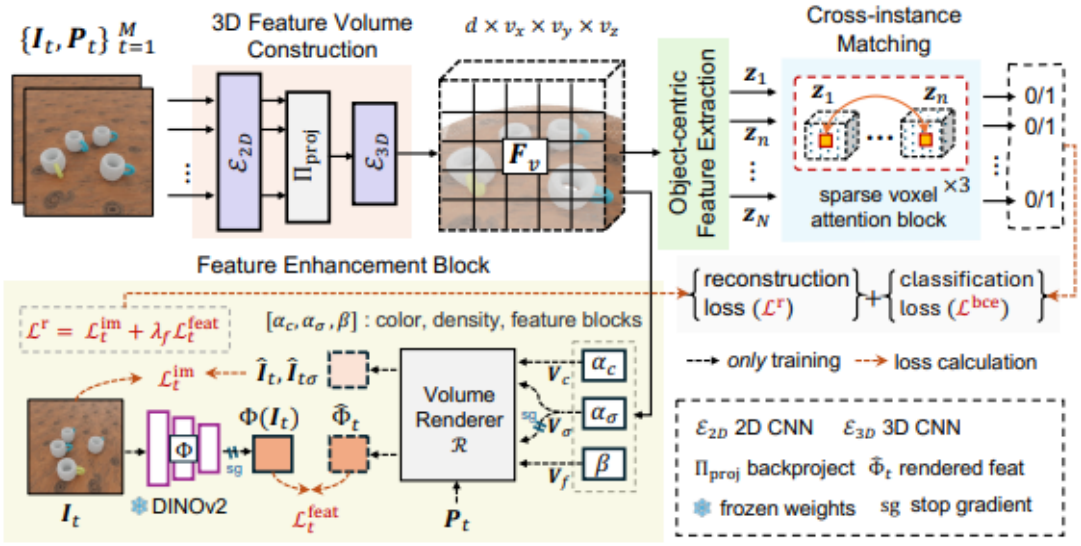

Figure 4.1: Illustration of the OddOneOut [56] framework.

In this work, we begin by analyzing the original method proposed in [56], highlighting its main strengths and limitations through extensive experiments. We tackle challenges related to the clusterability of object representations in feature space and explore how pre-trained models like DINOv2 can enrich 3D spatial features. Moreover, we investigate the role of contrastive learning in enhancing the discriminative power of features within object classes. We also propose a more memory-efficient embedding strategy for object representations that keep competitive performance while significantly reducing memory usage. On top of these considerations, we introduce a novel approach that leverages these compact embeddings to compute a 'normality' prototype for each scene and measure deviations from it, in order to detect odd objects.

4.2 MLLM Baseline

Recent advances in Multimodal Large Language Models (MLLMs) have demonstrated remarkable reasoning capabilities. By leveraging textual knowledge, these models significantly enrich visual understanding and enable a wide range of downstream tasks in a zero-shot setting (i.e. without task-specific training), including anomaly detection [73] [74]. While MLLMs have proven effective in identifying anomalies based on the global appearance of objects, they still exhibit limitations in visual grounding [75]. Furthermore, their ability to reason about fine-grained anomalous details is currently an open question [76], especially considering that most existing benchmarks are restricted to single-object scenes observed from a single viewpoint. To explore the performance of MLLMs on the odd-one-out task, we present a tailored pipeline that leverages the Set-of-Mark (SoM) prompting method [77]. This strategy supports spatial understanding by detecting the objects in the scene and overlaying visual marks directly on the images, which are then provided as input to the MLLM. Specifically, we use gemini-flash 2.0 [78] and feed it the following prompt: Here is a list of images taken from multiple view-points of the same scene, each object is annotated with an index and the bounding box, one or more objects are different from the majority of all other objects, reply only with a list of indices of the odd objects. The annotated images are concatenated to the prompt by following [77] as shown in 4.2. The model outputs the annotated index of the odd objects from which we can evaluate the prediction accuracy.

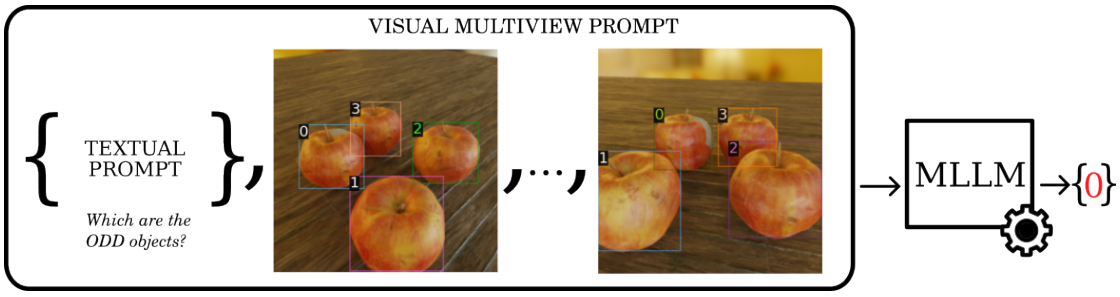

Figure 4.2: Illustration of the OddOneOut [56] framework.

Chapter 5

Experiments and Results

5.1 Dataset

We adopt the same experimental setup of [56], considering three tracks from two datasets containing objects with synthetically generated anomalies. Toys is composed of 8K scenes with 3-6 objects from the 51 categories of the Toys4D dataset [79] rendered in 20 views. The collection is divided into a training set with 5K scenes from 39 categories and two disjoint test sets: one (seen) with 1K scenes from the seen categories but with unseen object instances, and the other (unseen) containing 2K scenes from the remaining 12 novel categories like shown in 5.1. Parts was defined from the ABC dataset [80] mostly containing mechanical parts shapes. The collection contains 15K scenes, each consisting of 3-12 objects rendered from 20 viewpoints, and is divided into train and test splits of respectively 12K and 3K scenes, with the latter including only unseen shapes.

Categories
Seen
dinosaur, fish, frog, monkey, light, lizard, orange, boat, dog, lion, pig, cookie, panda, chicken, orange, ice, horse, car, airplane, cake, shark, donut, hat, cow, apple, bowl, hamburger, octopus, giraffe, chess, bread, butterfly, cupcake, bunny, elephant, fox, deer, bus, bottle, hammer, mug, key
Unseen
plate, robot, glass, sheep, shoe, train, banana, cup, penguin

Table 5.1: Toys 'seen' and 'unseen' object categories

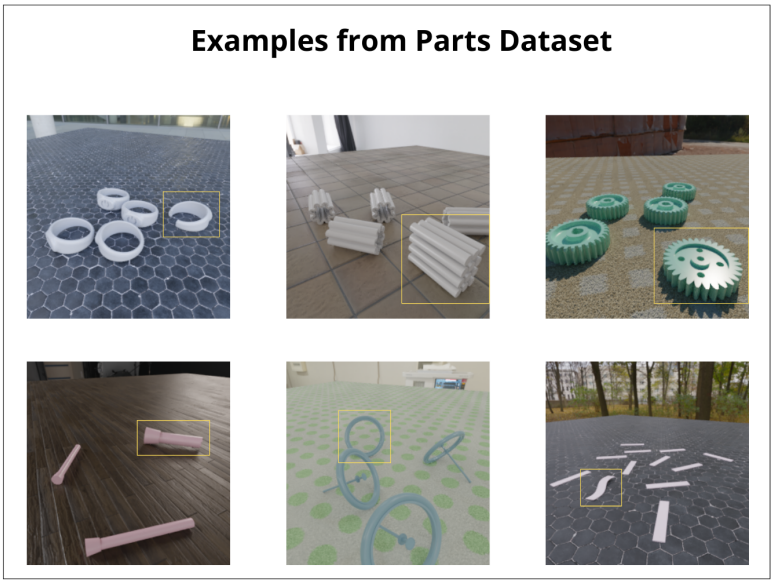

Figure 5.1: Illustration of the OddOneOut [56] framework.

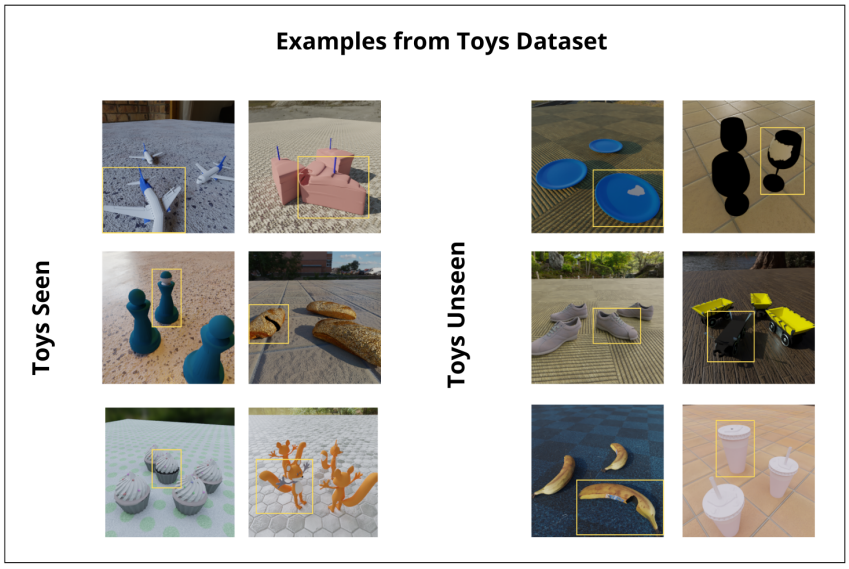

Figure 5.2: Illustration of the OddOneOut [56] framework.

5.2 Metrics

As in [56], we evaluate the anomaly prediction performance via two metrics. The Area Under the ROC curve (AUC) measures the ability of a model to discriminate odd from normal objects regardless of the chosen confidence threshold. The Accuracy considers an operating scenario with a threshold equal to 0.5. In our analysis we feed the models with the ground truth location of the objects and focus only on the odd identification performance. This is justified by observations of the previous work [56] about the negligibility of the localization error due to the relative simplicity of the setting. Note that for the zero-shot MLLM baseline we collect the Accuracy, but we do not have access to the confidence score so we omit the AUC. Moreover, since the concept of majority heavily depends on the number and type of objects inside the scene, we decided to introduce also a clustering metric when analyzing the approach with clustering algorithm, the adjusted random score, to evaluate how good the model is at grouping objects of the same type regardless of the class they belong to. The Adjusted Rand Score (ARS) measures the similarity between two clusterings by considering all pairs of samples and counting those that are assigned consistently in both. This is justified by the fact that a clustering algorithm could correctly group objects of the same type, but label them incorrectly due to the dynamic definition of anomaly.

5.3 Implementation details

A ResNet50-FPN [81] is employed as the 2D image encoder and a four-scale encoder-decoder-based 3D CNN [82] as the 3D backbone. For input, we use M=5 views, each with a resolution of 256×256, though our model can accept a different M during inference. The 3D volume F_v contains $96 \times 96 \times 16$ voxels with a voxel size of 4cm. Regarding rendering, 128 points per ray are sampled for rendering, and the rendered features have a spatial dimension of 32×32 ($h_f \times w_f$). The threshold applied to the density volume V_σ is set to be 0.2 and the DBScan algorithm is run with its default parameters. Moreover, the object crops are normalized to $32 \times 8 \times 8 \times 8$ through bilinear interpoation. Three sparse voxel attention blocks are employed, and each applies 8-headed attention. The model is trained in two different stages. The first one focuses on the training of the 3D feature volume construction and the feature enhancement module, in order to let the model learn how to represent the 3D scene starting from multi-view images and camera parameters (50 epochs). Once the model manages to accomplish this task, these modules are further trained along with the matching module, which is in charge of identifying odd objects from their 3D voxel representation (other 50 epochs). The batch size is 4, and the Adam optimizer is used with a fixed learning rate 2×10^{-5}. Moreover in the first

experiments, only the unseen test set has been tested regarding Toys dataset, while the seen test set has been employed in the last section of experiments (along with the unseen test set).

5.4 Study on clustering behaviour

A key question we addressed regarding the OddOneOut architecture was the actual contribution of the matching module in identifying anomalous objects. After training, the model is expected to produce 3D object representations that are informative enough to distinguish outliers from the dominant object category within a scene. To investigate this, we applied various clustering algorithms to the objects voxel representations, to evaluate whether the features produced at this stage were sufficient to separate normal and anomalous objects. This analysis allowed us to assess whether the matching module plays a critical role in odd object detection, or if the initial feature representations alone are already discriminative enough.

The followings are the clustering algorithms that we chose to use:

- **KMEAN**: it is a centroid-based clustering algorithm that partitions data into k clusters by minimizing the within-cluster variance. It iteratively assigns points to the nearest cluster center and updates the centers until convergence.

- **Spectral clustering**: it leverages the eigenvalues of a similarity matrix derived from the data to perform dimensionality reduction before applying a standard clustering algorithm, such as k-means. By projecting the data onto the subspace spanned by the top eigenvectors, it captures the most significant structural relationships in the similarity graph, enabling better separation of complex or non-linearly separable clusters.

- **DBSCAN**: DBSCAN (Density-Based Spatial Clustering of Applications with Noise) [83] is a density-based algorithm that groups together points closely packed in space and marks as outliers those that lie in low-density regions. It does not require specifying the number of clusters and is robust to noise.

- **Anomaly score based clustering**: computes a score for each object inside a scene by averanging the distances with all the other objects. In this case the threshold is scene-dependent since it is the mean of each object's average distance from all the others inside a scene.

- **Graph cut**: we cast our clustering problem to a problem solvable by using graphs. We first build our graph representing the scene (each object inside the scene is a node in the graph) and then try to divide it into two sub-graphs containing hopefully the anomalous elements.

- **N-cut**: This function performs spectral clustering on a set of features using the second eigenvector of the normalized Laplacian matrix (Fiedler vector) to partition the data into two groups (that represent anomalous and normal objects).

The main information that these kind of algorithms need is how to compute distances between instances: the more the distance in feature space, the more the objects are different. We test them with some basic and custom metrics:

- **Cosine similarity**: Measures the angular difference between two vectors, capturing their orientation regardless of magnitude.
- **Euclidean distance**: Computes the straight-line distance between two points in the feature space, reflecting their absolute difference.
- **Max dissimilarity**: We use the maximum euclidean distance between voxels of two objects representations.
- **Min Max dissimilairty**: We take inspiration from [4] and compute the euclidean distance for each pair of voxels between two objects representations. Then we compute the minimum distance for each voxel w.r.t. all the other ones belonging to the other object and finally take the maximum value.
- **Optimal transport**: The maximum distance among two objects representations is computed by solving the optimal transport problem over the voxels features (we want to discover "how far" is the distribution of the first object voxels features w.r.t. the second one).

5.4.1 Discussion and Results

The clustering results suggest that the object voxel features do not contain relevant discriminative information and are not sufficient for reliably identifying anomalies on their own. Performance varies significantly across both methods and datasets. On *Toys Unseen*, N-Cut and Graph Cut showed slightly better results, but overall scores remained modest, indicating that object features alone don't consistently separate normal and anomalous objects 5.2.

Employing a custom distance metric did not manage to get higher performance as shown in 5.3, 5.4, 5.5 and posed other challenges. The fact that we do not

Table 5.2: Performance of the clustering algorithms over objects representations obtained through ResNet50 feature projection. In this case we employ basic distance metrics such as cosine similarity and euclidean distance

		Toys Unseen			**Parts**		
Clustering approach	**Distance metric**	**AUC**	**ACC**	**ARS/ std. dev.**	**AUC**	**ACC**	**ARS/ std. dev.**
Anomaly score	cosine similarity	54.58	55.68	0.0395/0.1418	62.49	65.21	0.1092/0.2277
KMeans	euclidean	51.15	57.29	0.0223/0.1500	59.41	66.61	0.1258/0.2783
Spectral Clustering	euclidean	55.56	60.44	0.0706/0.1944	63.70	68.82	0.1882/0.3041
DBSCAN	euclidean	57.13	53.38	0.0060/0.1224	65.42	79.14	0.2556/0.3098
Graph-Cut	cosine similarity	56.16	64.05	0.0770/0.2188	59.85	75.15	0.1795/0.3345
N-Cut	cosine similarity	62.23	61.44	0.0872/0.1913	54.52	53.87	0.0448/0.1684

know the anomaly distribution a priori means we cannot define a fixed distance for determining how much an object differs from another, causing density-based algorithms like DBSCAN to suffer because their performance heavily depends on parameters such as ϵ (epsilon), which may not adapt well to varying densities or unknown anomaly patterns in the data (ϵ is the maximum radius within which points are considered neighbors and thus potentially part of the same cluster). Moreover, methods like optimal transport are often too time-consuming due to their high computational complexity, making them impractical for large or real-time datasets.

Table 5.3: Performance of the clustering algorithms when optimal transport is chosen as distance metric

	Toys Unseen		
Clustering Approach	**AUC**	**ACC**	**ARS/ std. dev.**
Anomaly score	50.20	50.68	0.0138/0.1144
DBSCAN	50.00	69.28	0.00/0.00
Graph-Cut	52.42	60.88	0.0327/0.1585
N-Cut	59.09	59.32	0.0533/0.1671

Overall, these findings confirm that the matching module adds meaningful value. Without it, object features only partially capture what makes an object "odd" in a scene-specific context. Clustering gives a useful baseline, but it's not robust enough

Table 5.4: Performance of the clustering algorithms when 'max dissimilarity' is chosen as distance metric

	Toys Unseen			Parts		
Clustering approach	**AUC**	**ACC**	**ARS/ std. dev.**	**AUC**	**ACC**	**ARS/ std. dev.**
Anomaly score	42.57	43.13	0.0264/0.1201	32.49	33.78	0.1413/0.2492
DBSCAN	50.00	31.46	0.00/0.00	50.00	23.44	0.00/0.00
Graph-Cut	44.90	54.47	-0.0151/0.1208	44.31	64.31	-0.0771/0.1298
N-Cut	54.99	55.76	0.0194/0.1427	50.56	50.67	-0.0069/0.1082

Table 5.5: Performance of the clustering algorithms when 'min max dissimilarity' is chosen as distance metric

	Toys Unseen			Parts		
Clustering approach	**AUC**	**ACC**	**ARS/ std. dev.**	**AUC**	**ACC**	**ARS/ std. dev.**
Anomaly score	39.97	40.28	0.0664/0.2804	32.07	31.97	0.2518/0.4768
DBSCAN	50.00	31.09	0.00/0.00	53.27	52.97	0.0845/0.3823
Graph-Cut	47.80	56.70	0.0030/0.2501	46.70	65.88	-0.1006/0.3799
N-Cut	58.56	59.32	0.0764/0.2957	-	-	-

to replace learned context-aware comparisons.

5.5 DINO-centric architecture

The feature enhancement block in the OddOneOut architecture leverages the rich semantic capabilities of DINOv2 features to improve the 2D feature maps generated by an existing encoder like ResNet50 through a distillation process. However, this distillation is computationally expensive due to the 3D nature of the data: to mimic DINOv2 features from multiple viewpoints, a renderer must operate on DINOv2 feature volumes predicted by a $1\times1\times1$ convolution block. The high dimensionality of these features (384-dimensional) does not scale efficiently with the voxel grid resolution used to represent the scene, and significant training time is consumed generating renders required for comparison with the DINOv2 ground truth feature maps. This complexity arises because volume rendering is needed to reconstruct features from different perspectives, which is costly both in memory and computation, especially for high-resolution 3D grids. With the goal of solving

these challenges, we decided to replace the ResNet50 image encoder employed in the original architecture directly with DINOv2, which inherently provides richer and more robust semantic features. In order not to project the entire DINOv2 features, we employ 1x1x1 convolution to reduce the channels dimensionality (from 384 to 32) and at the same time upscaling the 2D feature maps with bilinear interpolation so that it matches the ones coming out from ResNet50.

This substitution eliminates the need for costly volume rendering of high-dimensional feature volumes from multiple viewpoints, significantly reducing computational overhead and training time. Leveraging DINOv2 also enhances feature generalization across different perspectives, improving the overall efficiency and effectiveness of the feature enhancement block.

Moreover, with the image encoder now replaced by DINOv2, we re-evaluate whether the projected DINOv2 features can still be directly leveraged by clustering algorithms as in Section 5.4. Since DINOv2 embeddings are semantically richer, we hypothesize that they may enable better separation between normal and anomalous objects even without the matching module. This also allows us to reassess the necessity of additional context-aware mechanisms by isolating the contribution of the improved encoder alone.

5.5.1 Discussion and Results

We used DINOv2 as image encoder in three different setups:

- Dinov2 is simply substituted inside the original architecture and both training stages (one for the reconstruction and the other for the matching module part) are performed, therefore including also distillation.
- Dinov2 is employed,both training stages are performed but without distillation.
- Dinov2 is employed and only the second training stage is actually performed, without needing to train the reconstruction module and exploiting additional distillation.

In our analysis we feed the models with the ground truth location of the objects and focus only on the odd identification performance. This is justified by observations of the previous work [56] about the negligibility of the localization error due to relative simplicity of the setting.

By looking at the results in 5.6, we notice that exploiting DINOv2 features directly without distillation process and geometry reconstruction, yields better performance in identifying anomalous objects inside a scene, both for Toys and Parts datasets. For this reason, all the other experiments will follow this particular setup (therefore only the classification loss is retained).

Table 5.6: Performance results when leveraging different parts of the original [56] architecture

Toys Unseen		
Feature Encoder	**AUC**	**ACC**
Dinov2 + Distil-Dino-Vox	91.47	85.66
Dinov2 + No Distill	89.41	84.76
Dinov2 + No Recon + No Distill	92.63	84.88

However, the objects feature volumes alone are still insufficient for determining the anomaliness of an object with standard clustering algorithms as shown in 5.7.

Table 5.7: DINOv2 features clustering capabilities

Toys Unseen				
Odd Detector	**Distance Metric**	**AUC**	**ACC**	**ARS/ std. dev.**
Anomaly score	cosine similarity	49.72	62.61	-0.0010/0.1364
KMeans	euclidean	36.98	45.44	-0.0337/0.0600
Spectral Clustering	euclidean	47.15	52.97	-0.0207/0.0830
DBSCAN	euclidean	50.57	34.09	-0.0059/0.0571
Graph-Cut	cosine similarity	51.18	59.82	0.0084/0.1350
N-Cut	cosine similarity	59.15	58.38	0.0383/0.1384

5.6 Contrastive Learning in Dino-centric architecture

Simply projecting multi-view feature maps on a voxel grid has shown in the previous experiments about clustering algorithms, to be not enough for encoding discriminative information. Such behaviour happens due to the high dimensional nature of voxel data, since modeling such feature distribution is not trivial. Therefore, our main goal is to enhance this kind of representation, by separating as much as we can the features belonging to normal instances from the anomalous ones. Supervised contrastive loss can be a solution in this case by putting together normal objects in feature space, while pushing apart the anomalies. In particular, we do not mind about where the anomalies are scattered in feature space and clustering them, but rather to only cluster normal objects. To this end, we employ a InfoNCE

loss [84] [85], which enforces the embeddings of similar (positive) samples to be close and those of different (negative) samples to be distant. A query embedding, which in our case is a normal object representation, is compared with one positive key and with one or more negative keys. In our setting we consider the objects embeddings, within a scene, that belong to the normal class as positive keys while the one belonging to the anomalous class as negative keys.

$$\mathcal{L}_{\text{InfoNCE}} = -\log \frac{\exp(\text{sim}(\mathbf{q}, \mathbf{k}^+)/\tau)}{\exp(\text{sim}(\mathbf{q}, \mathbf{k}^+)/\tau) + \sum_{j=1}^{K} \exp(\text{sim}(\mathbf{q}, \mathbf{k_j})/\tau)}$$

- $\mathbf{q}$: Query embedding (e.g., from an anchor sample)
- $\mathbf{k}^+$: Positive key
- $\mathbf{k_j}$: Negative key
- $\text{sim}(\cdot, \cdot)$: Similarity function, typically cosine similarity
- τ: Temperature scaling factor
- K: Number of negative samples

In addition, two possible ways can be used for handling negative pairs:

- If **paired**, then each query sample is paired with a number of negative keys. It is comparable to a triplet loss, but with multiple negatives per sample.

- If **unpaired**, then the set of negative keys are all unrelated to any positive key.

We choose in our particular framework to use unpaired mode, due to the scarcity of negative samples inside each scene (the positive samples represent the majority inside each scene).

Unfortunately, due to the high dimensional features of individual objects encoded through voxel grids, contrastive loss is not directly usable on them. For this reason, we decide to take the final objects compact representation before being projected into logit space for classification. In this way, operating on much more compact representation is more doable and can let us employ our contrastive loss approach. The features obtained at this point of the architecture are further refined through an MLP (contrastive head) and then fed to the loss computation.

5.6.1 Discussion and Results

Despite the theoretical advantages of supervised contrastive learning, our empirical results did not show improvement over the previous baseline without contrastive loss, as shown in 5.8. Only a slight improvement on Parts dataset has been notices, but it is not significant.

Table 5.8: Performance results when using DINOv2 as feature encoder along with contrastive loss on the compact object features before classification

	Toys Unseen		**Parts**	
Note	**AUC**	**ACC**	**AUC**	**ACC**
Classification + Contrastive Learning	92.93	85.50	94.17	91.76
Classification Loss Only	90.40	84.17	94.27	90.05

We hypothesize several possible reasons for this outcome:

- **Lack of negative samples:** Each scene typically contains only one or very few anomalous instances, making the contrastive loss underconstrained in terms of meaningful negative sampling. Contrastive methods typically rely on rich negative sets to learn robust representations. In supervised contrastive setups, the imbalance between the number of normal and anomalous samples may cause the model to underestimate minority (anomalous) samples or bias the embedding space toward normal clusters, reducing sensitivity to true anomalies.

- **Dynamic definition of anomaly:** Since anomaly is defined in a scene-dependent way (i.e., based on context rather than absolute features), enforcing strict separation in feature space may not generalize well. The boundary between normal and anomalous shifts across scenes, making a single discriminative space hard to learn.

- **Which feature to select:** Choosing the right level in the network to apply contrastive loss is critical. Although we use compact object embeddings, it is possible that this representation is not sufficiently disentangled to benefit from contrastive objectives, or even too compressed to preserve meaningful differences.

Overall, while supervised contrastive learning remains a promising direction, our findings suggest that it requires careful adaptation when applied in scene-dependent anomaly detection with few negatives.

5.7 Contrastive Learning in more compact DINO-centric architecture

One of the primary drawbacks of employing sparse attention mechanisms, such as the approach described in [56], is the significant computational overhead they introduce. Sparse attention requires operations over high-dimensional data, which restricts the feasible batch size and substantially increases training time. This becomes particularly problematic when scaling to large datasets or deploying models in resource-constrained environments.

Additionally, high-dimensional representations can inhibit the application of advanced contrastive learning techniques, which often benefit from more compact and discriminative feature embeddings. Compact embeddings not only facilitate more efficient storage and faster computation but also enhance the ability to leverage a broader range of contrastive objectives, ultimately improving representation quality and downstream performance.

5.7.1 Context Match Head

To overcome these challenges, our objective is to develop a compact, one-dimensional representation that captures both spatial and semantic information. In our case, we squeeze the spatial dimensions of the objects voxel representations through a mean operation and project them into a 256 dimensional space with a linear layer. Such a representation enables efficient comparison between object embeddings without relying on voxel-based data, which can be cumbersome and computationally expensive. By embedding rich object information into a streamlined vector, we could therefore use more scalable and flexible learning frameworks.

To this end, we replace the original matching module, previously based on a sparse attention mechanism, with a Transformer Encoder that uses 3 layered multi-head attention with 8 different 32-dimensional heads.

The Transformer treats the objects 256-dimensional feature vectors as a sequence of tokens, allowing it to model complex relationships between features efficiently. Transformers excel at capturing contextual dependencies through their self-attention mechanism. Self-attention enables the model to weight the importance of each token relative to every other token in the sequence, effectively allowing it to aggregate

information from the entire input when forming each output representation. This property is crucial for context awareness, as it ensures that each feature vector is informed by the global structure and semantics present in the data [86].

5.7.2 New techniques for enhancing Contrastive Learning

Now that we have a more compact and suitable features to operate contrastive learning on, we can exploit further techniques that can enhance the discriminative power of object features:

- **Synthetic hard negative mining**: in contrastive learning, hard negative samples are critical for enhancing model performance. They are challenging samples that are similar to positive samples, but different enough to guide the model into learning distinctive features. They also prevent the model to easily minimize a loss due to the abundance of 'trivial' negative samples (the ones that are already distant in feature space) [87].

  We follow the approach mentioned in [87], where six methods are presented in order to synthetically generate hard negative samples $\mathbf{h}_i$ directly in feature space, by using query (positive) $\mathbf{q}$ and negative $\mathbf{n}$ samples:

  - **Interpolated Negatives** are generated by interpolating between the query feature vector $\mathbf{q}$ and negative $\mathbf{n}$ samples:

    $$\mathbf{h}_k^1 = \alpha_k \mathbf{q} + (1 - \alpha_k)\mathbf{n}_i, \quad \alpha_k \sim \mathcal{U}(0, \alpha_{\max})$$

    where $\alpha_{\max}$ is typically set to 0.5 to ensure the synthetic negative lie closer to the hard negative.

  - **Extrapolated Negatives**: Extrapolation generates a synthetic embedding that lies beyond the query embedding in the direction of the hardest negative.

    $$\mathbf{h}_{\text{extra}} = \mathbf{n} + \beta(\mathbf{n} - \mathbf{q}), \quad \beta > 0$$

  - **Noise-Injected Negatives**: Gaussian noise

    $$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 \mathbf{I})$$

    is added to a hard negative:

    $$\mathbf{h}_{\text{noise}} = \mathbf{n} + \boldsymbol{\epsilon}$$

  - **Perturbed Negatives**: Hard negatives are perturbed along the gradient of similarity loss

    $$\nabla_{\mathbf{h}} \mathcal{L}$$

to increase difficulty:

$$\mathbf{h}_{\text{perturb}} = \mathbf{n} + \eta \frac{\nabla_{\mathbf{n}} \mathcal{L}}{\|\nabla_{\mathbf{n}} \mathcal{L}\|}, \quad \eta > 0$$

– **Adversarial Negatives**: Adversarial perturbations maximize similarity to the query by solving:

$$\mathbf{h}_{\text{adv}} = \arg\max_{\mathbf{n}} \ \text{sim}(\mathbf{q}, \mathbf{n}) \quad \text{s.t.} \quad \|\mathbf{n} - \mathbf{h}\| \leq \delta$$

where δ constrains perturbation magnitude.

- **Hard positive mining**: Hard positives are generated to increase view variance within positive pairs, creating more challenging examples that help the model learn invariant and discriminative features [88]. By synthesizing positives that lie near the boundary of the positive class distribution, the model is better equipped to distinguish subtle differences and improve robustness. To improve the robustness of learned representations, we generate hard positives by extrapolating between a positive feature vector $\mathbf{p}_i$ and another randomly selected positive vector $\mathbf{p}_j$ from the same scene, where $i \neq j$. The hard positive is computed as:

$$\mathbf{p}_i^{\text{hard}} = \sigma \mathbf{p}_i + (1 - \sigma) \mathbf{p}_j,$$

with $\sigma = 1.5$ controlling the degree of extrapolation. This operation pushes the new sample beyond the original feature $\mathbf{p}_i$, creating a more challenging positive example. The resulting vector is then normalized to unit length:

$$\mathbf{p}_i^{\text{hard}} \leftarrow \frac{\mathbf{p}_i^{\text{hard}}}{\|\mathbf{p}_i^{\text{hard}}\|_2}.$$

- **Memory banks**: instead of synthetically generate new samples, we employ a batch-wise memory bank for hard negatives. Practically, we consider as negative samples also all the other ones in the scenes belonging to the current batch of data. This can be done since the newly introduced Context Match Head can handle bigger batches thanks to the more compact object representations.

5.7.3 Discussion and Results

Our results reveal that the Context Match Head, when trained with contrastive learning and without any tricks, achieves competitive performance with a much simpler setting w.r.t. the one employed in the original matching module as shown in 5.9. This suggests that even when reducing the input to a compact 256-dimensional representation, the model retains enough discriminative power to separate normal from anomalous instances effectively.

Table 5.9: Performance evaluation of Context Match Module, when different techniques for enhancing contrastive learning are employed. The basic contrastive learning setup is the one that performs the best.

	Toys Unseen	
Note	**AUC**	**ACC**
Context Match + basic CL	86.22	79.47
Context Match + Synth. Hard Neg. (UNPAIRED)	77.62	56.93
Context Match + Synth. Hard Neg. (PAIRED)	52.57	34.49
Context Match + Mem. Bank	86.41	79.25
Context Match + Mem. Bank + Hard Pos.	87.42	80.13

Injecting synthetic hard negatives actually degraded performance in our case. We hypothesize this is due to the difficulty of generating meaningful and semantically consistent anomalies in such a compressed latent space. In 256-dimensional space, perturbation can easily produce features that are either too unrealistic or fall outside the true distribution of plausible anomalies, leading to confused gradients and poor generalization.

Moreover, using memory banks instead does not give better results than the classic contrastive learning approach.

All this considerations highlight the fact that contrastive learning struggles to capture the dynamic nature of anomaly boundaries. In complex scenes where what is considered "normal" or "anomalous" can shift depending on context, fixed negative sampling or static memory banks fail to adapt. The assumption that a single global boundary can separate all anomalies from normal instances does not hold in such scenarios, especially in compact feature spaces, where subtle changes can carry significant semantic weight.

This underscores the need for more adaptive mechanisms that can reshape the decision boundary based on scene-level or object-level context. Without such dynamic modeling, contrastive learning may lead to over-simplified representations.

5.8 Conditioning objects embeddings on external tokens

Building on top of the Context Match Head, we want to enhance objects embeddings with more context-aware information. Since exploiting only the object embeddings inside the transformer encoder does not manage to give enough contextual information, we decide to inject inside the token sequence an additional token that represents an external conditioning signal. This conditioning signal should embed the concept of normalcy inside the scene and can be obtained in many ways. The self-attention mechanism is well suited for this kind of scenarios, as it allows the model to dynamically attend to the conditioning token and modulate object representations based on the broader scene context, effectively guiding the model toward more informed and adaptive anomaly decisions.

5.8.1 Conditioning on Dinov2 CLS token

The CLS token in Vision Transformers naturally absorbs domain-specific information during source-domain training, acting as a compact representation of the domain characteristics embedded in the data. This phenomenon occurs because the CLS token captures the low-frequency components of images, which correspond to domain features in the Fourier frequency space [89].

This makes the DINOv2 CLS token suitable for describing the context of a scene and therefore can be a valuable conditioning signal for enhancing scene-specific awareness. Since for each view of the scene we have a CLS feature vector, we average them in order to have a global prototype that represents the overall scene. Since the CLS token is 384-dimensional feature vector, we project it to the tokens 256-dimensional space.

We condition the input tokens (objects) by simply adding the averaged CLS token at the beginning of the sequence. The conditioning token is then discarded after the Context Match Head and only the objects embeddings are retained.

5.8.2 Learning normalcy via an external learnable token

Instead of enhancing contextual awareness by injecting an external token that embeds scene-level semantics, we propose a different approach: introducing a learnable token that represents a "normalcy prototype" of the scene. Rather than conditioning object embeddings on a predefined notion of normality, we allow this token to absorb and summarize the dominant characteristics of the scene directly from the object embeddings through the self-attention mechanism.

At the end of the Context Match module, we discard the individual object

embeddings and retain only this learned token, which serves as a compact centroid-like representation of normalcy for the given scene. This token can then be used as a reference point to compare with other object embeddings, enabling anomaly detection based on deviation from the learned scene prototype.

Residual Anomaly Module

We can boost the identification of scene-specific anomalies by guiding the model to focus on how much an object deviates from the average normality. This deviation can be encoded into discriminative features via a residual anomaly module. We propose to devise it by adding to the Context Match module a 1-layer transformer encoder that takes as input the objects' representations $\mathcal{O}$ together with a learnable token $\mathbf{t}_0 \in \mathbb{R}^d$. The latter attends to all object embeddings, and the corresponding output $z_0 \in \mathbb{R}^q$ is the only token retained for further processing. Specifically, it acts as a proxy for the scene-specific normality prototype and is supervised through a mean squared error loss

$$\mathcal{L}_{\text{normality}} = \|\bar{z} - z_0\|_2^2 \ , \tag{5.1}$$

where $\bar{z} = \frac{1}{|\{i|y_i=0\}|} \sum_{i:y_i=0} z_i$ is the average embedding of the ground-truth normal objects. Moreover, the residual anomaly module refines each object embedding via a projection head $f(z_i)$ (composed of linear layer, LayerNorm, and ReLU) to then calculate the difference between the obtained object representations and the learned centroid $r_i = f(z_i) - z_0$.

Finally, a linear layer elaborates on r_i to produce the anomaly score, representing the probability that a given input is anomalous. Thus, when the residual anomaly head is active, the network is trained by minimizing $\mathcal{L}_{\text{BCE}} + \mathcal{L}_{normality}$. An overview of the architecture is displayed in 5.3

5.8.3 Discussion and Results

The conditioning approach generally outperformed the contrastive learning-based method. While the CLS token achieved competitive results, the Residual Anomaly Module yielded the best performance, particularly on the Parts dataset 5.10. This improvement may be attributed to the nature of the CLS token, which tends to capture coarse-grained scene-level context and so might not align well with the fine-grained representations required for distinguishing individual objects.

We show a breakdown of the Accuracy of the Residual Anomaly Module across different anomaly types in Fig. 5.4 (a), together with success and failure qualitative results in Fig. 5.5. We can see from the breakdown that the Residual Anomaly Module is particularly good at detecting material issues, bumps, and fractures that usually consist of darker regions of the object, while it struggles with 3D specific

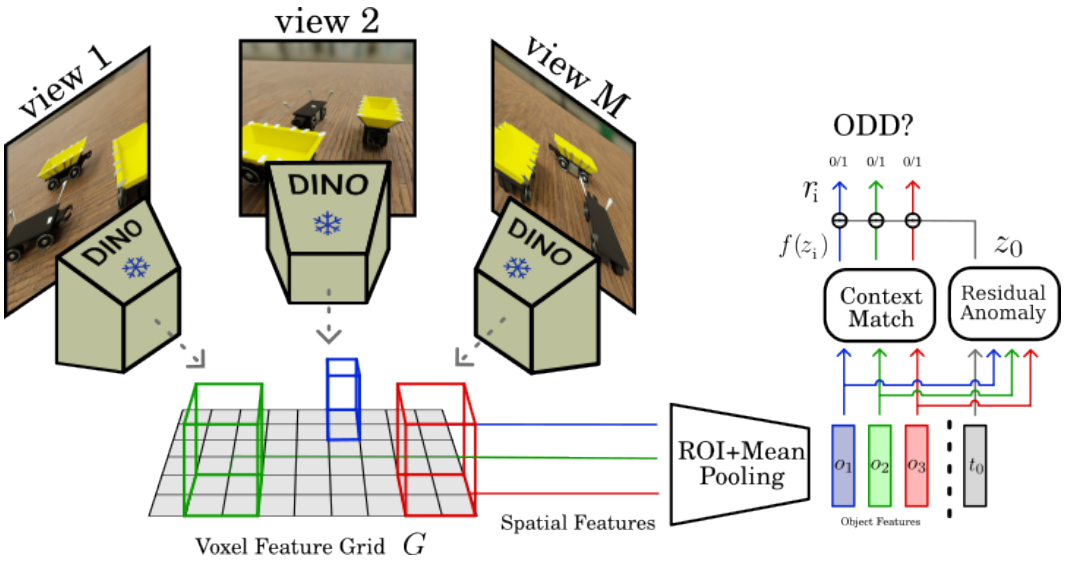

Figure 5.3: Schematic representation of our approach. It takes as input M views of the scene, which are processed using the DINOv2 encoder. The resulting features are projected onto a voxel grid and associated with each of the $N = 3$ objects. A subsequent pooling step yields per-object representations. These are further refined by the context and residual anomaly heads, which encode both object-to-object similarity and the relative deviations of each object from the scene-specific average normalcy.

Table 5.10: Results using the Context Match Head with the conditioning approach. Residual Anomaly Head reaches the best performance, in particular on Toys dataset

	Toys Seen		Toys Unseen		Parts	
Conditioning type	**SEEN AUC**	**SEEN ACC ACC**	**UNSEEN AUC**	**UNSEEN ACC**	**AUC**	**ACC**
Residual Anomaly	89.49	84.86	85.57	81.43	89.72	88.81
CLS token	88.10	83.49	84.33	80.95	87.53	88.64

anomalies like missing parts, translations, and deformations that require a more global 3D reasoning.

Furthermore, the right part of Fig. 5.4 shows the AUC of the model when varying the number of input views during testing (b) and the performance when changing the object count (c). Both results are obtained on the Parts Unseen dataset and highlight the advantage provided by a growing amount of images as well as the relative robustness across scenes becoming progressively more difficult with a range of 4% point variation around the main result (Accuracy 88.81 in Table 5.11).

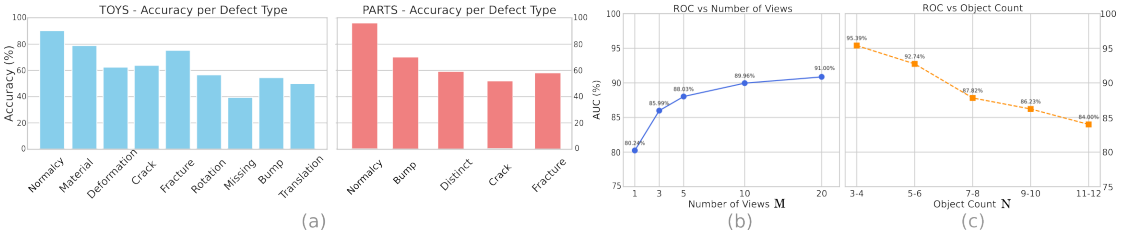

Figure 5.4: We report (a) accuracy across different anomaly types and normalcy data on Toys Unseen and Parts Unseen; AUC on Parts Unseen when changing (b) the number of views at inference time and (c) the number of objects in the scene.

Table 5.11: Main results for the three benchmark tracks. Ours presents comparable performance to that of the leading competitor (OOO) on Toys and largely outperforms it on the challenging Parts dataset.

	Toys Seen		Toys Unseen		Parts	
Model	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**
ImVoxelNet [90]	89.49	84.86	**85.57**	**81.43**	**89.72**	**88.81**
DETR3D [59]	88.10	83.49	84.33	80.95	87.53	88.64
OOO [56]	**91.78**	83.21	**89.15**	**81.57**	86.12	79.68
MLLM baseline	-	52.23	-	53.35	-	60.73
Residual Anomaly Module	89.49	**84.86**	85.57	81.43	**89.72**	**88.81**

Table 5.12: Ablation results across the three benchmark tracks for various versions of our model's head.

	Toys Seen		Toys Unseen		Parts		Memory	Inf. time
Our Head	**AUC**	**ACC**	**AUC**	**ACC**	**AUC**	**ACC**	**(GB)**	**(ms)**
Sparse Voxel Attn.	86.11	78.79	85.48	77.32	86.32	84.06	1.23	337
Context	89.18	**85.42**	85.09	81.27	89.14	88.65	**0.36**	**271**
Context + Residual	**89.49**	84.86	**85.57**	**81.43**	**89.72**	**88.81**	0.54	286

In Table 5.12 we compare different versions of our model, also ablating the Residual Anomaly Head. From the obtained results we can conclude that the sparse voxel attention performs worse than the other versions on average. Moreover, it consumes significantly more memory and is considerably slower. Adding the Residual Anomaly Module slightly boosts the performance with a minimal change in memory and inference time. The sparse version of the model also uses significantly more parameters (72M vs. 48M) and requires much longer training time, lasting about 6h

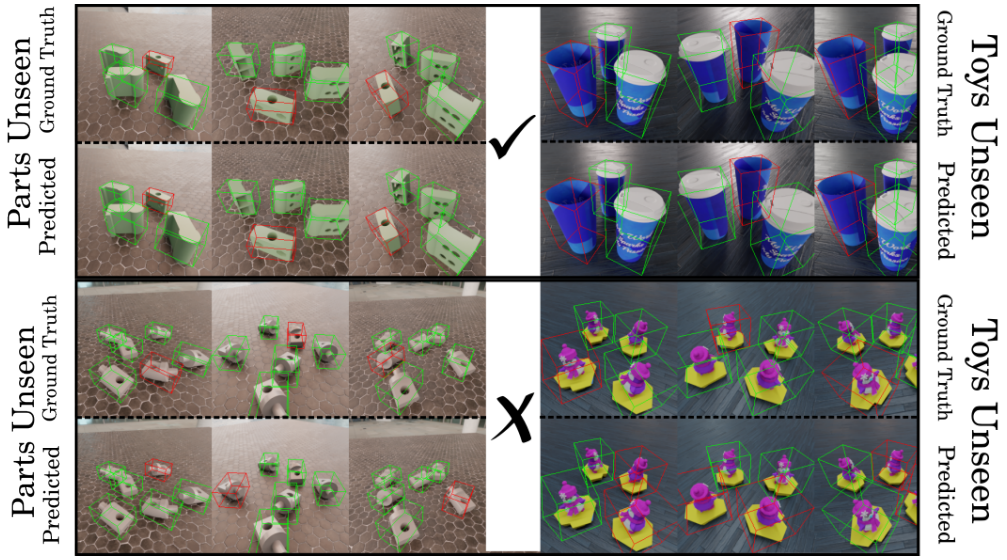

Figure 5.5: Qualitative results of our approach on Toys Unseen and Parts Unseen. We show three views for each scene. The green boxes indicate normal objects while the red ones indicate anomalous objects.

on Toys and 22h on Parts (our models take around 2.5h on Toys and 5.5h on Parts).

5.9 Qualitative Analysis

The following represents

5.10 MLLM Baseline

The MLLM baseline falls short on the Toys tracks, and on Parts it gets accuracy results close to those of the detection methods ImVoxelsNet and DETR3D as shown in 5.11. Overall, MLLM shows a bias towards normal data that might have been part of its large scale pre-training and, similarly to the detection methods, it recognizes large cracks and fractures but struggles when intra-group comparisons over multiple views are necessary. These limitations align with what was discussed in [76] and call for new tailored modules to guide generalist models when dealing with tasks requiring multi-view consistency.

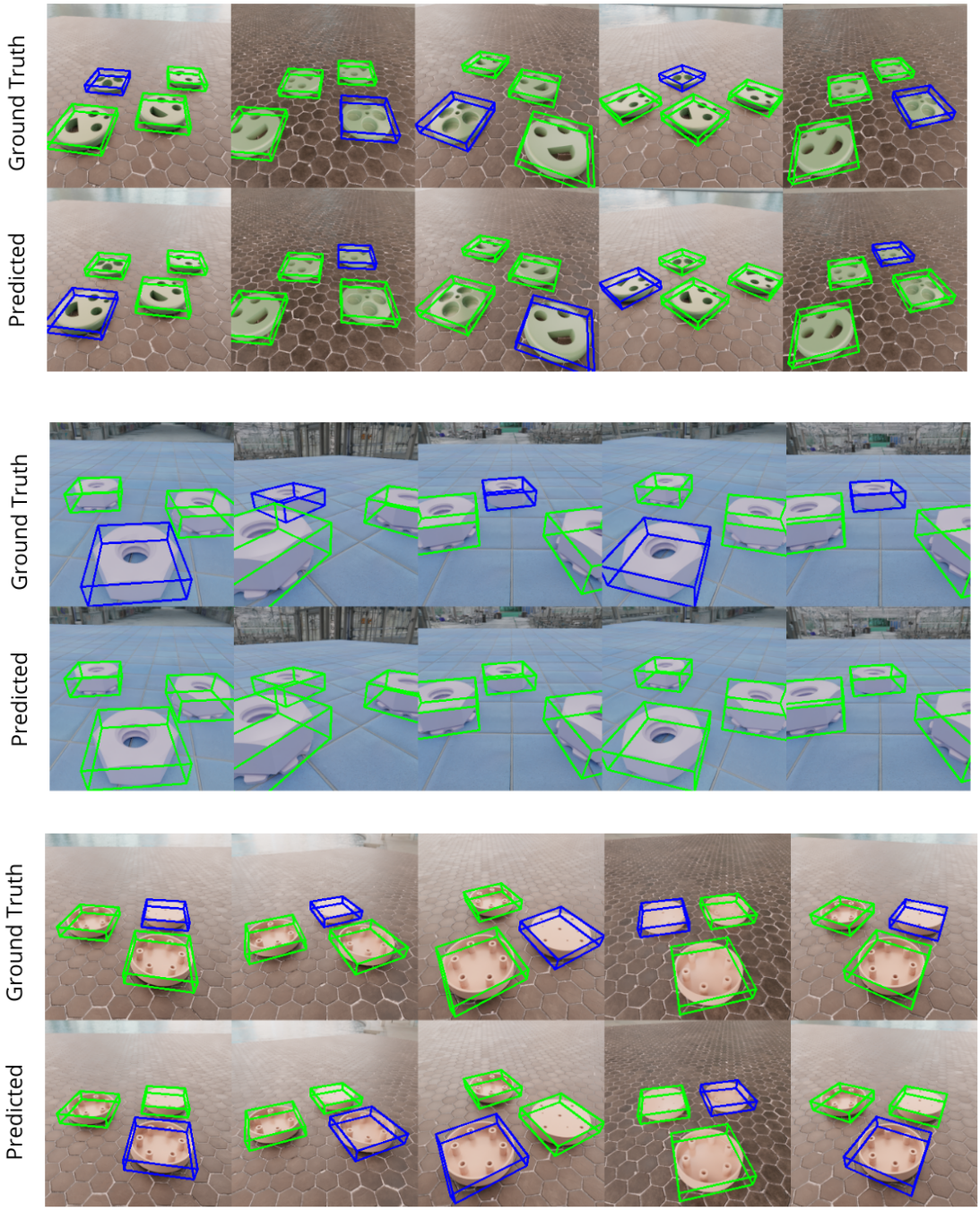

Figure 5.6: Incorrect predictions on Parts dataset

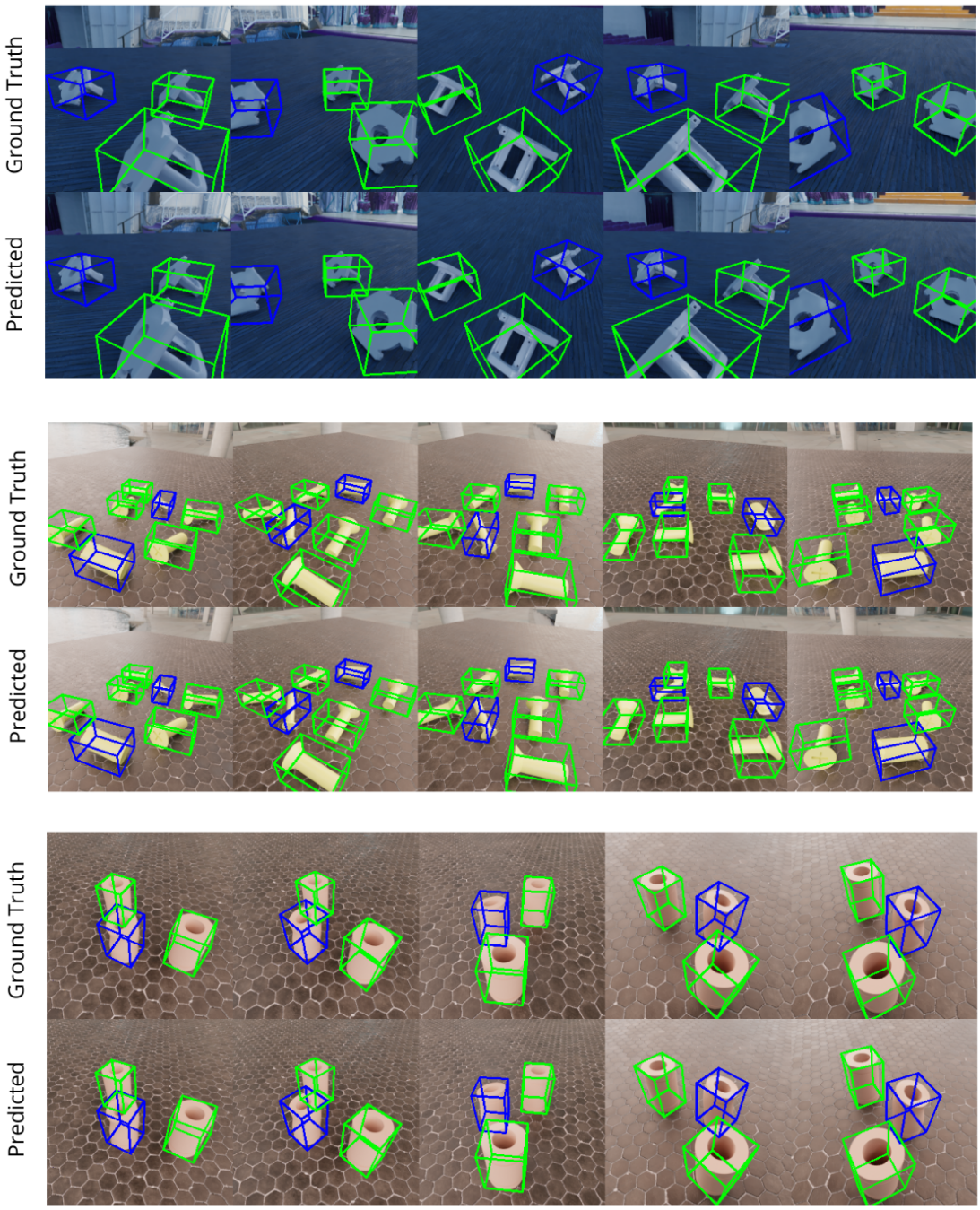

Figure 5.7: Good predictions on Parts dataset

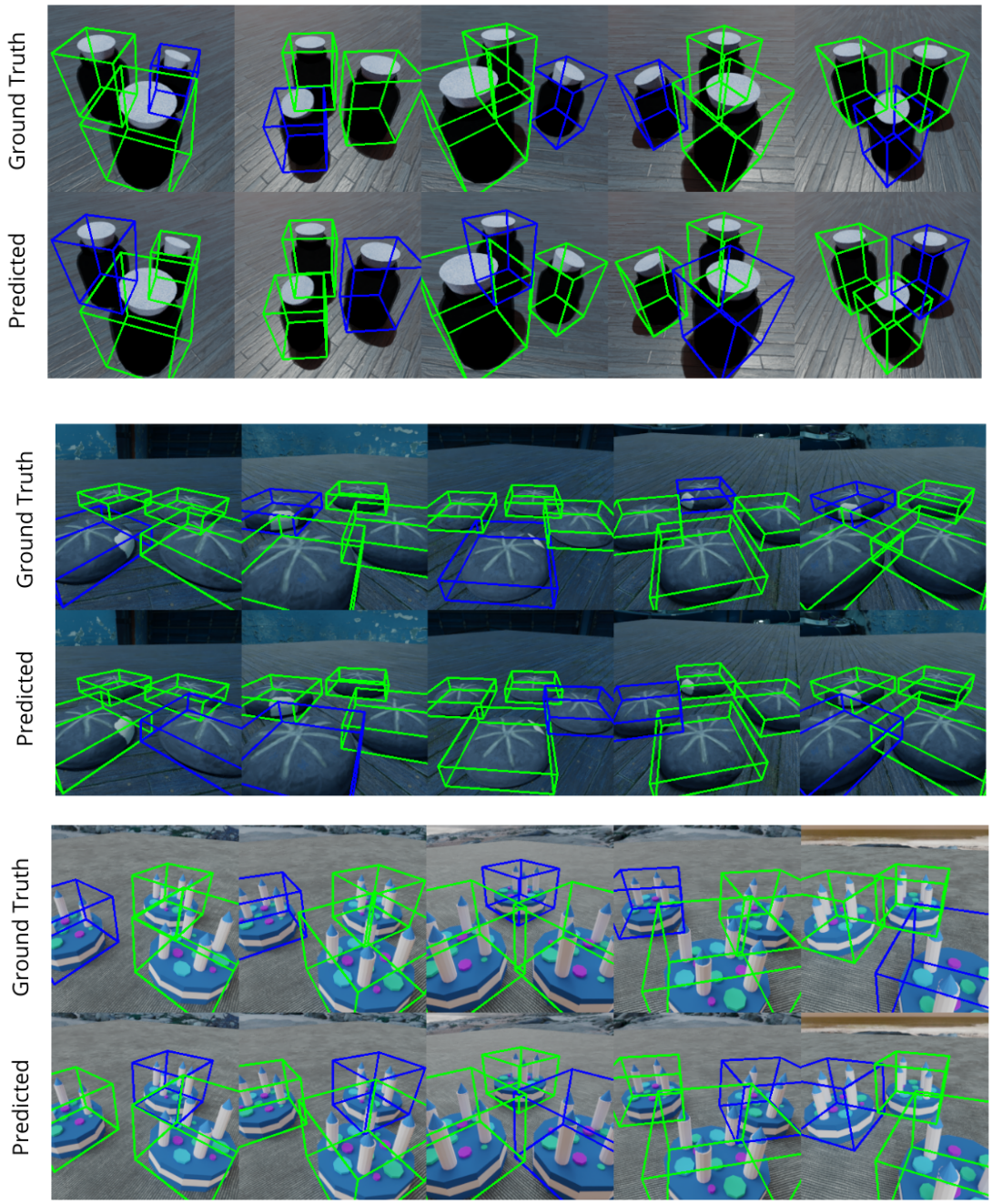

Figure 5.8: Incorrect predictions on Toys

5.11 3D Object Detection

Alongside the 3D detection method proposed in [56], we explored an alternative pipeline that operates directly on multi-view images. Our approach leverages

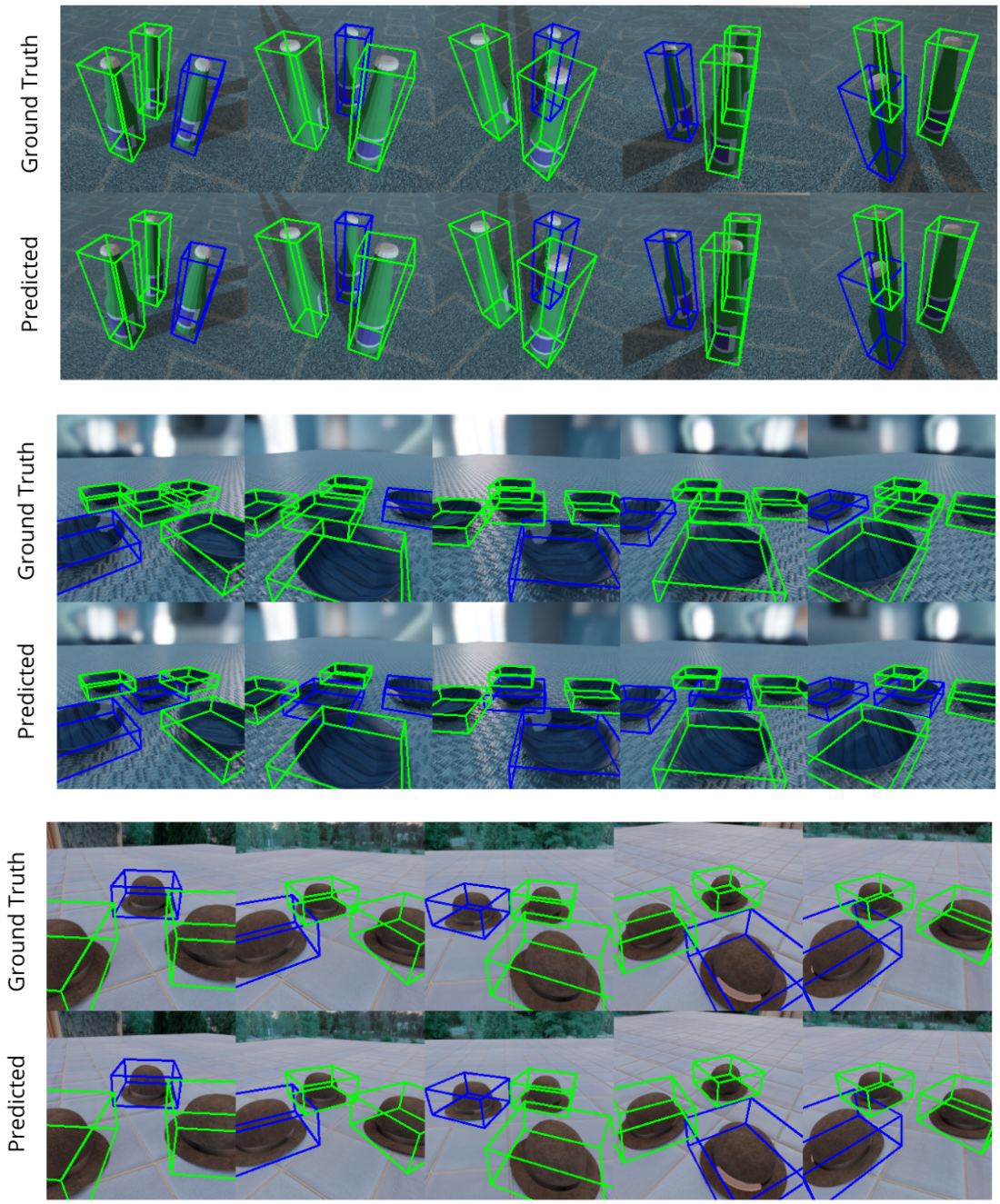

Figure 5.9: Good predictions on Toys dataset

Grounding DINO [91], a foundational model for open-set object detection, to predict 2D bounding boxes in each view. However, localizing objects in individual

views is not sufficient—we must also establish correspondence across views to recover 3D localization. Specifically, to compute 3D bounding boxes by projecting detections into a voxel grid, we need consistent detections for each object across all views.

To address this, we employ SAM2 [92], a model effective at tracking and segmenting objects across video frames given initial guidance. By treating the multi-view images as frames of a video, we use SAM2 to propagate the initial detection across views, enabling consistent object segmentation. These multi-view masks are then backprojected into a voxel grid representing the scene, from which we estimate the 3D bounding boxes.

Unfortunately, this method faces some important limitations:

- Grounding DINO may fail to detect certain object categories or produce imprecise boxes.
- Occlusions across views can hinder SAM2's ability to track objects consistently.

As a result, early detection errors from Grounding DINO propagate through the pipeline, leading to poor final 3D IoU scores. While promising in concept, this method remains highly sensitive to its initial detection quality.

Chapter 6

Conclusion

6.1 Conclusions

In this work we focused on the odd-one-out anomaly detection task that poses several challenges by moving anomaly detection closer to real world conditions. It involves multi-object scenes with scene-specific anomaly criteria that can only be inferred by cross-instance comparisons. This problem calls for models able to manage spatial reasoning to elaborate on multiple scene views, along with relational reasoning to interpret the context and generalize over different object categories and layouts. Considering the relevance of the task for industrial applications, efficiency is also a key requirement. By focusing on this aspect, we presented a model that, despite the significantly low capacity (number of parameters) and memory needs, gets competitive performance compared to the state of the art. Several in-depth analyses confirmed the effectiveness of the adopted architectural choices as well as the overall model's robustness.

We explored different techniques for enhancing feature discriminativeness in contrastive learning, but the task revealed really challenging especially due to the non fixed decision boundary that exist between normal and anomalous objects.

To test the zero-shot effectiveness of Multimodal Large Language Models on the odd-one-out task, we also introduced a baseline that leverages the Set-of-Mark prompting method to support visual grounding. The low experimental results revealed the limitations of this approach, signaling the need for new solutions to leverage the extensive internal knowledge of generalist models when performing fine-grained comparisons in complex multi-view settings.

We believe that future solutions for the odd-one-out task should continue to build on the capabilities of pre-trained foundation models, with efficiency remaining a key objective. This would enable the transition from anomaly detection to full 3D anomaly localization, while allowing models to articulate the rationale behind

the predictions through natural language. Furthermore, the task could be extended to encompass logical and functional aspects, bridging perception and robotics, and evolving into a true intelligence test for autonomous agents.

The work developed around the final Residual Anomaly architecture has been accepted to an italian computer vision conference, underscoring its scientific contribution and practical relevance: **Silvio Chito, Paolo Rabino, Tatiana Tommasi Efficient Odd-One-Out Anomaly Detection» accepted at ICIAP 2025 - the International Conference on Image Analysis and Processing**

6.2 Future Directions

One promising path for future research lies in leveraging vision-language models (VLMs) to solve the odd-one-out task. VLMs, with their strong capability to align visual content with semantic concepts, offer a great fit for identifying objects that deviate from a given scene's context. This direction also connects closely with recent advances in 3D object detection and identification using VLMs. Extending these methods to support contextual anomaly detection, where the anomaly is defined not by absolute categories but by deviation from the majority in a scene, could unlock more general and scalable anomaly detection pipelines.

Bibliography

[1] Hui Zhang et al. *Prototypical Residual Networks for Anomaly Detection and Localization*. 2023. arXiv: 2212.02031 [cs.CV]. URL: https://arxiv.org/abs/2212.02031 (cit. on p. 7).

[2] Choubo Ding, Guansong Pang, and Chunhua Shen. *Catching Both Gray and Black Swans: Open-set Supervised Anomaly Detection*. 2022. arXiv: 2203.14506 [cs.CV]. URL: https://arxiv.org/abs/2203.14506 (cit. on p. 7).

[3] Yunkang Cao et al. «Collaborative Discrepancy Optimization for Reliable Image Anomaly Localization». In: *IEEE Transactions on Industrial Informatics* 19.11 (2023), pp. 10674–10683 (cit. on p. 7).

[4] Karsten Roth et al. *Towards Total Recall in Industrial Anomaly Detection*. 2022. arXiv: 2106.08265 [cs.CV]. URL: https://arxiv.org/abs/2106.08265 (cit. on pp. 7, 22).

[5] Yong Shi, Jie Yang, and Zhiquan Qi. «Unsupervised anomaly segmentation via deep feature reconstruction». In: *Neurocomputing* 424 (2021), pp. 9–22 (cit. on p. 7).

[6] Chaoqin Huang et al. «Registration based Few-Shot Anomaly Detection». In: *ECCV*. 2022 (cit. on p. 7).

[7] Guoyang Xie et al. *Pushing the Limits of Fewshot Anomaly Detection in Industry Vision: Graphcore*. 2023. arXiv: 2301.12082 [cs.CV]. URL: https://arxiv.org/abs/2301.12082 (cit. on p. 7).

[8] Jongheon Jeong et al. *WinCLIP: Zero-/Few-Shot Anomaly Classification and Segmentation*. 2023. arXiv: 2303.14814 [cs.CV]. URL: https://arxiv.org/abs/2303.14814 (cit. on p. 7).

[9] Xuhai Chen, Yue Han, and Jiangning Zhang. *APRIL-GAN: A Zero-/Few-Shot Anomaly Classification and Segmentation Method for CVPR 2023 VAND Workshop Challenge Tracks 1 and 2: 1st Place on Zero-shot AD and 4th Place on Few-shot AD*. 2023. arXiv: 2305.17382 [cs.CV]. URL: https://arxiv.org/abs/2305.17382 (cit. on p. 7).

[10] Qihang Zhou et al. *AnomalyCLIP: Object-agnostic Prompt Learning for Zero-shot Anomaly Detection*. 2025. arXiv: 2310.18961 [cs.CV]. URL: https://arxiv.org/abs/2310.18961 (cit. on p. 7).

[11] Yunkang Cao et al. *A Survey on Visual Anomaly Detection: Challenge, Approach, and Prospect*. 2024. arXiv: 2401.16402 [cs.CV]. URL: https://arxiv.org/abs/2401.16402 (cit. on pp. 7, 8).

[12] Paul Bergmann et al. «MVTec AD — A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection». In: *CVPR*. 2019 (cit. on p. 8).

[13] Paul Bergmann et al. «Beyond Dents and Scratches: Logical Constraints in Unsupervised Anomaly Detection and Localization». In: *International Journal of Computer Vision* 130.8 (2022), pp. 2069–2089 (cit. on p. 8).

[14] Christopher Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer, 2007 (cit. on p. 8).

[15] Michael G. Khan Shehroz S.and Madden. «A Survey of Recent Trends in One Class Classification». In: *Artificial Intelligence and Cognitive Science*. 2010 (cit. on p. 8).

[16] Abiodun M. Ikotun et al. «K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data». In: *Information Sciences* 622 (2023), pp. 178–210 (cit. on p. 8).

[17] Harsh Vardhan Singh, Ashwin Girdhar, and Sonika Dahiya. «A Literature survey based on DBSCAN algorithms». In: *ICICCS*. 2022 (cit. on p. 8).

[18] Iwan Syarif, A. Prugel-Bennett, and Gary Wills. «Unsupervised Clustering Approach for Network Anomaly Detection». In: 2012 (cit. on p. 8).

[19] Amuthan Prabakar Muniyandi, R. Rajeswari, and R. Rajaram. «Network Anomaly Detection by Cascading K-Means Clustering and C4.5 Decision Tree algorithm». In: *Procedia Engineering* 30 (2012), pp. 174–182 (cit. on p. 8).

[20] Bernhard Schölkopf et al. «Estimating the Support of a High-Dimensional Distribution». In: *Neural Computation* 13 (2001), pp. 1443–1471 (cit. on p. 9).

[21] David M. J. Tax and Robert P. W. Duin. «Support Vector Data Description». In: *Machine Learning* 54.1 (2004), pp. 45–66 (cit. on p. 9).

[22] Haoqi Huang et al. *Deep Learning Advancements in Anomaly Detection: A Comprehensive Survey*. 2025. arXiv: 2503.13195 [cs.LG]. URL: https://arxiv.org/abs/2503.13195 (cit. on p. 9).

[23] Amir Markovitz et al. *Graph Embedded Pose Clustering for Anomaly Detection*. 2020. arXiv: 1912.11850 [cs.CV]. URL: https://arxiv.org/abs/1912.11850 (cit. on p. 9).

[24] Ivan Kobyzev, Simon J.D. Prince, and Marcus A. Brubaker. «Normalizing Flows: An Introduction and Review of Current Methods». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 43.11 (2021), pp. 3964–3979 (cit. on p. 9).

[25] Jiawei Yu et al. *FastFlow: Unsupervised Anomaly Detection and Localization via 2D Normalizing Flows*. 2021. arXiv: 2111.07677 [cs.CV]. URL: https://arxiv.org/abs/2111.07677 (cit. on p. 9).

[26] MyeongAh Cho et al. *Unsupervised Video Anomaly Detection via Normalizing Flows with Implicit Latent Features*. 2022. arXiv: 2010.07524 [cs.CV]. URL: https://arxiv.org/abs/2010.07524 (cit. on p. 9).

[27] Jonathan Pirnay and Keng Chai. «Inpainting Transformer for Anomaly Detection». In: *ICIAP*. 2022 (cit. on p. 9).

[28] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. «Reconstruction by inpainting for visual anomaly detection». In: *Pattern Recognition* 112 (2021), p. 107706 (cit. on p. 9).

[29] Hanqun Cao et al. « A Survey on Generative Diffusion Models ». In: *IEEE Transactions on Knowledge & Data Engineering* 36.07 (2024), pp. 2814–2830 (cit. on p. 10).

[30] Ian J. Goodfellow et al. *Generative Adversarial Networks*. 2014. arXiv: 1406.2661 [stat.ML]. URL: https://arxiv.org/abs/1406.2661 (cit. on p. 10).

[31] Ling Yang et al. *Diffusion Models: A Comprehensive Survey of Methods and Applications*. 2024. arXiv: 2209.00796 [cs.LG]. URL: https://arxiv.org/abs/2209.00796 (cit. on p. 10).

[32] Sam Bond-Taylor et al. «Deep Generative Modelling: A Comparative Review of VAEs, GANs, Normalizing Flows, Energy-Based and Autoregressive Models». In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 44.11 (2022), pp. 7327–7347 (cit. on p. 10).

[33] Zheng Fang et al. «FastRecon: Few-shot Industrial Anomaly Detection via Fast Feature Reconstruction». In: *ICCV*. 2023 (cit. on p. 10).

[34] Sukanya Patra and Souhaib Ben Taieb. «Revisiting Deep Feature Reconstruction for Logical and Structural Industrial Anomaly Detection». In: *TMLR* (2024) (cit. on p. 10).

[35] Paul Bergmann et al. « Uninformed Students: Student-Teacher Anomaly Detection With Discriminative Latent Embeddings ». In: *CVPR*. 2020 (cit. on p. 10).

[36] Mohammadreza Salehi et al. «Multiresolution Knowledge Distillation for Anomaly Detection». In: *CVPR*. 2021 (cit. on p. 10).

[37] Thomas Defard et al. «PaDiM: A Patch Distribution Modeling Framework for Anomaly Detection and Localization». In: *ICPR*. 2021 (cit. on p. 10).

[38] Karsten Roth et al. «Towards total recall in industrial anomaly detection». In: *CVPR*. 2022 (cit. on p. 10).

[39] Chun-Liang Li et al. «Cutpaste: Self-supervised learning for anomaly detection and localization». In: *CVPR*. 2021 (cit. on p. 10).

[40] Zhikang Liu et al. «Simplenet: A simple network for image anomaly detection and localization». In: *CVPR*. 2023 (cit. on p. 10).

[41] Luc PJ Sträter et al. «Generalad: Anomaly detection across domains by attending to distorted features». In: *ECCV*. 2025 (cit. on p. 10).

[42] Mathilde Caron et al. *Emerging Properties in Self-Supervised Vision Transformers*. 2021. arXiv: 2104.14294 [cs.CV]. URL: https://arxiv.org/abs/2104.14294 (cit. on p. 10).

[43] Maxime Oquab et al. «DINOv2: Learning Robust Visual Features without Supervision». In: *TMLR* (2024) (cit. on p. 10).

[44] Ting Chen et al. *A Simple Framework for Contrastive Learning of Visual Representations*. 2020. arXiv: 2002.05709 [cs.LG]. URL: https://arxiv.org/abs/2002.05709 (cit. on p. 10).

[45] Kaiming He et al. *Momentum Contrast for Unsupervised Visual Representation Learning*. 2020. arXiv: 1911.05722 [cs.CV]. URL: https://arxiv.org/abs/1911.05722 (cit. on p. 11).

[46] Fawaz Sammani, Boris Joukovsky, and Nikos Deligiannis. *Visualizing and Understanding Contrastive Learning*. 2023. arXiv: 2206.09753 [cs.CV]. URL: https://arxiv.org/abs/2206.09753 (cit. on p. 11).

[47] Prannay Khosla et al. *Supervised Contrastive Learning*. 2021. arXiv: 2004.11362 [cs.LG]. URL: https://arxiv.org/abs/2004.11362 (cit. on p. 11).

[48] Kaifang Long et al. *Revisiting Multimodal Fusion for 3D Anomaly Detection from an Architectural Perspective*. 2024. arXiv: 2412.17297 [cs.CV]. URL: https://arxiv.org/abs/2412.17297 (cit. on p. 11).

[49] Yuxuan Lin et al. *A Survey on RGB, 3D, and Multimodal Approaches for Unsupervised Industrial Image Anomaly Detection*. 2025. arXiv: 2410.21982 [cs.CV]. URL: https://arxiv.org/abs/2410.21982 (cit. on p. 11).

[50] Vitjan Zavrtanik, Matej Kristan, and Danijel Skočaj. *Cheating Depth: Enhancing 3D Surface Anomaly Detection via Depth Simulation*. 2023. arXiv: 2311.01117 [cs.CV]. URL: https://arxiv.org/abs/2311.01117 (cit. on p. 11).

[51] Kecen Li et al. *DAS3D: Dual-modality Anomaly Synthesis for 3D Anomaly Detection*. 2025. arXiv: 2410.09821 [cs.CV]. URL: https://arxiv.org/abs/2410.09821 (cit. on p. 11).

[52] Paul Bergmann et al. «The MVTec 3D-AD Dataset for Unsupervised 3D Anomaly Detection and Localization». In: *Proceedings of the 17th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*. SCITEPRESS - Science and Technology Publications, 2022. DOI: 10.5220/0010865000003124. URL: http://dx.doi.org/10.5220/0010865000003124 (cit. on p. 11).

[53] Zheyuan Zhou et al. *R3D-AD: Reconstruction via Diffusion for 3D Anomaly Detection*. 2024. arXiv: 2407.10862 [cs.CV]. URL: https://arxiv.org/abs/2407.10862 (cit. on p. 11).

[54] Jiaqi Liu et al. *Real3D-AD: A Dataset of Point Cloud Anomaly Detection*. 2023. arXiv: 2309.13226 [cs.CV]. URL: https://arxiv.org/abs/2309.13226 (cit. on p. 11).

[55] Wenqiao Li et al. *Towards Scalable 3D Anomaly Detection and Localization: A Benchmark via 3D Anomaly Synthesis and A Self-Supervised Learning Network*. 2023. arXiv: 2311.14897 [cs.CV]. URL: https://arxiv.org/abs/2311.14897 (cit. on p. 11).

[56] Ankan Bhunia, Changjian Li, and Hakan Bilen. *Odd-One-Out: Anomaly Detection by Comparing with Neighbors*. 2025. arXiv: 2406.20099 [cs.CV]. URL: https://arxiv.org/abs/2406.20099 (cit. on pp. 11, 14, 16–20, 25, 26, 29, 36, 40).

[57] Xiang Zhang, Hai Wang, and Haoran Dong. «A Survey of Deep Learning-Driven 3D Object Detection: Sensor Modalities, Technical Architectures, and Applications». In: *Sensors* 25.12 (2025). ISSN: 1424-8220. DOI: 10.3390/s25123668. URL: https://www.mdpi.com/1424-8220/25/12/3668 (cit. on p. 12).

[58] Yingfei Liu et al. *PETR: Position Embedding Transformation for Multi-View 3D Object Detection*. 2022. arXiv: 2203.05625 [cs.CV]. URL: https://arxiv.org/abs/2203.05625 (cit. on p. 12).

[59] Yue Wang et al. *DETR3D: 3D Object Detection from Multi-view Images via 3D-to-2D Queries*. 2021. arXiv: 2110.06922 [cs.CV]. URL: https://arxiv.org/abs/2110.06922 (cit. on pp. 12, 36).

[60] Yin Zhou and Oncel Tuzel. *VoxelNet: End-to-End Learning for Point Cloud Based 3D Object Detection*. 2017. arXiv: 1711.06396 [cs.CV]. URL: https://arxiv.org/abs/1711.06396 (cit. on p. 12).

[61] Alex H. Lang et al. *PointPillars: Fast Encoders for Object Detection from Point Clouds*. 2019. arXiv: 1812.05784 [cs.LG]. URL: https://arxiv.org/abs/1812.05784 (cit. on p. 12).

[62] Shaoshuai Shi et al. *PV-RCNN: Point-Voxel Feature Set Abstraction for 3D Object Detection*. 2021. arXiv: 1912.13192 [cs.CV]. URL: https://arxiv.org/abs/1912.13192 (cit. on p. 12).

[63] Yin Zhou et al. *End-to-End Multi-View Fusion for 3D Object Detection in LiDAR Point Clouds*. 2019. arXiv: 1910.06528 [cs.CV]. URL: https://arxiv.org/abs/1910.06528 (cit. on p. 12).

[64] Samuel Black and Richard Souvenir. «Multi-view Classification Using Hybrid Fusion and Mutual Distillation». In: *WACV*. 2024, pp. 269–279. URL: https://doi.org/10.1109/WACV57701.2024.00034 (cit. on p. 12).

[65] Duzhen Zhang et al. *MM-LLMs: Recent Advances in MultiModal Large Language Models*. 2024. arXiv: 2401.13601 [cs.CL]. URL: https://arxiv.org/abs/2401.13601 (cit. on p. 13).

[66] Kilian Carolan, Laura Fennelly, and Alan F. Smeaton. *A Review of Multi-Modal Large Language and Vision Models*. 2024. arXiv: 2404.01322 [cs.CL]. URL: https://arxiv.org/abs/2404.01322 (cit. on p. 13).

[67] Haotian Liu et al. *Visual Instruction Tuning*. 2023. arXiv: 2304.08485 [cs.CV]. URL: https://arxiv.org/abs/2304.08485 (cit. on p. 13).

[68] Deyao Zhu et al. *MiniGPT-4: Enhancing Vision-Language Understanding with Advanced Large Language Models*. 2023. arXiv: 2304.10592 [cs.CV]. URL: https://arxiv.org/abs/2304.10592 (cit. on p. 13).

[69] Anas Awadalla et al. *OpenFlamingo: An Open-Source Framework for Training Large Autoregressive Vision-Language Models*. 2023. arXiv: 2308.01390 [cs.CV]. URL: https://arxiv.org/abs/2308.01390 (cit. on p. 13).

[70] Quan Sun et al. *Emu: Generative Pretraining in Multimodality*. 2024. arXiv: 2307.05222 [cs.CV]. URL: https://arxiv.org/abs/2307.05222 (cit. on p. 13).

[71] Kaizhi Zheng, Xuehai He, and Xin Eric Wang. *MiniGPT-5: Interleaved Vision-and-Language Generation via Generative Vokens*. 2024. arXiv: 2310.02239 [cs.CV]. URL: https://arxiv.org/abs/2310.02239 (cit. on p. 13).

[72] Google DeepMind. *Google introduces Gemini 2.0: A new AI model for the agentic era*. https://blog.google/technology/google-deepmind/google-gemini-ai-update-december-2024/. Accessed: 2025-07-01. 2024 (cit. on p. 13).

[73] Zhaopeng Gu et al. «Anomalygpt: Detecting industrial anomalies using large vision-language models». In: *AAAI*. 2024 (cit. on p. 17).

[74] Wenxi Lv, Qinliang Su, and Wenchao Xu. «One-for-All Few-Shot Anomaly Detection via Instance-Induced Prompt Learning». In: *ICLR*. 2025 (cit. on p. 17).

[75] Mohamed El Banani et al. «Probing the 3D Awareness of Visual Foundation Models». In: *CVPR*. 2024 (cit. on p. 17).

[76] Jiacong Xu et al. «Towards Zero-Shot Anomaly Detection and Reasoning with Multimodal Large Language Models». In: 2025 (cit. on pp. 17, 37).

[77] Jianwei Yang et al. «Set-of-Mark Prompting Unleashes Extraordinary Visual Grounding in GPT-4V». In: *arXiv:2310.11441* (2023) (cit. on p. 17).

[78] Gemini Team et al. «Gemini: a family of highly capable multimodal models». In: *arXiv:2312.11805* (2023) (cit. on p. 17).

[79] Stefan Stojanov, Anh Thai, and James M Rehg. «Using shape to categorize: Low-shot learning with an explicit shape bias». In: *CVPR*. 2021 (cit. on p. 18).

[80] Sebastian Koch et al. «Abc: A big cad model dataset for geometric deep learning». In: *CVPR*. 2019 (cit. on p. 18).

[81] Tsung-Yi Lin et al. «Feature Pyramid Networks for Object Detection». In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017 (cit. on p. 20).

[82] Zak Murez et al. *Atlas: End-to-End 3D Scene Reconstruction from Posed Images*. 2020. arXiv: 2003.10432 [cs.CV]. URL: https://arxiv.org/abs/2003.10432 (cit. on p. 20).

[83] Martin Ester et al. «A density-based algorithm for discovering clusters in large spatial databases with noise». In: *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*. KDD'96. Portland, Oregon: AAAI Press, 1996, pp. 226–231 (cit. on p. 21).

[84] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. *Representation Learning with Contrastive Predictive Coding*. 2019. arXiv: 1807.03748 [cs.LG]. URL: https://arxiv.org/abs/1807.03748 (cit. on p. 27).

[85] Phuc H. Le-Khac, Graham Healy, and Alan F. Smeaton. «Contrastive Representation Learning: A Framework and Review». In: *IEEE Access* 8 (2020), pp. 193907–193934. ISSN: 2169-3536. DOI: 10.1109/access.2020.3031549. URL: http://dx.doi.org/10.1109/ACCESS.2020.3031549 (cit. on p. 27).

[86] Ashish Vaswani et al. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762 (cit. on p. 30).

[87] Nikolaos Giakoumoglou and Tania Stathaki. *SynCo: Synthetic Hard Negatives for Contrastive Visual Representation Learning*. 2024 (cit. on p. 30).

[88] Rui Zhu et al. *Improving Contrastive Learning by Visualizing Feature Transformation*. 2021. arXiv: 2108.02982 [cs.CV]. URL: https://arxiv.org/abs/2108.02982 (cit. on p. 31).

[89] Yixiong Zou et al. «A Closer Look at the CLS Token for Cross-Domain Few-Shot Learning». In: *NeurIPS*. Ed. by A. Globerson et al. Vol. 37. 2024, pp. 85523–85545 (cit. on p. 33).

[90] Danila Rukhovich, Anna Vorontsova, and Anton Konushin. «Imvoxelnet: Image to voxels projection for monocular and multi-view general-purpose 3d object detection». In: *WACV*. 2022 (cit. on p. 36).

[91] Shilong Liu et al. *Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection*. 2024. arXiv: 2303.05499 [cs.CV]. URL: https://arxiv.org/abs/2303.05499 (cit. on p. 41).

[92] Nikhila Ravi et al. *SAM 2: Segment Anything in Images and Videos*. 2024. arXiv: 2408.00714 [cs.CV]. URL: https://arxiv.org/abs/2408.00714 (cit. on p. 42).