

# Graph-Based Representation of Contour Images: A Step Toward Self-Describing Explainable AI

Mykyta Lapin

*Department of System Analysis and Information  
and Analytical Technologies  
of National Technical University  
“Kharkiv Polytechnic Institute”  
Kharkiv, Ukraine  
Mykyta.Lapin@cit.khpi.edu.ua*

Yurii Parzhyn

*Postdoctoral Fellow  
School of Computer and Cyber Sciences  
Augusta University  
Augusta, USA  
yparzhyn@augusta.edu*

Kostiantyn Bokhan

*Department of System Analysis and Information  
and Analytical Technologies  
of National Technical University  
“Kharkiv Polytechnic Institute”  
Kharkiv, Ukraine  
kostiantyn.bokhan@khpi.edu.ua*

Kyrylo Perevoznyk

*Department of System Analysis and Information  
and Analytical Technologies  
of National Technical University  
“Kharkiv Polytechnic Institute”  
Kharkiv, Ukraine  
kyrylo.perevoznyk@cs.khpi.edu.ua*

**Abstract**—Opaque perception pipelines hinder trust and diagnosis in real-world AI systems. We present a graph-based representation for contour images that makes structure the carrier of explanations. The approach encodes visual patterns as attributed graphs where nodes represent critical points (endpoints, corners, junctions) and edges represent connecting line segments, with semantic tags and geometric attributes attached directly to graph elements. Crucially, we demonstrate how multiple graph instances of the same class can be reduced to form stable *concept attractors*—canonical graph structures that capture shared topological and parametric patterns while filtering instance-specific variations. This concept formation process operates through iterative structural and parametric reduction without backpropagation, enabling few-shot learning from minimal training data. We validate the approach on MNIST handwritten digits, where concepts formed from 5–6 training samples achieve meaningful classification through graph matching. By encoding semantics explicitly in graph structure rather than in opaque weight matrices, the approach advances explainable AI while maintaining competitive few-shot learning performance.

**Index Terms**—explainable AI (XAI), artificial neural networks, hypergraph models, few-shot learning, structural reduction, graph matching

## I. INTRODUCTION

Modern perception systems achieve impressive accuracy yet often operate as opaque pipelines, making it difficult for practitioners to understand *why* a particular decision was reached or *where* evidence resides in the input. Post-hoc explanation techniques can be helpful, but they typically project reasoning onto models that were not designed to be interpretable, yielding artifacts that are hard to validate or reuse across tasks. Modern post-hoc explanation methods such as LIME and SHAP can be adversarially manipulated or yield inconsistent attributions [1]–[3]. Knowledge-graph-

based approaches demonstrate that explicit graph structures improve interpretability over opaque weight matrices [4]. This gap undermines trust, slows diagnosis when things fail, and complicates compliance in domains that require traceable decision paths. To tackle these challenges, we propose elevating **human-readable internal structure** to a core design principle from the outset.

Our main idea is to make the **structure the carrier of explanations**. To move beyond the reliance on primarily latent vectors—whose semantics remain implicit and largely uninterpretable—we propose encoding visual information using explicit, human-intuitive structures. Specifically, we use **graphs** whose semantics are transparent and readily comprehensible. For 2D images, **contours** offer a compact, model-agnostic shape description: endpoints, corners, and junctions delineate where curves terminate or intersect, while **line segments** define the connections between them. We represent this structure as a graph where both these critical points and the lines are nodes, with edges capturing their spatial adjacency. This design makes line segments first-class queryable entities rather than mere connectivity markers. **Attributes and tags** are stored directly in the graph elements to preserve meaning inside the representation: node types (labels) mark structural roles (e.g. Point, Line), other features capture orientation and position, and connectivity patterns reveal topological properties. Explanations become navigable objects: cycles justify closed contours, node degree reveals branching structure, and attribute ranges encode acceptable variation.

The goal of this paper is therefore modest and focused: we **show how contours can be encoded as attributed graphs** that remain readable to both humans and machines, and we demonstrate how multiple graphs of the same class

can be reduced to form **concept attractors**—stable canonical structures suitable for classification. We present the method at a **high level**, emphasizing what is represented rather than prescribing specific extraction algorithms. Concretely, we: (i) define a minimal vocabulary of node/edge types and attributes; (ii) describe normalization choices that make the representation invariant to scale and translation; and (iii) outline how **structural reduction rules** iteratively merge training samples into generalized concepts that preserve shared topology while abstracting parametric variation. We validate the approach on MNIST handwritten digits, demonstrating that concepts formed from 5–6 training samples per class enable meaningful few-shot classification through graph matching, with aggregate structural metrics confirming that the representation remains compact and interpretable.

What this paper does **not** attempt is equally important. We do not propose or benchmark a new contour-extraction algorithm; we assume edges or skeletons are available from any standard method. We do not optimize classification performance or compare against state-of-the-art benchmarks; our goal is to demonstrate the viability of graph-based concept formation and the interpretability it affords. We also avoid deep theory beyond what is necessary to state the representation and its reduction operators. Those aspects—algorithmic optimizations, large-scale benchmarking, and formal convergence properties—are intentionally left for **follow-up work**, where each topic can be treated with appropriate depth.

## II. BACKGROUND AND RELATED WORK

Current approaches to explainable AI (XAI) predominantly rely on post-hoc techniques such as LIME and SHAP that attempt to reverse-engineer already-trained model behavior [1], [5]. However, recent work has exposed fundamental vulnerabilities: Slack *et al.* [2] demonstrated that both methods can be adversarially manipulated to hide discriminatory behavior, while others have documented high sensitivity to hyperparameters and feature collinearity [3]. A fundamental challenge with post-hoc explanations is that, because they are generated *after* the model has been trained, it is inherently difficult to corroborate them against the model’s actual reasoning process.

Graph-based representations offer an alternative paradigm where structure itself carries semantic information. Vision GNNs treat images as graphs with content-based connectivity, enabling interpretable visual reasoning [6], while structured shape representations (contours and skeletons) have long been studied for capturing boundaries and topological structure [7], [8]. Evidence from graph-based contour representations [9] and from alternative vectorization techniques [10] further supports encoding visual patterns using explicit structural primitives. Graph Edit Distance (GED) provides a principled measure for comparing such representations through edit operations, and recent hybrid approaches combine GED with learned embeddings to achieve both efficiency and interpretability [11], [12]. Meanwhile, few-shot learning addresses concept formation from minimal data, typically through meta-learning frameworks that learn to adapt quickly across tasks

[13], [14]. Neuro-symbolic AI extends this by integrating explicit knowledge graphs and symbolic reasoning with learned representations, providing introspectable decision paths [4], [15].

Despite these advances, no existing approach combines structural explainability with few-shot capability through graph-based concept formation. Post-hoc methods remain disconnected from the learning process, while graph neural networks and few-shot learners rely on opaque learned weights rather than explicit structural attractors. Taken together, energy-based neural network formulations [16] and principles of information representation [17] provide the theoretical foundation for concept formation achieved through structural reduction. The present work addresses this by encoding visual patterns directly as attributed contour graphs and forming concepts through iterative structural reduction to stable attractors, making structure—not weights—the primary carrier of both representation and explanation.

## III. GRAPH REPRESENTATION

### A. Node Types and Bipartite Structure

The system represents image contours as bipartite graphs alternating between *Point* nodes and *Line* nodes. It is important that line segments are represented as first-class nodes rather than edges, allowing them to have attributes such as length, direction, and orientation. Point nodes and Line nodes connect via bidirectional `CONNECTED_TO` relationships, creating a traversal pattern of the form *Point* → *Line* → *Point* → *Line*.

Point nodes are classified into four topological types: **EndPoint** (terminal nodes of open contours), **CornerPoint** (nodes marking sharp directional changes, storing an angle attribute), **IntersectionPoint** (nodes where multiple segments meet), and **StartPoint** (anchor nodes for graph traversal). Each node in Neo4j carries multiple labels (e.g., ["Point", "CornerPoint"]), enabling both type-specific queries and generic structural operations. This bipartite design separates topological structure (points) from geometric properties (line segments), facilitating explicit feature extraction and structural comparison.

### B. Attributes and Parameters

Each node type carries attributes that encode geometric, directional, and topological properties. **Point nodes** store  $(x, y)$  coordinates and their normalized counterparts as well as an angle value between the lines it connects. **Line nodes** store endpoint pairs  $(x_1, y_1, x_2, y_2)$  and a length attribute (magnitude), enabling direct access to segment geometry without recomputing from connected points.

Directional attributes capture spatial orientation. Each Line node is assigned a *quadrant* (I, II, III, IV) based on the displacement vector from its start point, computed when the coordinate system is translated to that point. Lines also carry *horizontal direction* (LEFT, RIGHT, NONE) and *vertical direction* (TOP, BOTTOM, NONE) labels that encode relative

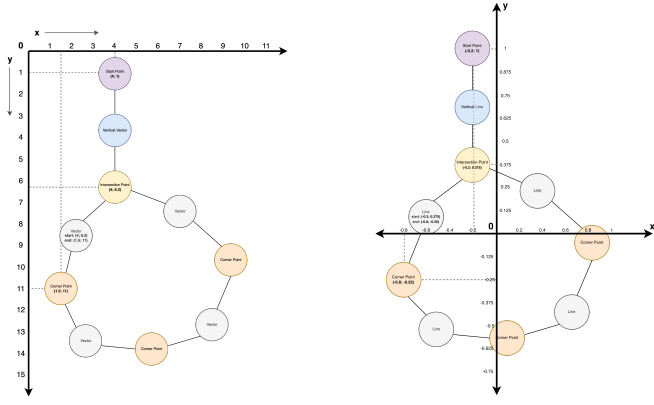


Fig. 1. Coordinate system transformation from absolute pixel coordinates (left) with origin at top-left to normalized centered coordinates (right) ranging  $[-1, 1]$ . This transformation enables scale and translation invariance for concept formation.

positioning. These attributes, computed by visitor classes during graph traversal, enable comparisons of spatial development patterns across different contours.

### C. Normalization

To achieve scale and translation invariance, all coordinates and distances are normalized. Point coordinates are transformed to a centered system ranging  $[-1, 1]$  via the formula  $\text{normalized}_x = (x - \text{center}_x) / \text{center}_x$ , and similarly for  $y$  (see Figure 1). This normalization helps concept attractors remain stable when the size, position, and proportions of the image change, allowing shapes to be recognized despite variations in viewing conditions.

## IV. CONCEPT FORMATION VIA STRUCTURAL REDUCTION

### A. From Multiple Graphs to Single Attractor

Concept formation proceeds through iterative structural composition following principles of architectural information representation and structural reduction [17]. Given training samples  $G_1, G_2, \dots, G_n$ , the algorithm initializes the concept with the first graph ( $C_0 = G_1$ ) and iteratively refines it through custom reduction operations (CRO):  $C_{i+1} = \text{CRO}(C_i, G_{i+1})$  for  $i = 1, 2, \dots, n-1$ . This formulation treats the first sample as a baseline that subsequent samples refine, preserving only structural elements common across all instances.

The integration process for each sample follows five coordinated steps: (1) align start points across both graphs using clustering-based selection to establish consistent traversal origins; (2) apply critical point preprocessing through iterative reduction strategies (see subsection IV-B) to achieve structural compatibility; (3) generate synchronized traversal paths that maintain correspondence between critical points in both graphs; (4) identify common structure along matched path segments by comparing all simple paths between consecutive critical points and selecting best matches via node similarity assessment; (5) merge geometric and semantic properties using type-specific integration strategies (see subsection IV-C). Each

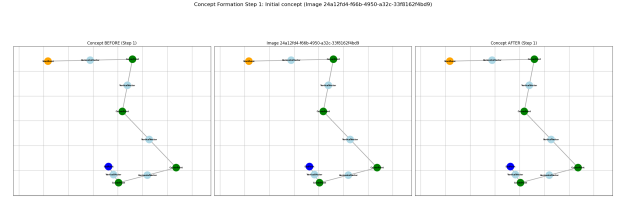


Fig. 2. Step 1 ( $C_0 = G_1$ ): The first training sample establishes the initial concept with complete structural detail including all endpoints, corner points, and intersection points.

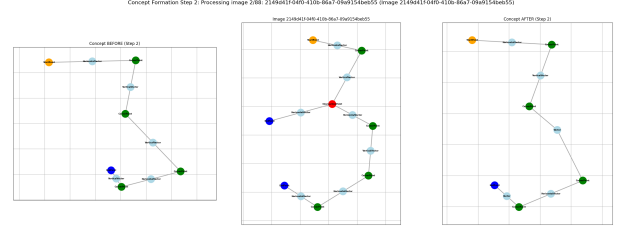


Fig. 3. Step 2 ( $C_1 = \text{MCM}(C_0, G_2)$ ): Integration of the second sample reveals a redundant endpoint branch. The algorithm removes the excess endpoint path and reduces the nearest IntersectionPoint to CornerPoint, then updates concept properties through parametric merging.

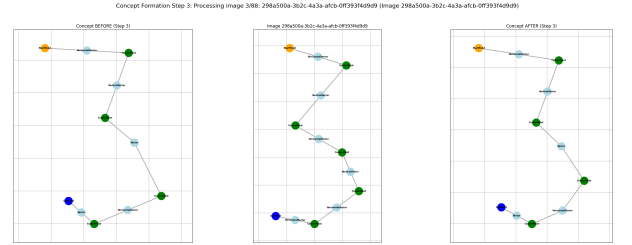


Fig. 4. Step 3 ( $C_2 = \text{MCM}(C_1, G_3)$ ): The third iteration identifies a redundant corner point in the lower structure. Removal and realignment of neighboring points maintains continuity while eliminating structural fragmentation. Property ranges continue to expand, capturing variation across samples.

iteration produces a refined concept  $C_i$  that represents the intersection of structural patterns encountered thus far.

Figures 2–5 illustrate this iterative reduction process across four training samples of a handwritten digit. The progression demonstrates how the algorithm systematically eliminates sample-specific variations while preserving essential structural patterns shared across all instances.

### B. Structural Reduction Rules

The custom reduction operations (CRO) employs three complementary reduction strategies to align critical point structures before path-level comparison. These strategies operate within a type hierarchy where  $\text{IntersectionPoint} \rightarrow \text{CornerPoint} \rightarrow \text{Point}$ , with the constraint that StartPoint and EndPoint types cannot be reduced as they define structural boundaries.

**Endpoint removal** eliminates terminal nodes that lack correspondence across samples. The algorithm computes a similarity matrix between endpoints in the concept and image graphs, removing endpoints with maximum similarity below given threshold. When endpoint counts differ, the algorithm

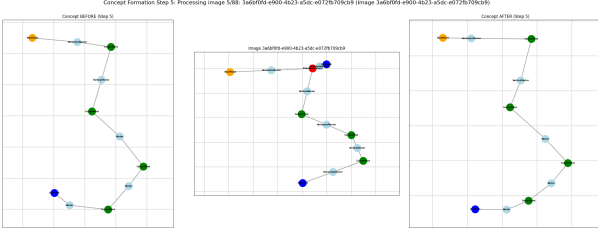


Fig. 5. Step 4 ( $C_3 = \text{MCM}(C_2, G_4)$ ): The fourth iteration detects noise manifested as a redundant substructure (endpoint and connecting path to nearest IntersectionPoint). Removal yields a refined concept capturing the canonical structural pattern with stabilized property distributions representing the concept attractor.

removes excess endpoints with the lowest maximum similarity scores. For each removed endpoint, the algorithm traverses from the terminal node toward the nearest critical point and removes the entire connecting path, eliminating sample-specific protrusions while preserving shared structural junctions.

**Intersection point merging** consolidates junction nodes representing the same structural feature. The algorithm first applies semantic reduction: any node labeled IntersectionPoint but having degree  $\leq 2$  is relabeled as CornerPoint (degree 2) or EndPoint (degree 1) to maintain type-topology consistency. When intersection counts differ between graphs, the algorithm identifies excess intersections via similarity matrix comparison (with given threshold) and removes those with lowest similarity scores. Removal proceeds by finding the path from the excess intersection to the nearest remaining critical point (preferring other intersections), removing intermediate nodes, and relinking preserved neighbors to the target junction. This operation reduces structural fragmentation while maintaining connectivity.

**Path pruning** normalizes segments between aligned critical points. For each pair of corresponding critical points, the algorithm identifies all simple paths connecting them and filters paths containing intermediate critical points to maintain segment isolation. It then computes node similarity matrices between all path pairs and selects the best matching pair based on average maximum similarity per node. Using the shorter path as a template, the algorithm matches nodes from the longer path via similarity scores and constructs the result path containing only matched positions. This template-based reduction captures essential connectivity patterns while eliminating length variations and intermediate detail specific to individual samples.

### C. Parametric Generalization

While structural reduction determines which nodes and edges persist in the concept graph, parametric generalization determines how their attributes are merged. The algorithm uses three different merging strategies based on attribute type, balancing the preservation of variations with the elimination of inconsistencies.

**Numeric properties** transform into range representations capturing central tendency and acceptable variation.

For values  $v_1, v_2, \dots, v_n$  from  $n$  training samples, the merged property becomes  $\{\min : \min_i v_i, \max : \max_i v_i, \text{center} : \frac{1}{n} \sum_i v_i, \text{type} : \text{"range"}\}$ . This strategy applies to coordinates (normalized\_x, normalized\_y), geometric measurements (length, angle), and topological counts (corner\_points\_count, intersection\_points\_count). For example, if three samples have corner\_points\_count values  $[5, 8, 3]$ , the merged concept stores  $\{\min : 3, \max : 8, \text{center} : 5.33\}$ , preserving both the expected value and the observed variability.

**Categorical properties** preserve only values consistent across all samples. For strings  $s_1, s_2, \dots, s_n$ , the merged property is  $s_1$  if  $s_i = s_1$  for all  $i$ , otherwise the property is removed entirely. This strict matching ensures that categorical attributes in the concept represent universal characteristics rather than incidental variations. For instance, contour\_type values  $[\text{"OPEN"}, \text{"OPEN"}, \text{"OPEN"}]$  merge to  $\text{"OPEN"}$ , while inconsistent horizontal\_direction values  $[\text{"Left"}, \text{"Right"}, \text{"Right"}]$  cause the property to be excluded from the concept.

**List properties** retain only elements present in all samples via set intersection: for lists  $L_1, L_2, \dots, L_n$ , the merged property is  $L_1 \cap L_2 \cap \dots \cap L_n$ . This strategy applies primarily to node labels, which encode multiple type classifications per node. For example, if three corresponding nodes have labels  $[[\text{"Point"}, \text{"CornerPoint"}], [\text{"Point"}, \text{"EndPoint"}], [\text{"StartPoint"}, \text{"Point"}]]$ , the merged labels become  $[\text{"Point"}]$ —the sole label common to all instances. Empty intersections indicate that no universal list-based features exist for that property, and the property may be excluded or represented as an empty set depending on semantic requirements.

These merging strategies ensure that concept representations capture statistically coherent patterns: range-based merging preserves natural variation in continuous parameters, exact matching filters inconsistent categorical labels, and set intersection retains only universal structural classifications. The resulting concept graphs encode not point estimates but *distributions* of acceptable values, enabling flexible matching during classification while maintaining interpretability through explicit parameter bounds.

## V. VALIDATION ON MNIST HANDWRITTEN DIGITS

### A. Experimental Setup

We validate the proposed graph-based concept formation approach on a subset of the MNIST handwritten digit dataset [18]. The experimental design emphasizes few-shot learning capability with minimal training data and interpretable graph structures.

**Dataset:** We selected six digit classes (1, 2, 3, 6, 7, 9) representing diverse structural complexity. These classes exhibit varying topological characteristics: digit 1 represents a simple linear structure, digits 2 and 3 exhibit curved open contours with multiple corner points, digits 6 and 9 contain closed loops with intersection points, and digit 7 demonstrates angular transitions. This selection provides sufficient structural

diversity to evaluate the representation’s expressive power while maintaining experimental tractability.

**Training samples:** Each digit class was further divided into structural subclasses to capture distinct writing styles. Training data consisted of 2–4 manually selected samples per subclass, with 8 concepts total across the six digit classes (1\_1, 1\_3, 2\_1, 2\_2, 3\_1, 6\_1, 7\_1, 9\_2). This few-shot configuration deliberately constrains the algorithm to learn from minimal supervision, testing whether structural reduction can extract generalizable patterns from sparse data without gradient-based optimization.

**Data augmentation:** To increase training set diversity while preserving structural integrity, we applied geometric transformations to each original training sample. Each image was augmented by generating 10 variants through random rotation (uniformly sampled angles within  $\pm 10^\circ$ ) and spatial translation (up to 10% of image dimensions in both horizontal and vertical directions). Images were rendered at  $100 \times 100$  pixels in grayscale, with transformations applied on a black background to maintain boundary clarity. This augmentation strategy simulates natural handwriting variation—slight angular rotations and positional shifts—without introducing unrealistic distortions that would compromise structural correspondence.

**Test set:** Classification performance was evaluated on 5467 test images drawn from the standard MNIST test split, with approximately 750–1000 samples per digit class (997 for digit 1, 948 for digit 2, 983 for digit 3, 753 for digit 6, 990 for digit 7, 786 for digit 9). No test images were used during concept formation, ensuring unbiased evaluation of generalization capability.

**Graph extraction pipeline:** Each image underwent a multi-stage processing pipeline to produce attributed graph representations. First, binary thresholding with adaptive threshold selection (converging to  $\theta = 180$  for connectivity) converted grayscale images to binary masks. Second, morphological skeletonization via medial axis transformation reduced contours to single-pixel-width skeletal structures while preserving topology. Third, Growing Neural Gas (GNG) learning fitted a network topology to skeleton point clouds, capturing structural connectivity. Fourth, Ramer-Douglas-Peucker simplification reduced network complexity while maintaining geometric fidelity. Fifth, graph construction generated NetworkX representations with Point and Line nodes, encoding critical points (endpoints, corners, intersections) and connecting line segments as first-class entities. Finally, coordinate normalization transformed absolute pixel positions to a centered  $[-1, 1]$  range, ensuring scale and translation invariance. This pipeline, detailed in prior sections, produces compact graph representations suitable for structural comparison and concept formation.

## B. Concept Characteristics

The iterative reduction process transforms multiple training graphs into single concept attractors characterized by explicit

TABLE I  
CONCEPT GRAPH STRUCTURAL METRICS

Concept	—V—	—E—	Mean Degree	Critical Points
1_1	3	2	1.33	1 EP, 1 SP
1_3	3	2	1.33	1 EP, 1 SP
2_1	7	6	1.71	1 EP, 2 CP, 1 SP
2_2	12	12	2.00	1 EP, 3 CP, 1 IP, 1 SP
3_1	7	6	1.71	1 EP, 2 CP, 1 SP
6_1	10	10	2.00	3 CP, 1 IP, 1 SP
7_1	5	4	1.60	1 EP, 1 CP, 1 SP
9_2	8	8	2.00	2 CP, 1 IP, 1 SP

EP = EndPoint, CP = CornerPoint, IP = IntersectionPoint, SP = StartPoint

structural metrics and parametric distributions. Table I summarizes graph-theoretic properties for all eight learned concepts, revealing systematic relationships between digit topology and representation complexity.

Structural complexity correlates with digit topology. Simple linear structures (digit 1) yield minimal graphs with only 3 nodes and mean degree 1.33, reflecting unbranched paths from start to endpoint. Curved open contours (digits 2, 3, 7) produce moderate-sized graphs (5–12 nodes) with mean degrees between 1.60 and 2.00, where corner points encode directional transitions. Closed-loop digits (6, 9) exhibit higher mean degrees (2.00) due to intersection points that create cyclic paths, increasing connectivity beyond simple chains. The most complex concept (2\_2) contains 12 nodes including an intersection point and three corner points, capturing the structural variability observed across diverse handwriting styles for digit 2.

Critical point distributions reveal topological distinctions: all concepts except closed-loop digits (6, 9, 2) contain exactly one endpoint, signifying open contours with terminal segments. Concepts 6\_1 and 9\_2 lack endpoints but contain intersection points where the loop closes, a topological signature differentiating circular from linear structures. Corner point counts range from 1 (digit 7) to 3 (digits 2\_2, 6), reflecting the number of directional changes required to trace each digit’s canonical form.

**Concept formation progression for Digit 3:** The evolution from three training samples to a single concept attractor demonstrates how parametric generalization encodes acceptable variation while preserving structural identity. Initially, each training sample produces a distinct graph with specific coordinate values, angle measurements, and segment lengths. Through iterative reduction (subsection IV-A), the algorithm merges these graphs by: (1) aligning start points via spatial clustering; (2) removing sample-specific endpoints with low cross-sample similarity; (3) consolidating corner points that represent shared directional transitions; (4) merging line attributes into range distributions that span observed variation.

The resulting concept graph for digit 3 (concept 3\_1) contains 7 nodes (4 Points, 3 Lines) with 6 edges, encoding the essential “curved S-shape” topology common across training instances. Parametric properties transform from point estimates to flexible ranges: normalized x-coordinates span  $[-0.7, 0.2]$  with center  $-0.33$ ; normalized y-coordinates span

[0.3, 0.9] with center 0.63. Structural counts similarly encode variation: endpoint counts range [2, 4] with center 2.67; intersection point counts range [0, 2] with center 0.67; line counts range [4, 9] with center 6.0. Categorical properties that hold consistently across samples (e.g., `contour_type` = OPEN, `monotony` = NON\_MONOTONIC) persist as exact values, while inconsistent properties are excluded from the concept.

This parametric generalization enables the concept to match test samples that vary continuously within learned bounds, providing flexible recognition without overfitting to training-specific details. The range-based encoding is interpretable: attribute centers indicate typical values, while min-max bounds reveal the extent of acceptable deviation, making the concept’s decision boundaries explicit and queryable.

### C. Classification via Graph Matching

Classification proceeds by comparing each test image’s graph representation against all concept attractors, computing structural similarity via Graph Edit Distance (GED), and selecting the concept with highest similarity score. The matching process incorporates the structural reduction strategies described in subsection IV-B to align test graphs with concept structures before distance computation.

**Graph Edit Distance methodology:** GED quantifies structural dissimilarity as the minimum-cost sequence of edit operations (node/edge insertion, deletion, substitution) required to transform the test graph into a concept graph. We employ custom cost functions that incorporate semantic and geometric constraints. Node substitution costs enforce label compatibility (concept labels must form a subset of test graph labels) and compute weighted property differences across shared attributes, with range-based cost functions enabling flexible matching against parametric distributions (subsection IV-C). Node deletion and insertion carry unit costs, while edge operations incur reduced costs (0.1) to prioritize topological over connectivity differences. The optimal edit path is computed via dynamic programming with a 60-second timeout per concept comparison, converting raw GED values to normalized similarity scores via  $\text{similarity} = 1.0 - (\text{GED}/\text{max\_cost})$ , where  $\text{max\_cost} = |V_{\text{test}}| + |V_{\text{concept}}|$  represents the theoretical upper bound for complete graph replacement.

**Overall performance:** Across 5467 test images, the system achieved 82.35% accuracy with 83.28% precision, 82.35% recall, and 82.16% F1 score (Table II). The pipeline success rate reached 100%, with only 10 images (0.18%) failing to complete processing due to skeletonization errors that produced disconnected graphs routed to a dead-letter queue. These results demonstrate that few-shot concept formation from 2–6 training samples per subclass can achieve meaningful classification performance without gradient-based optimization or deep feature extraction, relying solely on explicit structural comparison.

**Per-class performance:** Table III reveals substantial variation in classification accuracy across digit classes, correlated with structural distinctiveness. Digits 6 and 9 achieve the highest precision (94.23%, 91.55%) and F1 scores (85.40%,

TABLE II  
OVERALL CLASSIFICATION PERFORMANCE (5467 TEST IMAGES)

Metric	Value (%)
Accuracy	82.35
Precision	83.28
Recall	82.35
F1 Score	82.16

TABLE III  
PER-CLASS CLASSIFICATION METRICS

Digit	Precision (%)	Recall (%)	F1 (%)	Support
1	81.46	96.49	88.34	997
2	84.17	60.02	70.07	948
3	78.21	87.28	82.50	983
6	94.23	78.09	85.40	753
7	74.38	82.12	78.06	990
9	91.55	89.57	90.55	786

90.55%), likely due to their unique topological signatures: both contain closed loops with intersection points, distinguishing them from open-contour digits. Digit 7 exhibits the lowest precision (74.38%) and F1 score (78.06%), suffering from structural ambiguity with digit 1 (both are angular open contours with similar orientations, differing primarily in corner sharpness). Digits 2 and 3, despite sharing curved morphology, demonstrate moderate performance (precision 84.17% and 78.21%, respectively), with digit 2 achieving higher precision at the cost of lower recall (60.02%).

**Confusion matrix analysis:** Figure 6 visualizes the classification confusion matrix, highlighting systematic misclassification patterns. The primary confusion occurs between digits 2 and 3 (152 misclassifications of 2 as 3, 28 misclassifications of 3 as 2), reflecting their overlapping curved morphology where corner point positions and line orientations vary subtly. Secondary confusions appear between digits 7 and 1 (118 test images of digit 7 misclassified as 1), attributable to their shared angular open-contour structure with similar start-to-endpoint paths. Digits 6 and 9 exhibit minimal mutual confusion (48 and 4 misclassifications, respectively), confirming that their closed-loop topology provides discriminative power despite geometric similarity. The confusion matrix reveals that errors concentrate along structurally similar digit pairs, validating the hypothesis that graph-based representation captures meaningful topological distinctions while remaining sensitive to ambiguous boundary cases where structural differences are subtle.

**Few-shot learning evaluation:** The experimental results demonstrate that the structural reduction approach achieves competitive classification performance using only 2–6 training samples per concept—orders of magnitude fewer than typical supervised learning baselines. Traditional convolutional networks require hundreds to thousands of labeled examples per class to achieve comparable accuracy on MNIST, while prototypical networks and meta-learning approaches necessitate task-level training across numerous episodes. In contrast, the present approach forms concepts through direct structural composition without gradient descent, episodic sampling,



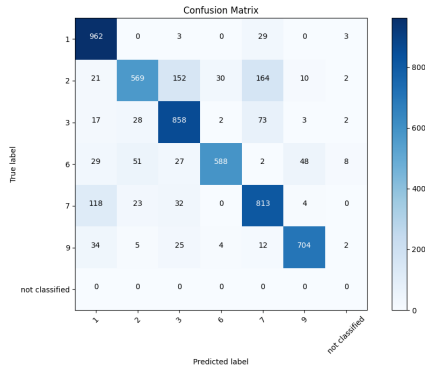


Fig. 6. Confusion matrix for 6-class MNIST digit classification. Primary confusion occurs between digits 7 and 1 (angular open contours), and secondary confusion between digits 2 and 3 (curved open contours). Closed-loop digits (6, 9) exhibit strong discrimination.

or auxiliary training tasks. The 82.35% accuracy represents meaningful few-shot capability, particularly given that the method operates on explicitly interpretable graph structures rather than opaque embeddings. This result validates the hypothesis that structural reduction can extract generalizable patterns from minimal supervision, enabling concept formation that is both data-efficient and transparent.

## VI. DISCUSSION

### A. Explainability Through Structure

The graph representation makes structure the carrier of explanations, providing inherent interpretability without post-hoc approximation. Graph cycles directly encode topological properties: closed-loop digits (6, 9) exhibit mean degree 2.0 and contain intersection points, while open-contour digits exhibit lower mean degrees (1.33–1.71) and terminal endpoints (Table I). Parametric generalization (subsection IV-C) encodes decision boundaries as explicit attribute ranges rather than implicit weight matrices: normalized coordinate spans (e.g.,  $x \in [-0.7, 0.2]$  for digit 3) define acceptable variation transparently, while categorical properties persist only when consistent across all training samples.

This approach contrasts fundamentally with post-hoc explanation methods (Section 2). LIME and SHAP generate attributions by perturbing inputs and approximating model behavior locally, yielding explanations vulnerable to adversarial manipulation [2]. In contrast, the present system embeds semantics directly in graph elements: explanations are navigable objects derived from structure itself. The confusion matrix (Figure 6) validates this interpretability—misclassifications concentrate along structurally similar pairs (digits 2/3 share curved morphology, digits 7/1 share angular open contours)—demonstrating that classification decisions trace directly to queryable topological and geometric properties.

### B. Limitations and Future Work

The approach exhibits four primary limitations. First, Graph Edit Distance computation is NP-hard in the general case,

requiring a 60-second timeout per concept comparison during classification (MNIST average time per image is 3.5 seconds). While tractable for moderate-sized graphs ( $|V| \leq 12$  in our MNIST concepts), this complexity may hinder real-time applications or large concept libraries without approximation algorithms. Second, graph quality depends critically on the preprocessing pipeline: skeletonization failures produced disconnected graphs for 0.18% of test images, and alternative contour extraction methods may yield different structural representations. Third, rotation invariance remains limited to the augmentation range employed during training ( $\pm 10^\circ$ ); larger rotations may disrupt structural alignment and reduce matching accuracy. Fourth, the contour-only representation discards texture and gradient information, limiting applicability to recognition tasks where surface appearance matters.

## VII. CONCLUSION

This paper presents a graph-based representation for contour images that makes structure the carrier of explanations. The approach encodes visual patterns as attributed graphs where nodes represent critical points and connecting line segments, with semantic tags and geometric attributes attached directly to graph elements. Multiple graph instances are reduced to stable concept attractors through iterative structural composition, requiring only 2–6 training samples per concept without gradient-based optimization. Validation on MNIST handwritten digits demonstrates that this few-shot learning approach achieves 82.35% classification accuracy through explicit graph matching, with confusion patterns tracing directly to structural similarity. By encoding semantics in queryable graph topology rather than opaque weight matrices, the approach advances explainable AI toward systems where decision paths are inherently transparent, verifiable, and suitable for domains requiring trustworthy, traceable reasoning.

## REFERENCES

- [1] M. T. Ribeiro, S. Singh, and C. Guestrin, “‘’ why should i trust you?’’ explaining the predictions of any classifier,” in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*, 2016, pp. 1135–1144.
- [2] D. Slack, S. Hilgard, E. Jia, S. Singh, and H. Lakkaraju, “Fooling lime and shap: Adversarial attacks on post hoc explanation methods,” 2020. [Online]. Available: <https://arxiv.org/abs/1911.02508>
- [3] D. Hooshyar and Y. Yang, “Problems with shap and lime in interpretable ai for education: A comparative study of post-hoc explanations and neural-symbolic rule extraction,” *IEEE Access*, vol. 12, pp. 137 472–137 490, 2024.
- [4] E. Rajabi and K. Etminani, “Knowledge-graph-based explainable ai: A systematic review,” *Journal of Information Science*, vol. 50, no. 4, pp. 1019–1029, 2024. [Online]. Available: <https://doi.org/10.1177/01655515221112844>
- [5] S. M. Lundberg and S.-I. Lee, “A unified approach to interpreting model predictions,” *Advances in Neural Information Processing Systems*, vol. 30, 2017.
- [6] K. Han, Y. Wang, J. Guo, Y. Tang, and E. Wu, “Vision gnn: An image is worth graph of nodes,” in *Advances in Neural Information Processing Systems*, vol. 35, 2022, pp. 8305–8319.
- [7] H. Chatbri, K. Kameyama, and P. Kwan, “A comparative study using contours and skeletons as shape representations for binary image matching,” *Pattern Recognition Letters*, vol. 76, pp. 59–66, 2016.

- [8] W. Shen, Y. Wang, X. Bai, H. Wang, and L. J. Zhang, "Shape recognition by bag of skeleton-associated contour parts," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 2125–2133.
- [9] Y. Parzhin, "Principles of modal and vector theory of formal intelligence systems," 2013. [Online]. Available: <https://arxiv.org/abs/1302.1334>
- [10] Y. Parzhin, S. Galkyn, and M. Sobol, "Method for binary contour images vectorization of handwritten characters for recognition by detector neural networks," in *2022 IEEE 3rd KhPI Week on Advanced Technology (KhPIWeek)*, Kharkiv, Ukraine, 2022, pp. 1–6.
- [11] F. Wang, P. Jiang, K. Riesen, and J. Zhang, "Combinatorial learning of graph edit distance via dynamic embedding," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 5184–5193.
- [12] C. Liu *et al.*, "Mata: Multi-attribute time series anomaly detection via meta-learning and attentive adaptive framework," in *Proceedings of the VLDB Endowment*, vol. 16, no. 12, 2023, pp. 3983–3995.
- [13] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *Proceedings of the 34th International Conference on Machine Learning*, 2017, pp. 1126–1135.
- [14] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. NeurIPS*, 2017.
- [15] U. Nawaz, M. Anees-ur Rahaman, and Z. Saeed, "A review of neuro-symbolic ai integrating reasoning and learning for advanced cognitive systems," *Intelligent Systems with Applications*, p. 200541, 2025.
- [16] Y. Parzhyn, M. Lapin, and K. Bokhan, "A new approach to building energy models of neural networks," *Advanced Information Systems*, vol. 9, no. 4, pp. 100–119, 2025.
- [17] Y. Parzhyn, "Architecture of information," 2025.
- [18] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 2002.