Neuro-Symbolic Logical Anomaly Detection with Interpretable Neural Rules

Malihe Dahmardeh[1] and Francesco Setti[1,2]

[1] Department of Engineering for Innovation Medicine, University of Verona, Italy
[2] Qualyco S.r.l., strada le Grazie 15, Verona, Italy
Corresponding Author: malihe.dahmardeh@univr.it

Abstract. Identifying logical anomalies in industrial settings remains a significant challenge for conventional anomaly detection methods, as these defects involve subtle violations of component relationships, quantities, and arrangements rather than structural flaws. In this paper, we propose a Neuro-Symbolic Logical Anomaly Detection (NeSy-LAD) framework that integrates deep learning-based visual analysis with explicit symbolic reasoning to address these complex constraints. At the core of our approach is the Neural Pattern Discovery (NPD) module, a novel rule-extraction engine that autonomously identifies component-specific visual signatures without manual supervision and learns a comprehensive taxonomy of nine distinct rule types. These symbolic rules are integrated into a Logic Tensor Network (LTN), enabling end-to-end differentiable reasoning that grounds logical constraints in visual data. By combining data-driven feature extraction with symbolic reasoning, NeSy-LAD provides human-readable explanations of detected anomalies through explicit rule-violation analysis—a capability no competing method offers—while achieving competitive detection performance on the MVTec LOCO dataset. Crucially, all rule parameters are discovered automatically from normal training images without manual specification, making the framework directly applicable across diverse product categories.

Keywords: Logical Anomaly Detection · Neuro-Symbolic AI · Logic Tensor Networks · Interpretable Rule Extraction · Visual Pattern Discovery

1 Introduction

Industrial anomaly detection is a crucial task in manufacturing quality control, aimed at identifying products that deviate from their expected specifications. Existing methods primarily use feature-based approaches to detect pixel-level deviations from normal patterns. These approaches have proven effective for structural anomalies but often fail to capture the complex relationships and constraints that define logical anomalies. Logical anomalies involve violations of constraints on the arrangement, quantities, and relationships among components, which conventional deep learning approaches often struggle to identify. For instance, as depicted in Figure 1, verifying that a breakfast box contains

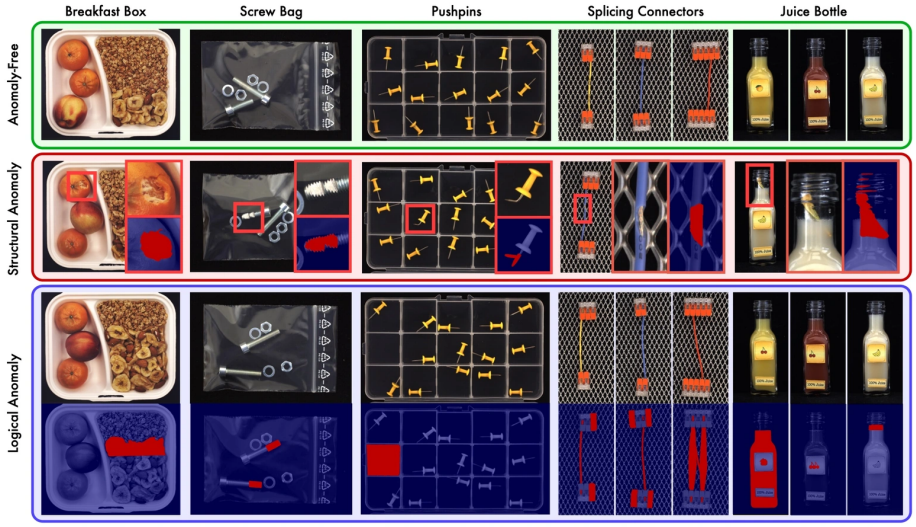

Fig. 1. Examples of logical anomalies presented in the MVTec LOCO dataset.

the correct components, a juice bottle carries the right label, or a pushpin box holds the expected number of pins demands relational reasoning beyond pixel statistics and, critically, requires an explanation that an operator can act on, not just a binary flag.

In safety-critical production, black-box anomaly scores are insufficient: operators must know *which constraint was violated* to intervene effectively. Neuro-symbolic AI (NeSy) [8] addresses this by combining neural perception with explicit symbolic reasoning, enabling models to detect violations and explain them in human-readable terms.

In our preliminary work [6], we introduced the NeSy Logical Anomaly Detection (NeSy-LAD) framework, which integrates deep learning-based component segmentation with symbolic rule extraction and logical reasoning. In this paper, we significantly extend NeSy-LAD through four enhancements: automatic product subtype discovery via adaptive clustering with type-specific rule application; a nine-type rule taxonomy with semantic kernel mapping for human-readable pattern descriptors; a dual-pipeline architecture separating logical from structural detection; and Bayesian hyperparameter optimization replacing manual tuning. Together, these yield substantial performance improvements with richer, type-aware explanations.

The main contributions of our work are as follows:

- We introduce an automatic product type discovery system that identifies subtypes within categories using adaptive clustering and applies specialized type-specific rules for each discovered variant;
- We propose a comprehensive NeSy approach for logical anomaly detection that integrates an Enhanced Neural Pattern Discovery module that au-

tonomously learns component-specific visual signatures and extracts a comprehensive nine-type rule taxonomy—including multimodal patterns, kernel relations, co-occurrence, count, area, histogram, presence, color, and patch histogram rules—with semantic kernel mapping for interpretable pattern descriptors;
– We implement systematic Bayesian hyperparameter optimization using Tree-structured Parzen Estimator to learn optimal rule-type weights and fusion parameters, replacing manual tuning; and
– We incorporate a Logic Tensor Network (LTN) [1] reasoning framework that enables differentiable logical reasoning and provides human-readable explanations of detected anomalies.

2 Related Work

2.1 Logical Anomaly Detection

Traditional methods for industrial anomaly detection have primarily focused on feature-based techniques. Approaches such as PatchCore [19], SimpleNet [15], and AST [20] demonstrate strong performance in detecting structural anomalies but often fall short in identifying logical anomalies, which involve intricate relationships and constraints among components. Early efforts in logical anomaly detection addressed this challenge by segmenting components and comparing their region features to stored prototypes in a memory bank [14]. Subsequent enhancements introduced multiple memory banks to capture both local and global feature representations [11].

More recent studies employ the Segment Anything Model (SAM) [12] for component segmentation, followed by lightweight networks utilizing DINO-derived features [9], statistical anomaly scoring for zero-shot detection [17], or compositional modeling strategies like SALAD [7]. While these approaches reduce reliance on manual annotations, they lack explicit mechanisms for understanding and explaining inter-component relationships.

In parallel, vision-language model (VLM)-based approaches have emerged to address logical anomalies. For example, LogSAD [23] integrates patch-level and compositional feature matching using VLMs for multi-granularity detection. LogicQA [13] utilizes VLMs to generate anomaly-relevant questions, enhancing interpretability, whereas LogicAD [10] incorporates VLM-extracted textual features into logical reasoning modules. LogiCode [24] introduces a large language model-driven pipeline that employs detailed LOCO annotations. However, LogicAD and LogicQA are designed primarily for few-shot scenarios, and LogiCode depends on custom, dataset-specific annotation schemes. Despite the progress made by these VLM-based methods, their reliance on foundation model reasoning rather than explicit symbolic logic frameworks limits their formal reasoning capabilities and the interpretability of their outputs in structured settings.

2.2 Neuro-symbolic Systems

Neuro-symbolic systems combine the strengths of connectionist models (neural networks) and symbolic reasoning (logic-based approaches), offering a unified paradigm for learning and inference [8]. One notable framework in this space is the Logic Tensor Network (LTN) [1], which facilitates differentiable logical reasoning within neural networks. By jointly optimizing both network parameters and logical formulas in an end-to-end manner, LTNs enable models to capture statistical patterns while adhering to explicitly defined logical constraints. This integration has paved the way for NeSy learning cycles, a conceptual model that connects neural and symbolic components in a closed feedback loop comprising three phases: (1) *knowledge extraction*, where neural models generate symbolic representations from data; (2) *user interaction*, where human experts refine the symbolic knowledge; and (3) *knowledge injection*, where the refined symbolic information is reintegrated to guide further neural learning.

Several studies have explored individual components or combinations of these phases. ERIC [22] presents a method to derive symbolic rules from CNN activations, thereby offering interpretable insights into the learned representations. Building on this, Ngan et al. [16] proposed an interactive framework for extracting symbolic rules in radiological imaging, enabling expert feedback to inform model behavior. In the field of visual reasoning, Serrano et al. [21] introduced a probabilistic abduction framework that integrates neural networks with vector-symbolic architectures, learning interpretable rules from visual data and performing inference via probabilistic logic. These neuro-symbolic methods provide a powerful foundation for logical anomaly detection by enabling models to reason over component relationships, enforce consistency with domain-specific rules, and offer interpretable explanations for violations of expected logical structures.

3 Methodology

Our Neuro-Symbolic Logical Anomaly Detection (NeSy-LAD) framework tackles the challenge of identifying logical anomalies through a dual-level system. The core contribution of this work is the Logical Detection Pipeline, which integrates deep learning-based component segmentation with a Neural Pattern Discovery (NPD) module. This module identifies component-specific visual signatures and extracts a structured taxonomy of nine rule types (e.g., multimodal patterns, relational constraints). These rules are encoded into a Logic Tensor Network (LTN) to enable differentiable symbolic reasoning. Complementing this, the Structural Detection Pipeline targets conventional visual anomalies such as scratches and dents. It utilizes multi-scale feature analysis, drawing on texture, gradient, and intensity features derived from encoder representations.

By combining these two pipelines, NeSy-LAD offers comprehensive anomaly detection, capturing both high-level semantic inconsistencies and low-level structural defects (see Fig. 2).

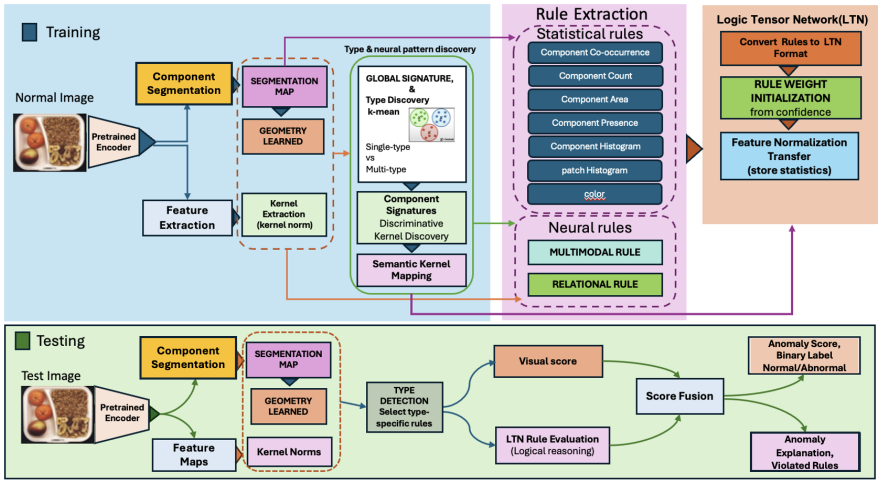

Fig. 2. Overview of the proposed pipeline for logical anomaly detection.

3.1 Feature Extraction and Kernel Analysis

We employ a WideResNet-50 backbone pre-trained on ImageNet for feature extraction. In contrast to multi-layer feature aggregation strategies, our approach leverages only the final convolutional layer, which captures high-level semantic representations well-suited for reasoning about component identity and spatial arrangement. Given an input image $x \in \mathbb{R}^{256\times256\times3}$ normalized using ImageNet statistics, the backbone network produces a final feature map with spatial dimensions $H \times W$ and $C = 2048$ channels. Each channel functions as a learned semantic pattern detector, capturing specific visual features across the dataset. We denote these extracted feature maps as $f \in \mathbb{R}^{H\times W\times C}$, where each channel acts as a kernel that responds to particular visual pattern. These feature maps form the foundation for our component signature discovery process.

3.2 Kernel Activation Extraction

Kernels in the final convolutional layer act as semantic pattern detectors. We quantify the presence of these patterns by computing a scalar activation value κ_c for each kernel $c \in \{1, \ldots, C\}$. To ensure robustness against scale differences and outliers, we first apply Z-normalization to each feature map channel independently, followed by a mean absolute value aggregation:

$$\kappa_c = \frac{1}{H \times W} \sum_{i=1}^{H} \sum_{j=1}^{W} \left| \frac{f(i,j,c) - \mu_c}{\sigma_c + \epsilon} \right| \tag{1}$$

where μ_c and σ_c are the spatial mean and standard deviation of channel c, and $\epsilon = 10^{-6}$ ensures numerical stability. This normalization standardizes the re-

sponse magnitude across different kernels, ensuring that naturally high-activation features do not dominate the logical reasoning process.

To convert continuous activations into logical predicates, we learn component-specific thresholds. We consider a generic kernel K_c as activated if the value of that kernel exceeds the 20th percentile of its activation distribution on normal training samples for component c. This establishes a robust lower bound for feature presence, requiring that a test sample exhibit at least the activation strength found in the top 80% of normal instances to be considered valid.

3.3 Component Segmentation and Geometric Analysis

We implement the component segmentation module using the DeepLabV3+ architecture [5], enhancing it with U-Net-style skip connections [18] to enable precise pixel-level identification of components. The network processes multi-scale feature maps extracted from the WideResNet-50 backbone and feeds them through an Atrous Spatial Pyramid Pooling (ASPP) decoder [4] to produce semantic segmentation maps.

For each segmented component m, we extract a set of geometric features essential for logical reasoning:

- **Normalized Area (A_m):** area of the component as a percentage of the whole image;
- **Compactness (M_m):** computed as the area/perimeter ratio;
- **Aspect Ratio (AR_m):** defined as the ratio between height and width of the component's bounding box;
- **Centroid (x_m, y_m):** coordinates of the component's centroid normalized to image dimensions.

To define normalcy, we compute the mean μ_g^m and standard deviation σ_g^m for each feature g across normal training samples. We consider a feature in the normal range if it falls into the $\mu_g^m \pm z\sigma_g^m$ interval, where z is a free parameter that can be set or learned for each feature.

3.4 Automated Type Discovery and Component Pattern Learning

To automatically determine category structure, we extract a global signature from each training image that captures its compositional characteristics. The signature $g_i = [a_i; c_i; s_i; v_i] \in \mathbb{R}^{19}$ combines four feature groups: kernel activation statistics ($a_i \in \mathbb{R}^6$) including diversity and dominance metrics, normalized area proportions for each component class ($c_i \in \mathbb{R}^7$), spatial layout characteristics ($s_i \in \mathbb{R}^3$) such as component count and distribution entropy, and visual complexity indicators ($v_i \in \mathbb{R}^3$) measuring activation patterns. We perform adaptive clustering on normalized signatures $\tilde{g}_i$ using K-means with silhouette analysis to automatically discover product subtypes within each category. Product types represent visually distinct variants of the same category. The silhouette score $S(k)$ measures cluster quality (ranging from -1 to 1, where higher values indicate

better-defined clusters) by comparing within-cluster cohesion to between-cluster separation. We select k^* that maximizes $S(k)$ under the constraints $S(k) > \tau_{\min}$ and minimum cluster size of 5. If $k^* = 1$ or $S(k^*) \leq \tau_{\min}$, the category is single-type and all samples share unified rules; otherwise, the category is multi-type with k^* subtypes, each receiving specialized rules tailored to its specific configuration. This automatic type discovery enables the framework to handle products with multiple valid configurations without manual specification, improving detection accuracy by applying configuration-specific constraints rather than attempting to fit all variants with a single rule set.

For each component m, we compute area-weighted kernel activations and rank them by importance using a score based on the mean-to-variance ratio. We select the top kernels as discriminative features. Additionally, we extract texture-based statistics, including activation strength, diversity, and high-frequency ratios. These features are aggregated into a structured component signature Σ_m, which captures both semantic and statistical patterns, supporting efficient evaluation and rule-based reasoning.

3.5 Semantic Kernel Mapping for Interpretability

To enable interpretable rule expressions, our framework automatically discovers semantic meanings for discriminative kernels by analyzing their activation patterns across training samples. For each component m and its discriminative kernels $k \in \mathcal{D}_m$, where $\mathcal{D}_m$ is the top-ranked kernels selected by importance scores, we compute correlation coefficients between kernel activations and geometric properties $\mathbf{A}_m^i$ and $\mathbf{AR}_m^i$, respectively, the normalized area and aspect ratio as described above. Strong correlations ($|\rho| > 0.3$) reveal semantic relationships, enabling automatic assignment of descriptive names. For instance, kernels with high positive correlation with the normalized area of a specific component act as detectors of instances where the component is large, while a large negative correlation with the aspect ratio identifies compact components.

3.6 Enhanced Rule Extraction with Nine-Type Taxonomy

Importantly, while the taxonomy defines nine rule types, all rule parameters (i.e., thresholds, component identities, statistical ranges, and reliability weights) are automatically discovered from normal training images without any manual specification per product. The same taxonomy applies unchanged across all five MVTec LOCO categories; the number and content of instantiated rules vary automatically to reflect each product's complexity. Leveraging learned component signatures, the Enhanced ERIC framework autonomously constructs a taxonomy of 9 distinct rule types to comprehensively capture both semantic and structural anomalies.

Type 1: Multimodal Component Pattern Rules combine visual features with geometric constraints to define a component's normal appearance.

For each component c, we generate a rule requiring satisfaction of both discriminative kernel activations and geometric properties:

$$\mathcal{R}_1(x) : \left(\bigwedge_{k \in D_c} \kappa_k(x) > \theta_k \right) \wedge \left(\bigwedge_g \text{feat}_g(x) \in [\mu_g \pm 2\sigma_g] \right) \rightarrow \text{Normal}_c(x) \quad (2)$$

Rule evaluation employs continuous fuzzy logic for differentiability. We use a sigmoid function $\sigma(10 \cdot (\kappa - \theta))$ to smooth the decision boundary around the threshold for kernel satisfaction and a Gaussian-like decay function for geometric features, ensuring smooth gradients during training.

Type 2: Kernel Relational Rules capture interactions between component pairs (m_a, m_b) . We extract relational features including visual similarity $(\kappa_{m_a} \cdot \kappa_{m_b})$, contrast $(|\kappa_{m_a} - \kappa_{m_b}|)$, and spatial metrics (centroid distance, size ratio). Let $\mathcal{F}$ denote the set of extracted relational features between a component pair. The rule structure enforces that all features fall within learned normal ranges:

$$R_2(m_a, m_b) : \bigwedge_{j \in \mathcal{F}} (f_j \in \mu_j \pm 3\sigma_j) \rightarrow \text{ValidRelation}(x) \quad (3)$$

where f_j represents the j-th relational feature, and μ_j, σ_j are its mean and standard deviation computed from normal training samples. We employ an automated discovery process using Gaussian Mixture Models (GMMs) with Bayesian Information Criterion (BIC) selection to determine the number of distinct relational patterns that exist, enabling the model to capture multi-modal relational patterns between component pairs.

Type 3: Component Co-occurrence Rules capture statistical patterns of component presence across normal training images. The extraction process analyzes segmentation maps to identify present components and maintains co-occurrence counters for each component pair. The co-occurrence probability is $P(c_b|c_a) = \frac{\text{co-occurrence\_count}(c_a, c_b)}{\text{presence\_count}(c_a)}$. A rule is generated when the conditional probability exceeds 80%, ensuring only strong associations are captured.

Type 4: Component Count Rules enforce quantity constraints for each component type by analyzing instance count distributions. We identify separate instances of each component from binary masks, and we compute statistics as mean (μ_N^c) and standard deviation (σ_N^c) of the number of instances for component c. For instance, the breakfast box class has exactly 2 oranges in each training image; thus, we learn that $\mu_N^{orange} = 2$ and $\sigma_N^{orange} = 0$. The formal structure ensures observed counts fall within learned ranges:

$$\mathcal{R}_4(x) : \text{count}_c(x) \in \mu_N^c \pm 2\sigma_N^c \quad (4)$$

Type 5: Component Area Rules detect geometric anomalies through statistical constraints on component sizes. We compute statistical models from the normalized area of each component in the training images in the form of

mean $\mu_{\mathbf{A}_c}$ and standard deviation $\sigma_{\mathbf{A}_c}$. Nominal samples will satisfy, for all components, the rule:

$$\mathcal{R}_5(x) : \mathbf{A}_c(x) \in \mu_{\mathbf{A}_c} \pm \sigma_{\mathbf{A}_c} \tag{5}$$

Type 6: Component Histogram Rules establish statistical baselines for component area distributions. For each sample i, we compute the histogram of component areas $\mathbf{h}^{(i)} = [\mathbf{A}_{c_1}, \mathbf{A}_{c_2}, \ldots, \mathbf{A}_C]$. The reference distribution is the mean across training samples: $\mu_{\mathbf{h}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{h}^{(i)}$ while the variability is captured by standard deviation: $\sigma_{\mathbf{h}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\mathbf{h}^{(i)} - \mu_{\mathbf{h}}\right)^2}$.

Type 7: Component Presence Rules define binary constraints for component existence. Presence frequency f_c is computed as the ratio between the number of training images containing that component and the total number of training images. Components with high frequencies generate mandatory presence rules. Low frequencies generate forbidden presence rules. We define two thresholds τ_h and τ_l such that:

$$\mathcal{R}_7 : \begin{cases} f_c > \tau_h & \rightarrow \text{required component} \\ f_c < \tau_l & \rightarrow \text{forbidden component} \end{cases} \tag{6}$$

Type 8: Component Color Rules ensure spectral consistency by detecting discoloration or incorrect fluid types. Rules are generated only for components with stable chromatic characteristics. Color features are extracted in HSV space with circular statistics for Hue:

$$\mu_H^c = \arctan 2\left(\frac{1}{|M^c|}\sum_{p\in M^c}\sin(h_p), \frac{1}{|M^c|}\sum_{p\in M^c}\cos(h_p)\right) \tag{7}$$

where h_p is Hue in radians and M^c denotes pixels belonging to component c. For each valid component, we extract statistical color parameters μ_H^c and σ_H^c These parameters define the expected color range used during testing. The formal rule structure is:

$$\mathcal{R}_8(x) : (\Delta H(c,x) \leq \tau_{\text{hue}}^c \wedge S(c,x) > S_{\min}) \rightarrow \text{correct\_color}_c(x) \tag{8}$$

Type 9: Patch Histogram Rules detect spatial structural anomalies through multi-scale local pattern analysis. A sliding window extracts patches and computes color histogram vectors $\mathbf{h}_k \in \mathbb{R}^C$ for each patch. Normal spatial distribution is modeled using multivariate Gaussian distributions. An Anomaly Z-Threshold (Z_T) is learned from held-out validation using 95th percentile with trimmed statistics: $Z_T = \frac{D_{95} - \mu_{\text{val}}}{\sigma_{\text{val}}}$ The learned parameters define the normal spatial pattern distribution used during testing. The formal rule structure is:

$$\mathcal{R}_9(x) : (Z_{\text{score}}(x,s) \leq Z_T^s) \rightarrow \text{Normal}(x) \tag{9}$$

3.7 Logic Tensor Network Integration

The Enhanced Logic Tensor Network (LTN) integrates extracted symbolic rules into a differentiable framework, enabling continuous logical reasoning over learned component features. Each rule is formalized as a fuzzy predicate, combining kernel activations, geometric properties, statistical attributes, and spatial patterns into a unified logic-based formulation.

Rule satisfaction is computed using differentiable fuzzy logic operators:

$$\phi_{\text{AND}}(a, b) = a \cdot b, \quad \phi_{\text{THR}}(x, \theta) = \sigma((x - \theta) \cdot 10), \quad \phi_{\text{IMP}}(a, c) = 1 - a + a \cdot c$$

These operators enable the smooth evaluation of rule conditions, with satisfaction values in the range $[0, 1]$ indicating the degree of compliance.

Each rule j is assigned an initial reliability weight $w_j \in [0, 1]$ that determines its influence during inference. These weights are initialized from ERIC confidence scores, $w_j = f_t(\mathcal{D}_{\text{train}})$, where f_t computes statistical reliability for rule type t from training data $\mathcal{D}_{\text{train}}$. For statistical rules (count, area, histogram), confidence reflects pattern consistency measured by coefficient of variation; for multimodal rules, it combines geometric stability with discriminative power. Higher weights indicate more reliable rules that consistently hold across normal samples, while lower weights indicate rules with greater variability. The LTN also transfers normalization statistics from training (kernel scalers, histogram mean/std, geometric ranges) to ensure consistent feature scaling during inference. For multi-type categories, type-specific rule sets $\mathcal{R}_t^{(k)}$ are stored for each detected subtype k.

3.8 Structural Anomaly Detection Pipeline

While the logical pipeline reasons about component relationships and constraints, the structural pipeline detects physical defects such as scratches, dents, and surface irregularities through visual feature analysis. This separation enables independent optimization of both detection paradigms. The structural detector extracts three complementary feature types from the encoder's final feature map $\mathbf{f} \in \mathbb{R}^{H \times W \times C}$:

Texture Features: Capture local pattern variations through learned convolutional filters:

$$\mathbf{f}_{\text{tex}} = \text{Conv}_{\text{tex}}(\mathbf{f}), \quad \mathbf{v}_{\text{tex}} = [\mu_{\text{tex}}; \sigma_{\text{tex}}] \tag{10}$$

where $\mu_{\text{tex}} = \text{AvgPool}(\mathbf{f}_{\text{tex}})$ and $\sigma_{\text{tex}} = \sqrt{\text{AvgPool}(\mathbf{f}_{\text{tex}}^2) - \mu_{\text{tex}}^2 + \epsilon}$ provide statistical texture characterization.

Gradient Features: Detect edge irregularities using Sobel-based gradient operators:

$$\nabla_x = \mathbf{S}_x * \mathbf{f}, \quad \nabla_y = \mathbf{S}_y * \mathbf{f}, \quad \mathbf{m} = \sqrt{\nabla_x^2 + \nabla_y^2 + \epsilon} \tag{11}$$

where $\mathbf{S}_x$ and $\mathbf{S}_y$ are Sobel kernels. The gradient magnitude $\mathbf{m}$ is processed through learned filters:

$$\mathbf{f}_{\text{grad}} = \text{Conv}_{\text{grad}}(\mathbf{m}), \quad \mathbf{v}_{\text{grad}} = [\mu_{\text{grad}}; \max(\mathbf{f}_{\text{grad}})] \tag{12}$$

Intensity Features: Statistical distribution analysis through higher-order moments:

$$\mathbf{v}_{\text{int}} = [\mu, \sigma^2, \gamma_3, \gamma_4] \tag{13}$$

where μ is mean, σ^2 variance, γ_3 skewness, and γ_4 kurtosis, computed via adaptive pooling.

The complete visual feature vector combines all three modalities:

$$\mathbf{v}_{\text{visual}} = [\mathbf{v}_{\text{tex}}; \mathbf{v}_{\text{grad}}; \mathbf{v}_{\text{int}}] \in \mathbb{R}^d \tag{14}$$

During training on N normal samples $\{\mathbf{x}_i\}_{i=1}^N$, we construct a statistical model of the normal feature distribution:

$$\boldsymbol{\mu}_{\text{normal}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{v}_{\text{visual}}(\mathbf{x}_i), \quad \boldsymbol{\sigma}_{\text{normal}} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\mathbf{v}_{\text{visual}}(\mathbf{x}_i) - \boldsymbol{\mu}_{\text{normal}})^2 + \epsilon} \tag{15}$$

For test sample $\mathbf{x}_{\text{test}}$, structural anomaly is quantified via normalized Mahalanobis distance:

$$\mathbf{d}_{\text{norm}} = \frac{\mathbf{v}_{\text{visual}}(\mathbf{x}_{\text{test}}) - \boldsymbol{\mu}_{\text{normal}}}{\boldsymbol{\sigma}_{\text{normal}}}, \quad S_{\text{structural}} = \min\left(\frac{||\mathbf{d}_{\text{norm}}||^2}{10}, 1.0\right) \tag{16}$$

where the division by 10 and clamping to $[0, 1]$ ensures bounded anomaly scores compatible with the logical pipeline.

3.9 Inference in NeSy-LAD

For each test image $\mathbf{x}_{\text{test}}$, kernel activations are extracted and normalized according to Section 3.2. In parallel, a DeepLabV3+ generates a pixel-level component segmentation map and geometric descriptors are extracted from each component as described in Section 3.3. A normalized component histogram $h \in \mathbb{R}^K$ is computed from the segmentation map, where h_k represents the area proportion of component class k.

Enhanced LTN Rule Evaluation with Type-Aware Rule Selection, For multi-type categories, the test image's global signature g_{test} is extracted, normalized, and classified via nearest cluster center t^*:

$$t^* = \arg\min_j \|\tilde{g}_{\text{test}} - c_j\|_2 \tag{17}$$

where c_j are cluster centers from training. The type-specific rule set $\mathcal{R}_t^{(t^*)}$ is selected; for single-type categories, the unified rule set is used.

Rule Satisfaction Computation, For each test image, individual rule satisfactions $S_j \in [0, 1]$ are computed using the fuzzy logic operators and rule-specific formulas defined in Section 3.6. Each rule type (multimodal patterns, relational constraints, count, area, histogram, presence, co-occurrence, patch histogram) is evaluated against the extracted features and geometric descriptors.

Hierarchical Weighted Aggregation, The LTN aggregates rule satisfactions through a two-level hierarchy. First, within each rule type t, individual satisfactions are combined using their initial reliability weights w_j from Section 3.7:

$$\bar{S}_t = \frac{\sum_{j \in \mathcal{R}_t} S_j \cdot w_j}{\sum_{j \in \mathcal{R}_t} w_j} \tag{18}$$

where $\mathcal{R}_t$ contains all rules of type t. Second, these type-averaged satisfactions are combined using optimized type-importance weights w_t^*:

$$S_{\text{LTN}} = \frac{\sum_{t \in \mathcal{T}} w_t^* \cdot \bar{S}_t}{\sum_{t \in \mathcal{T}} w_t^*} \tag{19}$$

where w_t^* are learned once via Bayesian optimization (Tree-structured Parzen Estimator with 500 trials on 20% validation data, maximizing image-level AUROC over $\alpha \in [0.0, 0.8]$ and $w_t \in [0.1, 1.0]$). This hierarchical scheme enables fine-grained per-rule control via w_j and coarse-grained category control via w_t^*. The LTN satisfaction is then converted to an anomaly indicator:

$$S_{\text{LTN\_anomaly}} = 1.0 - S_{\text{LTN}} \tag{20}$$

Visual-Logical Score Fusion. The structural anomaly score $S_{\text{structural}}$ (Section 3.8) and logical anomaly indicator $S_{\text{LTN\_anomaly}}$ are combined using the optimized fusion weight α^* (learned via Bayesian optimization):

$$S_{\text{final}} = \alpha^* \cdot S_{\text{structural}} + (1 - \alpha^*) \cdot S_{\text{LTN\_anomaly}} \tag{21}$$

Binary Classification. Continuous anomaly scores $S_{\text{final}} \in [0, 1]$ are converted to binary predictions using threshold τ^* optimized via F1-score maximization on validation data:

$$\tau^* = \arg\max_{\tau} F_1(\tau) = \arg\max_{\tau} \frac{2 \cdot \text{Precision}(\tau) \cdot \text{Recall}(\tau)}{\text{Precision}(\tau) + \text{Recall}(\tau)} \tag{22}$$

$$\text{is\_anomalous} = (S_{\text{final}} > \tau^*) \tag{23}$$

Interpretable Explanations. For each detected anomaly, the system identifies violated rules (satisfaction < 0.8) and generates human-readable explanations containing: (1) rule type and category (e.g., multimodal pattern, histogram constraint), (2) affected components, and (3) observed versus expected values. This symbolic reasoning provides transparency into anomaly decisions, addressing the interpretability limitations of black-box approaches.

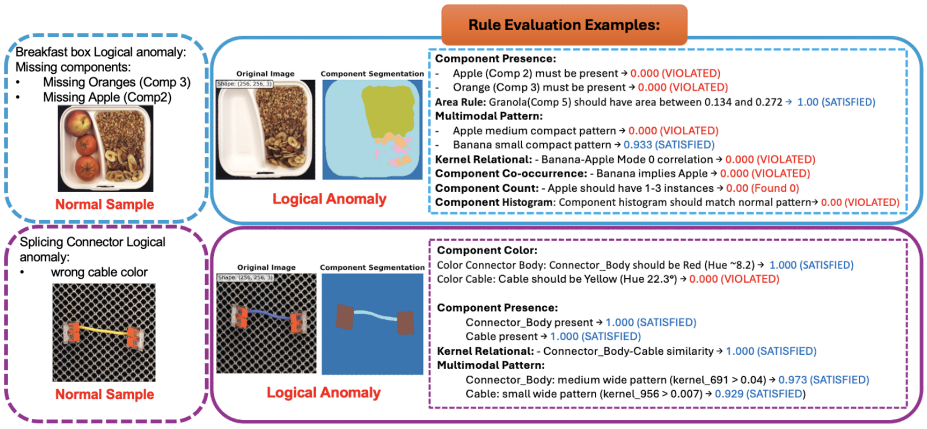

Fig. 3. Rule violation traces for two detected anomalies. Each violated rule reports the observed value and expected range. See Section 4 for discussion.

Table 1. Comparison of MVTec LOCO performance with state-of-the-art methods, measured by image AUROC (%). The top row indicates the type of component segmentation used.

Category	SimpleNet [15]	PatchCore [19]	AST [20]	EfficientAD [2]	PSAD* [11]	ComAD [14]	CSAD [9]	LogSAD [23]	SALAD [7]	**Ours**
Logical Anomalies (LA)										
Breakfast Box	77.1	74.8	80.0	85.5	100.0	91.1	94.4	-	92.9	98.7
Juice Bottle	87.8	93.9	91.6	98.4	99.1	95.0	94.9	-	99.7	91.6
Pushpins	69.0	63.6	65.1	97.7	100.0	95.7	99.5	-	100.0	92.3
Screw Bag	51.6	57.8	80.1	56.7	99.3	71.9	99.9	-	93.9	100.0
Splicing Connectors	72.0	79.2	81.8	95.5	91.9	93.3	94.8	-	96.0	89.6
Average (LA)	71.5	73.9	79.7	86.8	98.1	89.4	96.7	89.3	96.5	94.4
Structural Anomalies (SA)										
Breakfast Box	80.9	80.1	79.9	88.4	84.9	81.6	91.1	-	85.7	75.3
Juice Bottle	90.4	98.5	95.5	99.7	98.2	98.2	95.6	-	99.6	80.3
Pushpins	81.6	87.9	77.8	96.1	89.8	91.1	97.8	-	98.8	78.7
Screw Bag	83.3	92.0	95.9	90.7	95.7	88.5	93.2	-	96.0	91.2
Splicing Connectors	82.6	88.0	89.4	98.5	89.3	94.9	92.2	-	98.6	72.3
Average (SA)	83.7	89.3	87.7	94.7	91.6	90.9	94.0	93.1	95.7	79.5
Total Average	77.6	81.6	83.7	90.7	94.8	90.1	95.3	91.2	96.1	86.9

4 Experiments

We evaluate our Enhanced NeSy-LAD framework on the MVTec LOCO dataset [3], which comprises five product categories: breakfast box, juice bottle, pushpins, screw bag, and splicing connectors. Following established protocols [3], we use Area Under the ROC Curve (AUROC) as our primary evaluation metric. All experiments are conducted on normal training images only, with evaluation on both normal and anomalous test images. The framework is implemented using PyTorch with WideResNet-50 for feature extraction and DeepLabV3+ for component segmentation. Images are resized to 256×256 pixels.

NeSy-LAD achieves 94.4% logical AUROC, only 2.1 points below SALAD (96.5%), while providing per-rule satisfaction traces that SALAD, CSAD, and all other methods in Table 1 cannot produce. our Fig. 3 demonstrates the qualitative output directly. The structural AUROC (79.5%) reflects a deliberate trade-off: the structural pipeline complements logical reasoning rather than competing with appearance-specialized detectors. Logical Anomalies Strong performance on breakfast box (98.7%), screw bag (100.0%), and pushpins (92.3%) demonstrates effective rule extraction and LTN reasoning. Challenges in juice bottle (91.6%) stem from segmentation quality issues with small fruit icons. Splicing connectors (89.6%) faces fine-grained type distinction difficulties due to visually similar configurations. reflects our design emphasis on interpretable logical reasoning over black-box pattern matching.

5 Conclusions

NeSy-LAD addresses a strictly harder problem than anomaly detection alone: every decision must be accompanied by a formal, rule-grounded explanation auditable by a human operator. Achieving 94.4% logical AUROC—within 2.1 points of the state of the art—with full symbolic transparency is the core result. Our approach provides: (1) explicit symbolic reasoning with Logic Tensor Networks enabling formal verification of anomaly decisions, (2) human-readable rule-based explanations through semantic kernel mapping, and (3) differentiable end-to-end training that jointly optimizes neural pattern recognition and logical reasoning. Unlike discriminative methods that learn implicit decision boundaries, our neuro-symbolic approach offers transparency and interpretability critical for industrial deployment where explainability is mandatory.

References

1. Badreddine, S., d'Avila Garcez, A., Serafini, L., Spranger, M.: Logic tensor networks. Artificial Intelligence **303**, 103649 (2022) 3, 4
2. Batzner, K., Heckler, L., König, R.: Efficientad: Accurate visual anomaly detection at millisecond-level latencies. In: IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) (2024) 13
3. Bergmann, P., Batzner, K., Fauser, M., Sattlegger, D., Steger, C.: Beyond dents and scratches: Logical constraints in unsupervised anomaly detection and localization. International Journal of Computer Vision **130**(4), 947–969 (2022) 13
4. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Rethinking atrous convolution for semantic image segmentation. In: arXiv preprint arXiv:1706.05587 (2017) 6
5. Chen, L.C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder-decoder with atrous separable convolution for semantic image segmentation. In: European Conference on Computer Vision (ECCV) (2018) 6
6. Dahmardeh, M., Saadatpour, M., Manigrasso, F., Morra, L., Setti, F.: Nesylad: A neuro-symbolic approach for unsupervised logical anomaly detection. In: International Conference on Image Analysis and Processing (ICIAP) (2025) 2

7. Fučka, M., Zavrtanik, V., Skočaj, D.: Salad – semantics-aware logical anomaly detection. In: IEEE/CVF International Conference on Computer Vision (ICCV) (2025) 3, 13
8. Garcez, A.d., Lamb, L.C.: Neurosymbolic AI: The 3rd wave. Artificial Intelligence Review **56**(11), 12387–12406 (2023) 2, 4
9. Hsieh, Y.H., Lai, S.H.: CSAD: Unsupervised component segmentation for logical anomaly detection. arXiv preprint arXiv:2408.15628 (2024) 3, 13
10. Jin, E., Feng, Q., Mou, Y., Lakemeyer, G., Decker, S., Simons, O., Stegmaier, J.: Logicad: Explainable anomaly detection via VLM-based text feature extraction. In: AAAI Conference on Artificial Intelligence. vol. 39, pp. 4129–4137 (2025) 3
11. Kim, S., An, S., Chikontwe, P., Kang, M., Adeli, E., Pohl, K.M., Park, S.H.: Few shot part segmentation reveals compositional logic for industrial anomaly detection. In: AAAI Conference on Artificial Intelligence (2024) 3, 13
12. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: IEEE/CVF International Conference on Computer Vision (ICCV)********* (2023) 3
13. Kwon, Y., Moon, D., Oh, Y., Yoon, H.: LogicQA: Logical anomaly detection with vision language model generated questions. arXiv preprint arXiv:2503.20252 (2025) 3
14. Liu, T., Li, B., Du, X., Jiang, B., Jin, X., Jin, L., Zhao, Z.: Component-aware anomaly detection framework for adjustable and logical industrial visual inspection. Advanced Engineering Informatics **58**, 102161 (2023) 3, 13
15. Liu, Z., Zhou, Y., Xu, Y., Wang, Z.: SimpleNet: A simple network for image anomaly detection and localization. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2023) 3, 13
16. Ngan, K.H., Mansouri-Benssassi, E., Phelan, J., Townsend, J., Garcez, A.d.: From explanation to intervention: Interactive knowledge extraction from convolutional neural networks used in radiology. PLoS ONE **19**(4), e0293967 (2024) 4
17. Peng, Y., Lin, X., Ma, N., Du, J., Liu, C., Liu, C., Chen, Q.: SAM-LAD: Segment anything model meets zero-shot logic anomaly detection. Knowledge-Based Systems **314**, 113176 (2025) 3
18. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: International Conferenece on Medical Image Computing and Computer Assisted Intervention (MICCAI) (2015) 6
19. Roth, K., Pemula, L., Zepeda, J., Schölkopf, B., Brox, T., Gehler, P.: Towards total recall in industrial anomaly detection. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2022) 3, 13
20. Rudolph, M., Wehrbein, T., Rosenhahn, B., Wandt, B.: Asymmetric student-teacher networks for industrial anomaly detection. In: IEEE/CVF Winter Conference on Applications of Computer Vision (WACV) (2023) 3, 13
21. Serrano, P., Fernandez, D., Ren, B., Sierra, A., Malone, B., Kochenderfer, M.J.: Probabilistic abduction for visual abstract reasoning via learning rules in vector-symbolic architectures. arXiv preprint arXiv:2401.16024 (2024) 4
22. Townsend, J., Kasioumis, T., Inakoshi, H.: ERIC: Extracting relations inferred from convolutions. In: Asian Conference on Computer Vision (ACCV) (2021) 4
23. Zhang, J., Wang, G., Jin, Y., Huang, D.: Towards training-free anomaly detection with vision and language foundation models. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) (2025) 3, 13
24. Zhang, Y., Cao, Y., Xu, X., Shen, W.: Logicode: an LLM-driven framework for logical anomaly detection. IEEE Transactions on Automation Science and Engineering **22**, 7712–7723 (2024) 3