



Minerva Access is the Institutional Repository of The University of Melbourne

Author/s:

Huang, Z.;Khoshelham, K;Tomko, M

Title:

Learning geometric invariant features for classification of vector polygons with graph message-passing neural network

Date:

2025-10-01

Citation:

Huang, Z., Khoshelham, K. & Tomko, M. (2025). Learning geometric invariant features for classification of vector polygons with graph message-passing neural network. *Geoinformatica*, 29 (4), pp.951-974. <https://doi.org/10.1007/s10707-025-00554-y>.

Persistent Link:

<https://hdl.handle.net/11343/362157>

License:

[CC-BY](#)



Learning geometric invariant features for classification of vector polygons with graph message-passing neural network

Zexian Huang¹ · Kourosh Khoshelham¹ · Martin Tomko¹

Received: 13 May 2024 / Revised: 21 April 2025 / Accepted: 3 June 2025 /

Published online: 16 June 2025

© The Author(s) 2025

Abstract

Geometric shape classification of vector polygons remains a challenging task in spatial analysis. Previous studies have primarily focused on deep learning approaches for rasterized vector polygons, while the study of discrete polygon representations and corresponding learning methods remains underexplored. In this study, we investigate a graph-based representation of vector polygons and propose a simple graph message-passing framework, PolyMP, along with its densely self-connected variant, PolyMP-DSC, to learn more expressive and robust latent representations of polygons. This framework hierarchically captures self-looped graph information and learns geometric-invariant features for polygon shape classification. Through extensive experiments, we demonstrate that combining a permutation-invariant graph message-passing neural network with a densely self-connected mechanism achieves robust performance on benchmark datasets, including synthetic glyphs and real-world building footprints, outperforming several baseline methods. Our findings indicate that PolyMP and PolyMP-DSC effectively capture expressive geometric features that remain invariant under common transformations, such as translation, rotation, scaling, and shearing, while also being robust to trivial vertex removals. Furthermore, we highlight the strong generalization ability of the proposed approach, enabling the transfer of learned geometric features from synthetic glyph polygons to real-world building footprints.

Keywords Spatial vector polygons · Geometric shape classification · Transformation invariance · Permutation invariance · Deep neural networks

Kourosh Khoshelham and Martin Tomko contributed equally to this work.

✉ Zexian Huang
zexianh@student.unimelb.edu.au

Kourosh Khoshelham
k.khoshelham@unimelb.edu.au

Martin Tomko
tomkom@unimelb.edu.au

¹ Department of Infrastructure Engineering, The University of Melbourne, Grattan Street, Melbourne, 3010, Victoria, Australia

1 Introduction

Geometric shape classification of spatial objects is a non-trivial task in spatial analysis. The recognition of spatial objects assisted by automated shape classification is a major enabler of data intensive tasks, including cartographic generalisation, building pattern recognition, archaeological feature analysis, and road geometry identification [1–3].

One of the main challenges in geometric shape classification is the identification of object footprints (i.e., outlines) in the geographic context. A key requirement is the classification of objects invariant to geometric transformations including rotation, scaling, and shearing. The human visual system relies on the Gestalt properties of perceived objects [4, 5] during the classification of shapes. A key property of Gestalt is the invariance to geometric transformations, assuring that geometric shapes are recognized regardless of geometric transformations. In contrast to the human visual perception capabilities, deep learning architectures have been designed and shown to be mainly translation-invariant (e.g., Convolution Neural Networks, CNN) or permutation-invariant (e.g., Graph Neural Networks, GNN) on classification tasks. The ability to reflect Gestalt principles through transformation invariances thus present a strong motivation for an inductive bias in the design of learning-based models for geometric shape recognition of spatial objects.

Spatial objects are often conveniently represented as vector polygons, a discrete data representation thus far neglected in deep learning research. Learning geometric-invariant features from vector polygons has the following requirements: (1) a generic data representation that encodes geometric features of polygons without information loss; and (2) a learning model built on this input data representation that enables learning latent geometric-invariant features robust to geometric transformations. Existing geospatial applications (i.e., shape coding and retrieval [6], building pattern recognition [1, 2] and building grouping [7]) utilizing polygonal geometries motivate us to systemically study geometry encoding methods in conjunction with appropriate and robust learning architectures that enable learning transformation invariant features of spatial polygon geometries.

Veer et al. [2] proposed VeerCNN, a deep convolution model to learn convolutional features on fixed-size 1D vertex sequences of polygon vertices for building attribute recognition. The architectural limitations of CNNs attributable to shared-weights convolutional kernels followed by non-linear activation and fixed-size pooling layers (e.g., *mean*, *sum* or *max*) only enable learning intermediate hierarchical features invariant to translation, with poor handling of rotation, scaling and shearing. Mai et al. [8] noted that CNN models learning on 1D sequences are also sensitive to (1) permutations of the feed-in order of polygon vertices (i.e., sensitive to *loop origin invariance*); and (2) the impact of trivial vertices on the exterior and interiors of polygons, defined by Mai et al. [8] as “[vertices]...where the addition or removal of the vertex have no effect on the geometric shape and topological properties of the outlines of the polygons”. The addition or removal of trivial vertices in polygons does not alter the semantic information of the shape (as captured by the human-assigned label), closely aligning with the Gestalt principles.

Learning models tailored for discrete (i.e., not grid-like), permutation invariant data representations are predicated on data domains where entries hold no explicit neighborhood information (e.g., point sets [9–12]). Polygons can then be encoded into point sets of vertices on the exterior and interior rings of the polygons with arbitrary ordering. In contrast to the 1D sequence encoding of polygons, point set encoding assumes input data to have

varying input sizes and feed-in order. Models learning on point sets are designed to map input points with permuted order to task-specific outputs (i.e., *permutation invariance* [13]). Here, we argue that such point set representations do not sufficiently capture the connectivity information between vertices of polygons, leading to information loss and performance degradation in geometric shape classification.

Graph data structures present a suitable data encoding for polygons. Recent studies [6, 14, 15] convert vector polygons into un-directed graphs where the vertices on the exteriors and interiors of polygons are captured as graph nodes linked by un-directed graph edges. Compared to fixed-size 1D sequences and varying-size point-set encoding of polygons, graph encoding effectively encapsulates both the geometric and connectivity information of the vertices along the exterior and interior linear rings defining the polygons. Graph-based representations also enable the feed-in order of polygon vertices to be independent of the model outcomes, while the connectivity (topology) between parts of polygon representations remains invariant to geometric transformations.

Graph convolutional autoencoders (GCAE) [6] extend graph convolutional neural networks (GCNs) [16] by learning spectral graph embeddings from polygon graphs, demonstrating the effectiveness of graph latent embeddings in polygon shape retrieval. Bronstein et al. [13] noted that spectral graph convolutions aggregate node features from neighboring nodes with constant weights, making them highly dependent on the topological structure of the input graphs. In vector polygons, vertices on linear rings have fixed neighbors (i.e., the left and right adjacent vertices), which reduces the expressivity of graph convolutional features learned from polygon graphs. In this study, we leverage graph message-passing mechanisms [17] and propose a simple graph message-passing network, PolyMP, along with its densely self-connected variant, PolyMP-DSC, to learn more expressive and robust latent representations of polygons. We hypothesize that combining graph-based representations of polygons with graph message-passing models facilitates learning robust latent features that remain invariant to geometric transformations (i.e., rotation, scaling, and shearing), thereby improving the generalizability of shape classification across datasets.

If our hypothesis holds, we suggest that downstream tasks performed on the learned features of vector polygons will exhibit robust performance on shapes with varying amounts of trivial vertices. We present a series of experiments designed to evaluate model robustness on polygons subjected to geometric transformations and investigate the generalizability of our findings across different combinations of polygon representations and model architectures (permutation-invariant vs. translation-invariant models) for classifying vector polygons with and without holes (inner linear rings).

Following a literature review (Section 2), we introduce a synthetic dataset of polygonal shapes with high shape variability and well-defined human labels, derived from Latin alphabet character glyphs (Section 4.1). Despite significant shape variations across fonts, particularly in terms of trivial vertices, characters maintain strong recognizability due to their Gestalt properties. We then benchmark the performance of learning models on three distinct data representations (1D sequence, set, and graph) under varying geometric transformations (i.e., rotation, scaling, and shearing). Our results (Section 5) document the models' robustness to geometric transformations through performance evaluation on the synthetic glyph dataset. Building on these findings, we further assess model generalizability using a real-world building footprint dataset from OpenStreetMap (OSM) [6].

Our main contributions are:

1. we provide a systematic investigation of vector polygon representations for geometric shape classification, highlighting the strengths of graph-based encoding over sequences and point sets;
2. the introduction of PolyMP and PolyMP-DSC, two lightweight graph message-passing architectures designed to learn geometric-invariant and robust latent representations;
3. the release of a synthetic dataset of vector polygons derived from character glyphs, enabling controlled benchmarking of geometric learning models; and
4. extensive experiments which show that the proposed message-passing models, PolyMP and PolyMP-DSC, significantly improve robustness to transformations and generalize well to real-world geospatial data.

Together, these contributions aim to provide a principled and practical foundation for learning geometric-invariant representations of spatial polygons—pushing forward the capabilities of shape-based analysis in geospatial domains.

2 Background

2.1 Machine learning with vector geometries

Building on computer vision approaches for image classification, polygonal shape classification has traditionally relied on learning geometric features from rasterized 2D vector geometries using deep CNN architectures [18, 19]. The geospatial community has adopted these methods for various remote sensing tasks, including vector shape generation. For example, the Microsoft Open Building Footprints dataset [20] was generated by segmenting and vectorizing building polygons from satellite and aerial imagery using deep semantic segmentation networks [21].

This expertise in image-based learning naturally influenced vector shape learning. Xu et al. [22] trained a deep convolutional autoencoder to assess the quality of rasterized building footprints collected from OpenStreetMap (OSM) [23]. Similarly, Veer et al. [2] evaluated deep neural networks for classifying spatial vector geometries, encoding geometry vertices as 1D sequences directly fed into deep models. While CNNs and RNNs performed comparably to hand-crafted feature-based methods, they lacked the ability to learn transformation-invariant shape features.

Beyond geospatial applications, typography and computer graphics research also leverage vector shape learning. Lopes et al. [24] developed a convolution-based generative model to create vector text glyphs, but it was optimized primarily for scale-invariant font style learning. Mino and Spanakis [25] and Sage et al. [26] generated vector icons and logos from rasterized images using deep generative autoencoders [27, 28]. Additionally, Ha and Eck [29] introduced Sketch-RNN, a recurrent neural network for generating stroke-based vector drawings, while Carlier et al. [30] proposed DeepSVG, a hierarchical generative model using Transformer-based encoding for scalable vector graphics (SVG).

2.2 Vector geometry as discrete data representation

Recent advances in graph representation learning [16, 31–33] have reformulated vector polygon learning as a graph-based or set-based problem. Yan et al. [1] applied graph convolution networks (GCNs) to building pattern classification, structuring buildings into clusters based on geometric relationships using Delaunay triangulation and Minimum Spanning Trees. Their GCN model classified building clusters into regular or irregular patterns. Similarly, Bei et al. [15] introduced a spatially adaptive model using graph encoding for group pattern recognition, where buildings served as nodes in a graph convolutional network (GCN).

Yan et al. [6] proposed a graph convolutional autoencoder (GCAE) for building shape analysis, encoding building polygons as graphs where boundary vertices acted as nodes. However, their method was sensitive to rotations, limiting its robustness for polygon shape retrieval. Addressing this limitation, Huang et al. [34] introduced a contrastive graph autoencoder to enable robust polygon shape matching and retrieval in large-scale vector datasets.

Liu et al. [35] developed a deep point convolutional network (DPCN), a modification of a DGCNN [36], for building shape recognition in map space. Their model introduced the TriangleConv operator, which extracts local geometric features from triangle-based representations of polygon vertices: $f^{\text{TriangleConv}}(x_{i-1}, x_i, x_{i+1}) = \{(x_i - x_{i-1}), (x_i - x_{i+1}), (x_{i-1} - x_{i+1})\}$, where (x_{i-1}, x_i, x_{i+1}) are adjacent point features of a building polygon. Despite achieving competitive classification performance on building footprints [6], this method requires pre-defining local triangles of polygons and repeatedly computing redundant local geometric features (i.e., angles and areas) of polygons. Furthermore, it does not evaluate the robustness of learned representations to geometric transformations such as rotation, scaling, and vertex perturbations.

Mai et al. [8] introduced ResNet1D, a 1D CNN-based polygon encoder that captures local geometric features for shape classification. It encodes polygon vertices using relative offsets from neighboring vertices: $\{x_i, x_{i-1} - x_i, x_{i+1} - x_i, \dots, x_{i+k} - x_i, x_{i-k} - x_i\}$. They compared this encoding with NUFTSpec, a Non-Uniform Fourier Transform (NUFT)-based method that aggregates global shape features in the spectral domain. While ResNet1D focuses on shape classification, NUFTSpec is more suited for predicting topological relations. Their work aligns with our study in exploring different polygon encoding methods, but our focus is on evaluating transformation-invariant learning architectures across various representations (CNN, set-based, graph-based, and attention-based models).

While previous studies explore graph-based and local geometric encoding for vector polygons, they often lack a principled justification for applying graph learning methods to polygon representations. We address this gap by providing: 1. Experimental validation grounded in theoretical motivation for robust polygon shape learning; 2. A principled framework demonstrating how permutation-invariant learning architectures combined with discrete data representations improve robustness to geometric transformations and vertex perturbations.

2.3 Graph representation learning

Graph representation learning model leverages the property of data permutations and learns a whole graph embedding \mathcal{G}^* from the input graph \mathcal{G} with differentiable permutation-invariant function \mathcal{F} . The learned embedding \mathcal{G}^* can be applied to graph-wise and node-wise classifications, or even edge predictions.

By representing polygon linear rings as graphs, we can define a differentiable permutation-invariant function \mathcal{F} , which takes the graph representations as inputs and returns a corresponding compact graph embedding $\mathcal{F}(\mathcal{G}; \theta) := \mathcal{F}(X, A; \theta) \rightarrow \mathcal{G}^* \in \mathcal{R}$.

Given a permutation matrix \mathcal{P} acting on a graph \mathcal{G} , it reorders the node feature matrix X and the adjacency matrix A , producing a permuted graph representation $\mathcal{G}' = (PX, PAP^T) = (X', A')$. The ideal permutation-invariant function \mathcal{F} should satisfy $\mathcal{F}(X', A'; \theta) = \mathcal{F}(X, A; \theta)$. We relate the idea of permutation matrix \mathcal{P} to the aforementioned geometric transformations.

In the view of invariant feature learning, the permutation invariance of \mathcal{F} outputs are guaranteed through the local aggregations of node features of $x \in X$ and the linear transformations of aggregated node features:

$$\mathcal{F}(X; \theta) = \rho\left(\sum_{x \in X} \phi(x, \theta)\right).$$

The local differentiable function ϕ linearly transforms each node feature individually x to latent space and the aggregation function sums over latent node features¹, and ρ is a global differentiable function applied on the summed node features followed by a non-linear activation function. In this simple setting, the outputs of \mathcal{F} are invariant to the permutation of node features since the summation-based aggregation returns the same outputs for any input permutation: $Aggr_{sum}(x_1, x_2, \dots, x_n) = Aggr_{sum}(x_n, \dots, x_2, x_1)$.

This framework (i.e., DeepSet [9]) assumes that permutations affect only individual node features while disregarding their connectivity, which is crucial for polygonal structures. Graph convolution neural networks (GCN) by Kipf and Welling [16] learn convolutional features of node matrix X with the normalized graph Laplacian matrix $\hat{A} = \tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2}$, where $\tilde{A} = A + I$ is the adjacency matrix with a identity matrix (i.e., self-connections of nodes), and \tilde{D} is the diagonal degree matrix. The graph Laplacian matrix describes the divergence of energy flowing from source nodes to target nodes, analogous to the spatial relationships in polygonal structures. Specifically, it captures connectivity patterns both within polygons (e.g., rings with holes) and between adjacent polygons.

Thus, GCN can learn graph Laplacian embedding of node feature with a differentiable neural network layer as follows:

$$\begin{aligned} \mathcal{F}_{conv}(X, A; \theta) &= \tilde{D}^{-1/2} \tilde{A} \tilde{D}^{-1/2} X \theta \\ &= \hat{A} X \theta. \end{aligned} \tag{1}$$

We can express (1) from a vector-wise view:

¹The basic aggregation in this setting can also be averaging or max pooling.

$$f(x_i; a_{ij}, \theta) = \sum_{\substack{a_{ij} \in A, \\ x \in X}} \phi(x_i, x'_j, \theta), \text{ where } x'_j = \frac{\tilde{a}_{ij}}{\sqrt{\tilde{d}_i \times \tilde{d}_j}} x_j, \tag{2}$$

$\frac{\tilde{a}_{ij}}{\sqrt{\tilde{d}_i \times \tilde{d}_j}}$ corresponds to the edge weight between nodes x_i and x_j in the normalized graph Laplacian matrix \hat{A} . From (2), the latent feature of target node x_i depends on the relation with its neighbour node x_j and constant edge weight $\frac{\tilde{a}_{ij}}{\sqrt{\tilde{d}_i \times \tilde{d}_j}}$. However, the constant edge weight between nodes x_i and x_j largely limits the expressivity of latent node features of polygonal graph since every node has a constant node degree ($d = 2$).

As discussed in Yan et al. [6], graph autoencoders GCAE based on GCN are sensitive to orientation and rotation of polygons, and therefore current graph-based learning methods are not optimized for geospatial settings with polygonal geometries of arbitrary orientation, or conversely, arbitrary observer orientation. Here we propose to leverage the message-passing mechanism [17, 37] with the permutation-invariant function \mathcal{F} to learn expressive and robust latent features of polygons.

3 Approach

3.1 Graph representation of polygon

Polygons are sets of points connected by lines, forming a collection of clockwise or counter-clockwise linear rings [38]. According to the Gestalt principles of invariance, the compact embedding of linear rings should ideally be invariant to geometric transformations such as rotation, scaling, and shear. We define a graph $\mathcal{G} = (X, E)$, where X is a node matrix containing the coordinates of the geometry vertices, and E is an edge matrix capturing the connectivity of these vertices along the linear rings (i.e., the boundaries of the polygon and any holes). The adjacency matrix A encodes the connectivity between nodes, where a binary value $a_{ij} = 1$ indicates that nodes i and j are connected.

3.2 Message-passing encoder

The principle of message-passing involves aggregating neighboring node features in graphs based on messages computed between source and target nodes. We define the message-passing function operating on a single node as:

$$h_i = \phi(x_i, msg(x_i, x_j), \theta), \tag{3}$$

where $\{x_i, x_j\} \in A = 1$ and $msg(\cdot)$ is the message function that takes the source and target nodes as inputs. From (3), the node feature of x_i is updated with the computed message (x_i, x_j) and transformed linearly into latent space, producing the latent feature $h_i \in \mathcal{H}$. The permutation-invariant message-passing encoder is represented in vector form as:

$$\mathcal{F}_{msg}(X, A; \theta) = \rho_\theta(readout(h_i | h_i \in \mathcal{H})). \tag{4}$$

, where a global readout function (e.g., mean or max pooling) aggregates the latent node-wise features (\mathcal{H}) and computes a global graph embedding by applying a non-linear transformation to the pooled features via ρ_θ .

Building on geometric message-passing methods for point clouds [11, 36] and polygons [35], we define a specific message-passing function $msg(\cdot)$ for PolyMP as:

$$msg(x_i, x_j) = \alpha * \frac{(x_j - x_i)}{std_{\mathcal{N}_{x_i}} + \epsilon} + \beta. \tag{5}$$

, where α and β are learnable parameters that adjust the local neighborhoods to a normal distribution, and $std_{\mathcal{N}_{x_i}}$ computes the standard deviation of the local neighborhoods \mathcal{N}_{x_i} with a small ϵ to ensure numerical stability. This operation facilitates stable geometric feature learning of graph neighborhoods across diverse geometric structures [39], accommodating the varying properties of polygons (Fig. 5). After the message-passing encoder layers, a global pooling layer aggregates the node features to generate a whole-graph embedding, which is then used for downstream polygon shape classification.

Following (3)-(5), we introduce a simple **Message-passing** neural network for **Polygon** geometries, named PolyMP, as shown in Fig. 1. PolyMP processes polygon point coordinates as input node features and the connectivity of polygon edges as input edges. The message-passing encoders of PolyMP compute the “message” for each node based on the graph adjacency and update the node features in each layer.

3.3 Densely self-connected message-passing

As noted in recent studies of graph message-passing networks [40], the basic message-passing mechanism restricts information flow due to the absence of self-loop information in graph representation learning. To address this limitation and enhance PolyMP’s learning capability, we adopt a densely self-connected (DSC) message-passing mechanism [18, 40, 41], defined as:

$$\mathcal{G}_l = f(\rho_\theta(readout(h_i|h_i \in \mathcal{H}_l))) + \mathcal{G}_{l-1} + \mathcal{G}_{l-2} + \dots + \mathcal{G}_0, \tag{6}$$

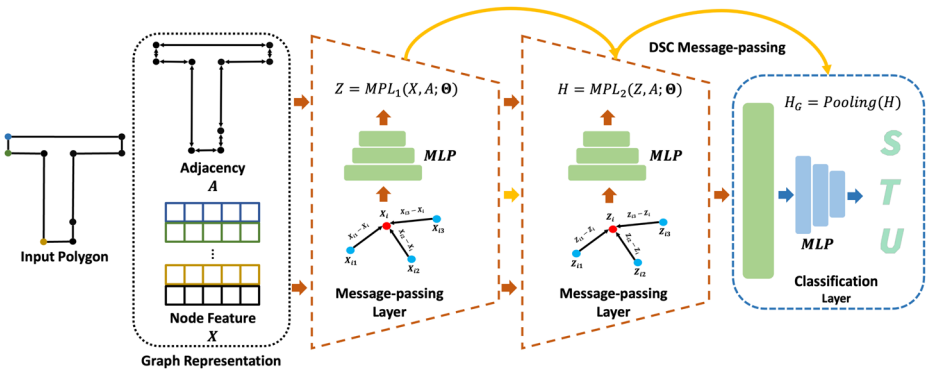


Fig. 1 Model architecture of PolyMP and PolyMP-DSC. PolyMP consists of a simple message-passing encoder followed by a classification layer (multi-layer perceptron). Yellow arrows indicate the densely self-connected message-passing in PolyMP-DSC

where we extend the basic PolyMP model to PolyMP-DSC, as shown in Fig. 1. The PolyMP-DSC aggregates latent graph-wise features \mathcal{G}_l at layer l by incorporating latent graph-wise features $\mathcal{G}_{l-1...0}$ from previous message-passing layers. This enables the model to capture hierarchical self-looped graph information, resulting in richer graph-wise representations.

3.4 Implementation

We evaluate the performance of learning models from previous studies, specifically: the VeerCNN model from Veer et al. [2], which uses 1D sequences converted from polygons as input; the DeepSet model from Zaheer et al. [9] and SetTransformer model from Lee et al. [12], which treat polygon vertices as input point clouds; the GCAE model from Yan et al. [6], which represents polygons as input graphs; and the DSC-NMP model from Fan et al. [40], a baseline graph learning model that processes polygons as input graphs and applies DSC neural message-passing for graph feature learning. We also evaluate the NUFTSpec model from Mai et al. [8], which encodes polygon geometries by converting them into feature vectors in the rasterized spectral domain. These models are compared against the proposed graph message-passing models for polygons, *PolyMP* and *PolyMP-DSC*.

The model configurations used in our experiments are summarized in Table 1, and the data representations of polygons and baseline learning models are illustrated in Fig. 2. For experimental control, every model trained and tested was constrained to a comparable model depth and latent space of feature embedding. During model training on the Glyph dataset and fine-tuning on the OSM dataset, we compute cross-entropy loss [42] to determine training gradients for fast convergence and good model generalizability. We use Adam optimization [43] with an initial learning rate of 0.01. For better model generalization, we adopt a learning rate decay strategy that reduces the learning rate by a factor of 10 once the model training reaches a performance plateau after 25 epochs. An early-stopping mechanism halts the training once the loss ceases to decrease for 50 epochs, mitigating overfitting. The batch size is set to 64, and the total training epochs are set to 100. Importantly, the models are not designed to individually outperform state-of-the-art models for each architecture, but rather to maintain comparability across architectures and enable the evaluation of the effects of data representations on results.

Table 1 Model configuration

Model	Conv. depth	Latent dim.	#Param.	Pooling
VeerCNN [2]	2	{2, 32, 64}	13,853	Global avg.
DeepSet [9]	2	{2, 64, 64}	11,837	Global avg.
SetTransformer [12]	2	{2, 64, 64}	113,531	Attention
GCAE [6]	2	{2, 64, 64}	7,613	Global avg.
NUFTSpec [8]	2	{288, 128, 64}	48,635	-
DSC-NMP [40]	2	{2, 64, 64}	20,539	Global add.
PolyMP (Ours)	2	{2, 64, 64}	11,837	Global max
PolyMP-DSC (Ours)	2	{2, 64, 64}	16,579	Global max

Each model is configured to have the same number of convolution layers and dimensions of latent embedding, to keep the model size (i.e., number of training parameters) comparable. Gray-highlighted rows correspond to the proposed methods: PolyMP and PolyMP-DSC

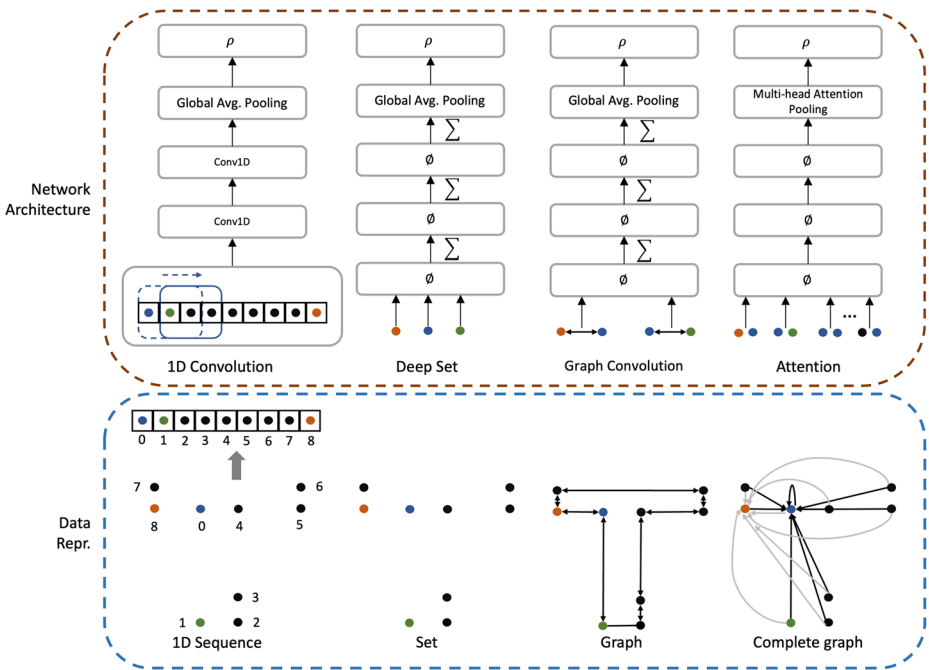


Fig. 2 Data representations of polygons (as 1D sequence, Point Set, Graph and Complete Graph) and the corresponding learning models (Baselines: VeerCNN, DeepSet, GCAE and SetTransformer)

4 Experiment

4.1 Glyph dataset

We introduce a synthetic dataset of highly variable geometric shapes to benchmark the classification performance of learning models on vector polygons. This dataset consists of 26 Latin alphabet character glyph geometries, representing semantic classes from *A* to *Z*, gathered from an online source [44]. A similar approach has been used previously in computational geometry to construct rich datasets for algorithm testing [45].

The synthetic dataset includes the boundaries (contour lines) of glyphs extracted from 1,413 sans serif and 1,002 serif fonts, resulting in 2D simple polygon geometries compliant with the standards of [38]. Serif and sans serif fonts are the two primary typographic families. Serif glyphs feature decorative strokes that enhance the legibility of body text, while sans serif glyphs have clean, minimal strokes, making them more suitable for headers. Importantly, these minor variations do not affect the overall Gestalt of the shapes, as readers can consistently recognize the symbols. This stability ensures that the labels assigned to the symbols are both stable and robust.

Each polygon geometry is encoded into a fixed-size feature matrix of size $\in \mathbb{R}^{n \times 3}$, where each feature vector $\in \mathbb{R}^3$ contains the 2D coordinates of vertices (x, y) as the geometric feature, and a binary feature (0, 1) to indicate the position of each 2D coordinate (whether it lies on the outer rings or holes of the polygons). Examples of glyph geometries are shown in Fig. 3.



Fig. 3 Glyph dataset samples. Left to right columns: A-shape to J-shape glyphs

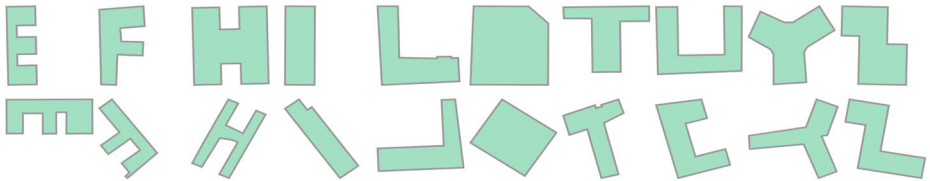


Fig. 4 OSM dataset samples. Top row: building geometries of standard shapes. Bottom row: geometric transformation counterparts. Left to right columns: E, F, H, I, L, O, T, U, Y and Z-shape buildings. [6]

4.2 OSM dataset

Latin alphabet glyph geometries share geometric similarities with building footprints. To test the generalizability of learning models on spatial objects, we use a building geometry dataset proposed by Yan et al. [6]. The dataset contains 10,000 real-world building footprints extracted from OpenStreetMap (OSM) [23], labeled into 10 categories based on template matching to letters [46]. The building footprints are randomly rotated and reflected from the original canonical samples, as shown in Fig. 4. This dataset is currently the most comprehensive real-world collection of vector geometries with purely shape-based labels (as opposed to labels based on building use, which should not be used for shape classification). However, we note that the assignment of labels in this dataset is somewhat subjective and may be less robust compared to the glyph-based dataset. Specifically, shape reflection can pose challenges for label stability, especially for asymmetric letters (e.g., *R*) or mirror image shapes of distinct labels (e.g., *S* and *Z*).

4.3 Pre-processing

Label-preserving transformations *To assess the invariance of geometric transformations in polygon representations and their effect on learning models, we apply label-preserving geometric transformations to the glyph polygons. We create four transformed training datasets, each consisting of 20%, 40%, 60%, and 80% of glyph polygons that have undergone these transformations. The details of the transformations are listed in Table 2, with the original and transformed data samples visualized in Fig. 5. This results in five training datasets: 0%, 20%, 40%, 60%, and 80% transformations, which will be used for the experiment. The test set consists of data randomly sampled from these five datasets.*

Polygon Simplifications *To assess the invariance of polygon representations to trivial vertices, we apply the Douglas-Peucker algorithm [47] with a tolerance of 1.0 to the original glyph polygons (typically sized around 50×50 units). This simplification process generates polygon samples with fewer vertices, particularly by excluding co-linear or nearly co-linear*

Table 2 Label-preserving transformations applied to samples of Glyph dataset maintains the semantic information (label) of original polygons (i.e., 180 degrees rotation of letter M alter its semantic label, converting to letter W)

Transformation	Operation
Rotation	Rotate data around its centroid by a random angle $\in [-75^\circ, 75^\circ]$.
Scaling	Scale data by random distinct factors $\in [0.1, 2]$ on the x and y axes.
Shearing	Shear data in a random angle $\in [-45^\circ, 45^\circ]$ on the x and y axes.

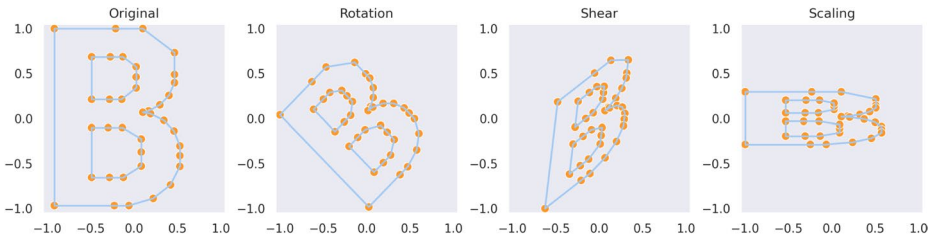


Fig. 5 Data samples of Glyph dataset under label-preserving transformations

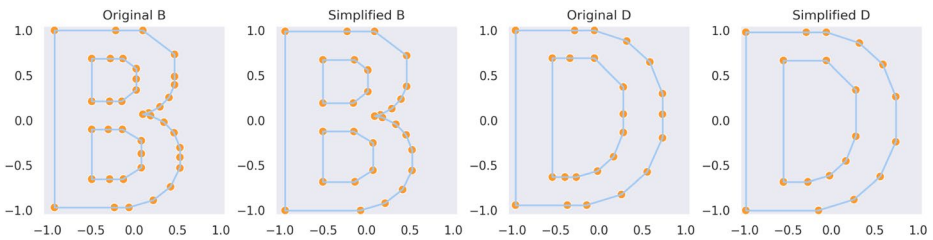


Fig. 6 Visualisation of original and simplified polygon examples generated with Douglas-Peucker algorithm [47]. The simplified polygons, compared to original polygons, contain a subset of the vertices that defined the original curve

vertices. Importantly, the simplification is applied before normalizing the polygons to the range of $(-1, 1)$ and before any augmentation, ensuring that the simplified geometries preserve the topology and essential shape of the original glyph (see Fig. 6).

5 Result

We report model performances for all experiments as the overall accuracy (O.A.), calculated as the ratio of true positives over the total number of samples.

5.1 Results on glyph dataset

Tables 3 and 4 present the model performances on the Glyph dataset and simplified polygons, respectively.

At the 0% transformation ratio in Table 3, we observe that PolyMP achieves 99.68%, 39.91%, and 82.35% accuracy on the original, rotated, and scaled test samples, respectively,

while VeerCNN achieves 59.07% accuracy on sheared samples. In comparison, DeepSet, SetTransformer, and GCAE maintain comparable performance on the original and scaled test samples (94.62%, 65.01%; 96.25%, 67.91%; and 95.16%, 64.54%, respectively). However, models that learn from set (DeepSet and SetTransformer) and graph inputs (GCAE and DSC-NMP) experience significant performance deterioration on rotated and sheared samples, with accuracy of 23.67%, 26.00%; 25.23%, 27.79%; 31.38%, 38.84%; and 35.71%, 43.21%, respectively. Overall, PolyMP-DSC, which learns densely connected graph features, achieves the highest classification accuracy at 71.21%, significantly outperforming the baseline DSC-NMP model (62.24% O.A.).

At the 20% transformation ratio, both PolyMP (84.89% (+44.98%), 93.26% (+10.91%), and 88.17% (+34.95%)) and PolyMP-DSC (86.20% (+44.46%), 94.60% (+8.15%), and 90.65% (+34.49%)) show significant improvements in test accuracy on rotated, scaled, and sheared samples, respectively. In terms of overall accuracy, PolyMP achieves 91.30% O.A. (+22.27%), outperforming VeerCNN's 81.05% O.A. PolyMP-DSC records the highest O.A. at 92.64% (+21.43%). In contrast, the baseline learning models are unable to match the performance of PolyMP and PolyMP-DSC, despite the increase in training set diversity.

At the 40% and 60% transformation ratios, we observe that the accuracy improvements from increasing the proportion of geometrically transformed samples in the training set have plateaued. In this case, PolyMP achieves 93.11% and 93.69% O.A., while PolyMP-DSC leads with 94.18% and 94.91% O.A. Both models maintain their superior performance compared to others. VeerCNN records 83.30% and 84.88% O.A., followed by DeepSet (80.12% and 81.49% O.A.), SetTransformer (80.33% and 81.94% O.A.), GCAE (72.72% and 75.50% O.A.), DSC-NMP (57.97% and 67.73% O.A.), and NUFTSpec (70.74% and 78.71% O.A.), all of which fall behind.

At the 80% transformation ratio, PolyMP and PolyMP-DSC achieve the highest test performances on rotated and sheared samples, with accuracy of 91.64%, 92.36%, and 93.19%, 94.26%, respectively. These empirical results suggest that, compared to sequence encoding of polygons using CNNs and set representations with set-based learning models, graph representations combined with message-passing neural networks (PolyMP and PolyMP-DSC) are more robust to permutations of polygon vertices. This robustness is particularly evident in geometric transformations such as rotation and shearing, where the feed-in order of vertices can vary. Comparing basic PolyMP with the extended PolyMP-DSC, we observe testing performance improvements of up to 2% O.A. on the Glyph dataset. These results highlight the effectiveness of incorporating hierarchical self-looped graph representations via densely self-connected message passing for polygon encoding and shape-based classification.

From a different perspective, as the proportion of original samples progressively decreases in the training set (from 0% to 80% transformation ratios), PolyMP and PolyMP-DSC experience only a slight test performance deterioration, with a decrease from 99.68% to 98.48% (-1.20%) and 99.58% to 98.93% (-0.65%), respectively. In contrast, VeerCNN (-4.13%), DeepSet (-3.96%), SetTransformer (-5.08%), GCAE (-5.58%), NUFTSpec (-2.27%), and DSC-NMP (-2.50%) exhibit more significant performance drops in test accuracy on the original samples.

In Table 4, we present the test performances of models trained and evaluated on simplified polygons from the Glyph dataset. This experimental setting focuses on evaluating the models' performance on polygons with a reduced number of trivial vertices (Fig. 6) and compares these results with those shown in Table 3.

Table 3 Test performances on Glyph dataset

Trans. Ratio	Model	Glyph-O Acc.	Glyph-R Acc.	Glyph-SC Acc.	Glyph-SH Acc.	Glyph O.A.
0%	VeerCNN	97.64	41.51	79.15	59.07	69.43
	DeepSet	94.62	23.67	65.01	26.00	52.56
	SetTransformer	96.25	25.23	67.91	27.79	54.42
	GCAE	95.16	31.38	64.54	38.84	57.83
	NUFTSpec	98.42	34.90	72.07	39.68	61.34
	DSC-NMP	96.81	35.71	72.83	43.21	62.24
	PolyMP	99.68	39.91	82.35	53.22	69.03
PolyMP-DSC	99.58	41.74	86.45	56.16	71.21	
20%	VeerCNN	95.12	67.63	83.53	77.78	81.05
	DeepSet	93.88	67.95	77.31	64.72	76.25
	SetTransformer	93.87	71.72	82.43	72.56	80.18
	GCAE	93.31	47.31	71.47	57.11	67.61
	NUFTSpec	98.00	65.27	84.03	70.74	79.61
	DSC-NMP	93.68	52.38	78.86	61.72	71.73
	PolyMP	99.10	84.89	93.26	88.17	91.30
PolyMP-DSC	99.19	86.20	94.60	90.65	92.64	
40%	VeerCNN	94.75	73.26	84.77	80.29	83.30
	DeepSet	92.57	74.24	79.24	72.08	80.12
	SetTransformer	91.97	72.73	83.25	73.25	80.33
	GCAE	91.70	59.55	74.35	64.42	72.72
	NUFTSpec	97.44	78.84	86.00	78.71	85.24
	DSC-NMP	86.71	54.01	71.96	57.97	67.71
	PolyMP	98.85	88.53	94.35	90.74	93.11
PolyMP-DSC	99.05	90.18	95.33	92.57	94.18	
60%	VeerCNN	94.50	77.16	85.38	82.36	84.88
	DeepSet	91.72	76.89	81.46	73.97	81.49
	SetTransformer	90.93	77.04	82.39	77.31	81.94
	GCAE	90.46	65.02	76.38	68.78	75.50
	NUFTSpec	96.80	83.13	88.40	81.76	87.55
	DSC-NMP	90.97	64.17	80.00	67.73	75.76
	PolyMP	98.59	90.08	94.11	91.65	93.69
PolyMP-DSC	98.79	91.80	95.76	93.20	94.91	
80%	VeerCNN	93.51	78.97	85.25	83.17	85.25
	DeepSet	90.66	78.68	82.14	76.55	82.47
	SetTransformer	91.17	79.94	83.69	79.78	83.67
	GCAE	89.58	67.95	77.99	70.55	76.83
	NUFTSpec	96.15	84.40	89.24	83.46	88.32
	DSC-NMP	94.31	74.31	86.75	78.36	83.46
	PolyMP	98.48	91.64	94.69	92.36	94.31
PolyMP-DSC	98.93	93.19	95.95	94.26	95.61	

Transformation ratios 0%, 20%, 40%, 60% and 80% refer to training datasets consisting of 0%, 20%, 40%, 60% and 80% geometric transformed samples. Glyph-O, Glyph-R, Glyph-SC and Glyph-SH Acc. indicate the classification accuracy of models tested on original, rotated, scaled and sheared data. Gray-highlighted rows correspond to the proposed methods: PolyMP and PolyMP-DSC. Overall accuracy is reported as Glyph O.A. Best performance is highlighted in bold

Table 4 Test performances on Glyph dataset, consisting of simplified polygons generated by Douglas-Peucker algorithm (tolerant=1.0) with the removal of trivial vertices. Gray-highlighted rows correspond to the proposed methods: PolyMP and PolyMP-DSC. Best performance is highlighted in bold

Trans. Ratio	Model	Glyph-O Acc.	Glyph-R Acc.	Glyph-SC Acc.	Glyph-SH Acc.	Glyph O.A.
0%	VeerCNN	94.72	39.65	75.42	56.12	66.56
	DeepSet	94.01	23.75	65.11	25.53	52.30
	SetTransformer	94.57	23.67	65.72	26.61	52.76
	GCAE	94.83	31.64	64.21	39.90	58.08
	NUFTSpec	98.03	34.91	71.96	40.07	61.30
	DSC-NMP	95.40	35.21	71.61	41.56	61.04
	PolyMP	99.27	40.52	81.90	53.50	68.92
	PolyMP-DSC	99.36	42.26	86.16	56.68	71.07
20%	VeerCNN	90.55	62.58	77.80	72.31	75.85
	DeepSet	93.81	68.26	77.83	64.93	76.47
	SetTransformer	92.31	70.51	80.54	70.65	78.54
	GCAE	92.26	47.29	71.28	57.33	67.39
	NUFTSpec	97.76	65.10	83.97	71.41	79.65
	DSC-NMP	92.20	51.73	77.60	60.44	70.55
	PolyMP	98.37	83.37	91.41	87.25	89.74
	PolyMP-DSC	98.93	84.57	93.56	89.15	91.19
40%	VeerCNN	89.63	67.82	78.72	74.61	77.73
	DeepSet	92.66	74.81	79.60	71.62	80.46
	SetTransformer	89.86	70.91	80.70	70.97	78.14
	GCAE	91.65	59.30	74.95	64.79	72.94
	NUFTSpec	97.23	78.85	86.09	78.95	85.25
	DSC-NMP	85.08	52.92	69.90	56.73	66.21
	PolyMP	98.10	86.47	92.72	88.73	91.09
	PolyMP-DSC	98.66	89.04	94.45	91.84	93.28
60%	VeerCNN	89.29	71.91	79.42	77.29	79.50
	DeepSet	91.58	77.44	81.62	73.84	81.57
	SetTransformer	89.70	75.03	80.26	75.53	80.15
	GCAE	90.06	63.61	76.64	68.05	74.79
	NUFTSpec	96.62	82.49	88.48	82.16	87.46
	DSC-NMP	90.02	62.99	79.47	66.67	74.83
	PolyMP	97.74	89.50	92.99	90.59	92.48
	PolyMP-DSC	98.38	91.53	94.89	92.76	94.13
80%	VeerCNN	88.05	73.89	79.55	77.17	79.69
	DeepSet	91.04	79.21	82.44	76.45	82.67
	SetTransformer	89.99	79.14	82.70	78.60	82.63
	GCAE	88.22	66.55	78.01	70.44	76.06
	NUFTSpec	96.10	84.56	89.16	83.28	88.27
	DSC-NMP	93.06	73.10	85.03	76.99	82.08
	PolyMP	97.42	90.55	93.25	91.36	92.97
	PolyMP-DSC	98.47	92.78	95.10	93.43	94.77

We observe similar performance trends in the models when evaluated on the simplified polygons. Notably, VeerCNN experiences a significant drop in performance compared to the results in Table 3. Specifically, at the 0% transformation ratio, VeerCNN’s test performance decreases from 69.43% O.A. (Table 3) to 66.56% O.A. (Table 4), representing a 2.87% decline. As data diversity increases, the performance deterioration in VeerCNN becomes

more pronounced, with a drop of 5.2%, progressively increasing to 5.56% O.A. at transformation ratios from 20% to 80%.

The performance deterioration of VeerCNN strongly suggests that CNN models are not robust to changes in the number of vertices in polygons. This can be attributed to several factors, including the data encoding method used in VeerCNN, which represents polygons as fixed-length 1D sequences with zero padding. Simplifying polygons by removing trivial vertices alters the length of these sequences, which not only dilutes the geometric information through zero padding but also disrupts the explicit encoding of vertex adjacency within the 1D sequences.

In comparison, we observe only minor performance drops in general set-based and graph-based learning models. This can be attributed to the flexibility of set and graph representations of polygons, which allow these models to accept input polygons of varying lengths. Specifically, graph-based message-passing networks (i.e., PolyMP and PolyMP-DSC) effectively learn local geometric features from neighboring nodes using a message-passing mechanism, followed by permutation-invariant pooling operations (e.g., max-pooling), as shown in (4). A similar pooling operation is utilized in set-based learning models [9, 12].

5.2 Results on OSM dataset

The results in Table 5 show the test accuracy of fine-tuned classifiers on real-world building footprints from the OSM dataset, using feature encoders pre-trained on the Glyph dataset (80% subset). These fine-tuned models were evaluated on building footprints with both normalized and original coordinates.

PolyMP achieves an accuracy of 88.58% for building footprints with normalized coordinates and 46.20% for footprints with original coordinates. PolyMP-DSC demonstrates slightly lower test accuracy, with 87.20% for normalized coordinates and 40.68% for original coordinates, but still outperforms other baseline methods, except for DeepSet. This can be attributed to the ability of both models to compute local geometric features via message-passing networks, reducing the impact of absolute positioning while preserving polygon geometry. DeepSet performs best, with 88.62% O.A. on normalized coordinates and 82.31% O.A. on rotated and reflected buildings, making it the top performer in some cases.

The NUFTSpec model, which transforms polygon geometries into consistent spectral feature vectors via Non-Uniform Fourier Transforms, demonstrates stable performance at approximately 74% O.A. across both normalized and original coordinates. This stability stems from a series of affine transformations (e.g., scaling and translation) applied to input polygons during feature transformation, ensuring that each polygon is projected into a unit space and positioned within the same relative coordinate system, as discussed in [8].

In Fig. 7, classification results of PolyMP on building footprints from the two OSM subsets (OSM-O and OSM-R) are visualized. The results show PolyMP's robustness in classifying building footprints of varying sizes and geometric shapes, leveraging the geometric invariant features learned through message-passing networks.

These findings underline the effectiveness of graph message-passing models, particularly PolyMP and PolyMP-DSC, in handling both normalized and original coordinate systems and achieving strong performance across various data transformations.

Table 5 Fine-tuned test performance on the OSM dataset. OSM-O and OSM-R indicate the classification accuracy of models tested on original and rotated & reflected samples, respectively. Gray-highlighted rows correspond to the proposed methods: PolyMP and PolyMP-DSC. Overall accuracy is reported as OSM O.A. Best performance is highlighted in bold

Model	OSM-O Acc.	OSM-R Acc.	OSM O.A.	
VeerCNN	86.30	72.44	79.36	
DeepSet	94.79	82.31	88.62	
SetTransformer	89.10	68.73	78.90	
GCAE	95.23	78.15	86.94	Normalised coordinates $\in (-1, 1)$
NUFTSpec	88.02	60.66	74.26	
DSC-NMP	87.58	66.73	77.14	
PolyMP	96.59	81.03	88.58	
PolyMP-DSC	95.75	78.63	87.20	
VeerCNN	42.23	42.09	42.16	
DeepSet	22.00	21.65	21.84	
SetTransformer	35.62	35.30	35.46	
GCAE	22.12	22.32	22.22	Original coordinates
NUFTSpec	88.02	61.38	74.42	
DSC-NMP	31.81	33.23	32.52	
PolyMP	50.64	39.86	46.20	
PolyMP-DSC	44.19	36.78	40.68	

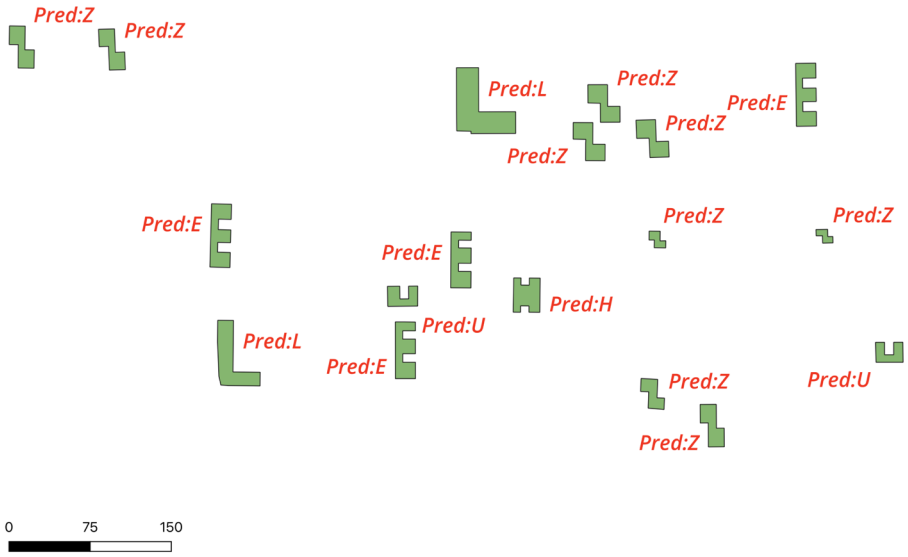
6 Discussion

6.1 Model expressivity

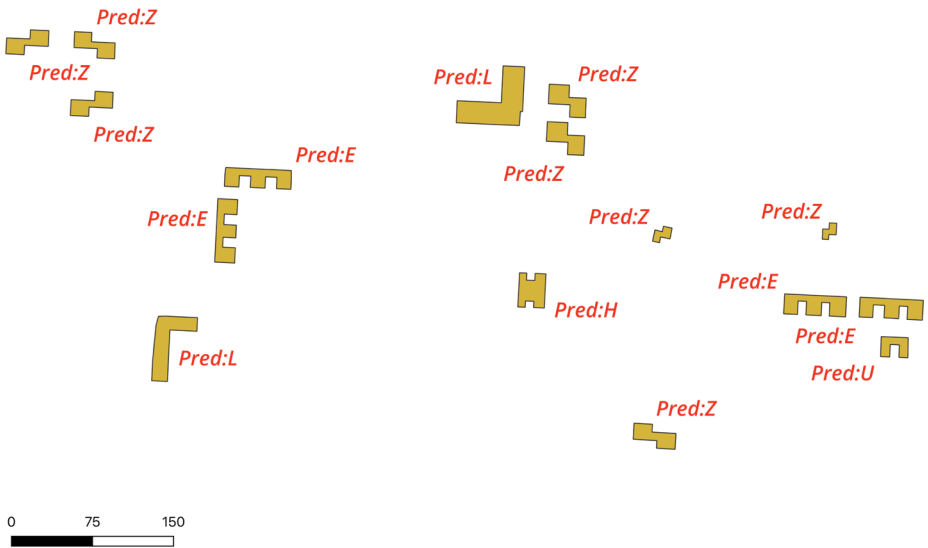
To interpret the expressivity of different polygon encoding and learning models, we visualize the feature maps of models tested on original and geometrically transformed samples from the Glyph dataset in Fig. 8.

Figures 8a and 8b illustrate the salient feature maps and class predictions of set-based (DeepSet, Set Transformer) and graph-based (GCAE, DSC-NMP, PolyMP, and PolyMP-DSC) learning models on geometrically transformed samples. Notably, PolyMP and PolyMP-DSC correctly predict the class label of the sheared and rotated glyph *M*, whereas DeepSet, Set Transformer, GCAE, and DSC-NMP make incorrect predictions. By comparing the feature maps in Figs. 8c and 8d, we observe that PolyMP and PolyMP-DSC effectively capture the global structure or “skeleton” of polygons by identifying salient, non-trivial vertices along the boundary. The message-passing encoder, coupled with a local pooling function (i.e., max-pooling), extracts geometric features from local neighborhoods, where non-trivial vertices (darker color) are distinguished within clusters of trivial vertices (lighter color). This enables the models to generate robust embeddings of polygons, even under geometric transformations. In contrast, set-based models and GCAE learn sparser geometric representations, capturing regional features in node clusters while neglecting relational information between more distant nodes. This limitation arises from their reliance on set representations, which inherently discard connectivity information between vertices.

Figures 8e and 8f illustrate the feature maps for polygons with holes. DeepSet and Set Transformer primarily capture geometric features from non-trivial vertices along the exte-

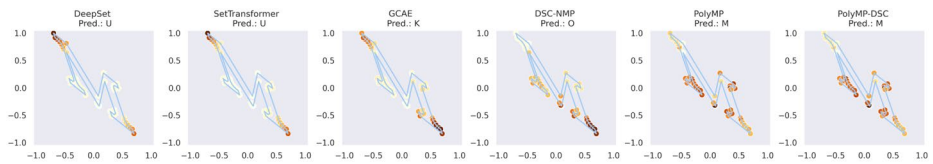


(a) OSM-O

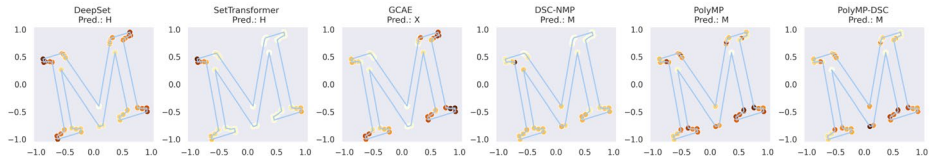


(b) OSM-R

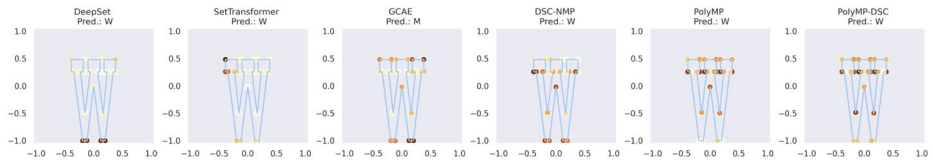
Fig. 7 Shape classification of building footprints using PolyMP on a selected region of the OSM dataset [6]. Model predictions are highlighted in red



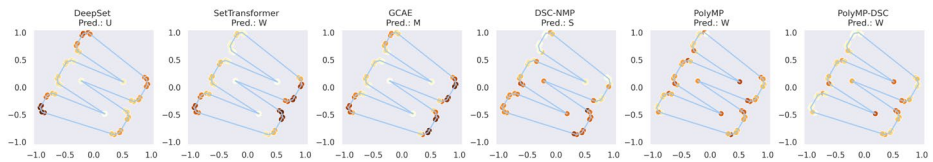
(a) Sheared M glyph.



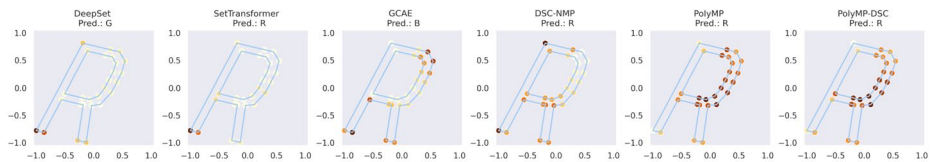
(b) Rotated M glyph.



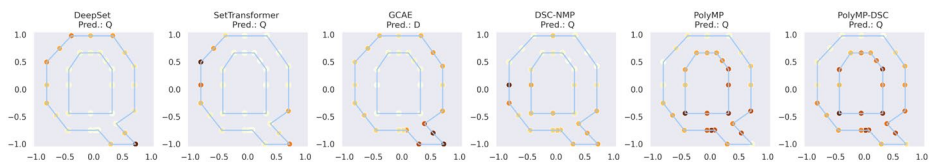
(c) Scaled W glyph.



(d) Rotated W glyph.



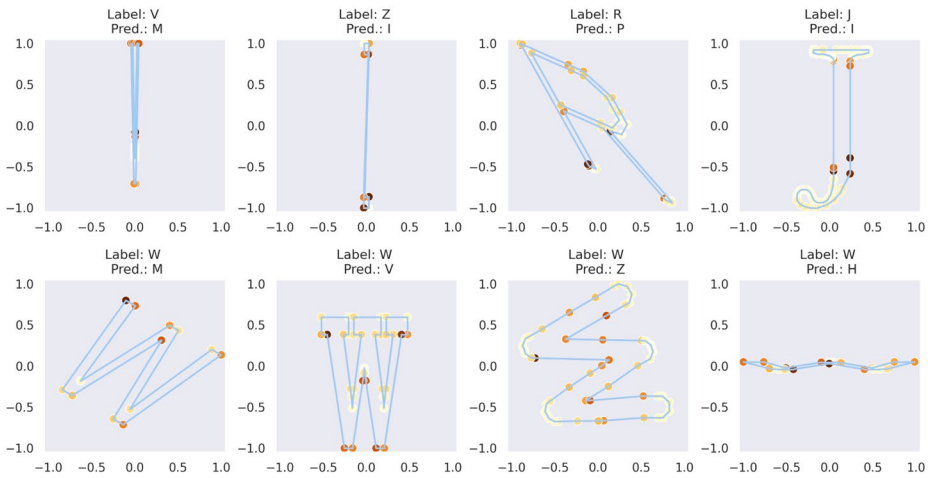
(e) Sheared R glyph.



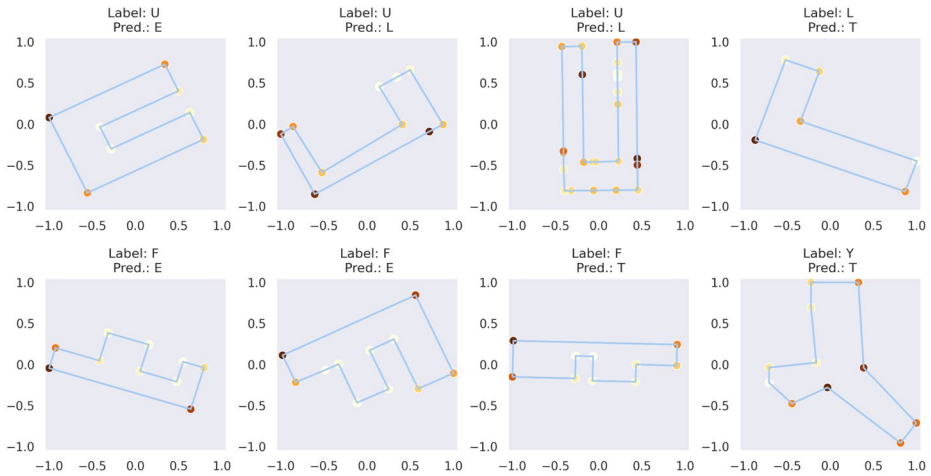
(f) Original Q glyph.

Fig. 8 Feature maps to visualize geometric features learned by individual models to classify input geometries to output categories. The darker the color, the more salient are the features learned. Feature values are normalised $\in [0, 1]$

rior boundary, while largely ignoring those within the interior. Although set-based models can encode polygons of varying lengths and produce permutation-invariant feature representations, their lack of connectivity modeling restricts their ability to learn geometric structures in polygons with holes. Similarly, the graph convolution model GCAE struggles to flexibly capture local geometric features, as discussed in previous sections. In contrast, PolyMP and PolyMP-DSC effectively capture geometric features from both the exterior and interior boundaries of polygons. This is achieved through graph message-passing encoders, where geometric features (i.e., the “message”) are exchanged only between directly connected nodes within the same connected component of a graph—defined in this case by



(a) Glyph dataset.



(b) OSM dataset.

Fig. 9 Failure cases of PolyMP on Glyph and OSM datasets

the polygon's linear rings. As a result, PolyMP-based models can better preserve structural relationships in polygons with complex topologies.

6.2 Limitations

Despite its strong performance, our proposed method still encounters misclassification limitations. Figure 9 highlights instances where PolyMP fails to classify samples correctly in the Glyph and OSM datasets.

On synthetic data, misclassifications predominantly occur in cases where excessive geometric transformations—such as heavy shearing and anisotropic scaling—severely distort the shape's structure, rendering it unrecognizable: rotated W shapes (Fig. 9a) are sometimes mistaken for M or even Z, while heavily sheared W shapes may be misclassified as H or V.

Similarly, in the OSM dataset, PolyMP and PolyMP-DSC occasionally misclassify building polygons with similar geometric structures, such as E, L, and U shapes (Fig. 9b). This can be attributed to the model's reliance on local geometric features, which may not sufficiently distinguish between structures with high shape similarity.

Considering the characteristics of the two datasets (Sec. 4.1 and 4.2), PolyMP and PolyMP-DSC trained on the Glyph dataset primarily learns locally salient geometric features through message-passing encoders. These features are particularly effective for serif glyphs, which contain distinctive decorative stroke endings. However, when excessive geometric transformations distort these key features, the model struggles to preserve label consistency, leading to misclassification among shapes with similar structures.

7 Conclusion

In this study, we investigated the challenge of geometric-invariant shape classification from 2D vector polygons, a common but underexplored data format in spatial analysis. Through comprehensive empirical evaluation, we demonstrated that combining discrete, non-grid-like polygon representations with graph-based message-passing neural networks—PolyMP and its densely self-connected variant, PolyMP-DSC—enables the learning of robust and expressive geometric features.

Our findings show that graph representations of polygons, which encode both geometric structure and vertex connectivity, significantly outperform sequence- and set-based encodings in terms of transformation invariance and generalization. In particular, PolyMP-DSC enhances feature propagation by incorporating dense self-connections, resulting in improved classification performance and robustness to structural perturbations such as trivial vertices. These properties are crucial for geospatial tasks where object shapes may undergo rotation, scaling, shearing, or digitization noise.

Importantly, our approach aligns well with practical geospatial workflows: polygon-to-graph conversion is natively supported by many GIS libraries, enabling seamless adoption of our models in downstream applications such as automated cartographic generalization, building footprint recognition, and road geometry analysis.

Nevertheless, current limitations remain. Both PolyMP and PolyMP-DSC occasionally struggle with locally distinctive features that vary across shape classes or domains. To address this, future work will explore the integration of complementary geometric-invariant

descriptors—such as spectral features (e.g., NUFT) and local shape attributes (e.g., turning angles, curvature, or radii of arc segments)—to further improve model robustness and invariance.

Overall, this work highlights the importance of representation choice in spatial deep learning. By designing models with inductive biases that reflect geometric structure and transformation invariance, we move closer to human-like perception of shape in geospatial domains. Beyond classification, our findings open promising directions for extending graph-based learning to polygon regression, spatial clustering, and topological inference in large-scale vector datasets.

Author Contributions Z.H. wrote the main manuscript and prepared all figures. K.K and M.T edited the main manuscript. All authors reviewed the manuscript.

Funding Open Access funding enabled and organized by CAUL and its Member Institutions. No funding was obtained for this study.

Data Availability The data and code supporting the findings of this study are available at <https://github.com/zexhuang/PolyMP>.

Declarations

Conflicts of Interest This paper has been approved by all co-authors. The authors have no competing interests to declare that are relevant to the content of this article.

Competing interests The authors declare no competing interests.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Yan X, Ai T, Yang M, Yin H (2019) A graph convolutional neural network for classification of building patterns using spatial vector data. *ISPRS J Photogramm Remote Sens* 150:259–273
2. Veer R, Bloem P, Folmer E (2018) Deep learning for classification tasks on geospatial vector polygons. arXiv preprint [arXiv:1806.03857](https://arxiv.org/abs/1806.03857)
3. Andrášik R, Bíl M (2016) Efficient road geometry identification from digital vector data. *J Geogr Syst* 18(3):249–264
4. Lehar S (2003) *The World in Your Head: A Gestalt View of the Mechanism of Conscious Experience*. Lawrence Erlbaum, Mahwah, N.J
5. Lehar S (2003) Gestalt isomorphism and the primacy of subjective conscious experience: A gestalt bubble model. *Behavi Brain Sci* 26(4):375–408
6. Yan X, Ai T, Yang M, Tong X (2021) Graph convolutional autoencoder model for the shape coding and cognition of buildings in maps. *Int J Geogr Inf Sci* 35(3):490–512
7. Yan X, Ai T, Yang M, Tong X, Liu Q (2022) A graph deep learning approach for urban building grouping. *Geocarto Int* 37(10):2944–2966
8. Mai G, Jiang C, Sun W, Zhu R, Xuan Y, Cai L, Janowicz K, Ermon S, Lao N (2023) Towards general-purpose representation learning of polygonal geometries. *GeoInformatica* 27(2):289–340

9. Zaheer M, Kottur S, Ravanbakhsh S, Póczos B, Salakhutdinov RR, Smola AJ (2017) Deep sets. *Adv Neural Inf Process Syst* 30
10. Qi CR, Su H, Mo K, Guibas LJ (2017) Pointnet: Deep learning on point sets for 3d classification and segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 652–660
11. Qi CR, Yi L, Su H, Guibas LJ (2017) Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Adv Neural Inf Process Syst* 30
12. Lee J, Lee Y, Kim J, Kosiorek A, Choi S, Teh YW (2019) Set transformer: A framework for attention-based permutation-invariant neural networks. In: *International conference on machine learning*, PMLR, pp 3744–3753
13. Bronstein MM, Bruna J, Cohen T, Velicković P (2021) Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*
14. He X, Zhang X, Xin Q (2018) Recognition of building group patterns in topographic maps based on graph partitioning and random forest. *ISPRS J Photogramm Remote Sens* 136:26–40
15. Bei W, Guo M, Huang Y (2019) A spatial adaptive algorithm framework for building pattern recognition using graph convolutional networks. *Sens* 19(24):5518
16. Kipf TN, Welling M (2017) Semi-supervised classification with graph convolutional networks. In: *International conference on learning representations (ICLR)*
17. Gilmer J, Schoenholz SS, Riley PF, Vinyals O, Dahl GE (2017) Neural message passing for quantum chemistry. In: *International conference on machine learning*, PMLR, pp 1263–1272
18. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 770–778
19. Wu Q, Diao W, Dou F, Sun X, Zheng X, Fu K, Zhao F (2016) Shape-based object extraction in high-resolution remote-sensing images using deep boltzmann machine. *Int J Remote Sens* 37(24):6012–6022
20. Microsoft (2018) Microsoft/USBUILDINGFOOTPRINTS: Computer generated building footprints for the United States. <https://github.com/microsoft/USBuildingFootprints>
21. Lin G, Milan A, Shen C, Reid I (2017) Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 1925–1934
22. Xu Y, Chen Z, Xie Z, Wu L (2017) Quality assessment of building footprint data using a deep autoencoder network. *Int J Geogr Inf Sci* 31(10):1929–1951
23. (2017) OpenStreetMap contributors: Planet dump retrieved from <https://planet.osm.org>. <https://www.openstreetmap.org>
24. Lopes RG, Ha D, Eck D, Shlens J (2019) A learned representation for scalable vector graphics. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp 7930–7939
25. Mino A, Spanakis G (2018) Logan: Generating logos with a generative adversarial neural network conditioned on color. In: *2018 17th IEEE international conference on machine learning and applications (ICMLA)*, IEEE, pp 965–970
26. Sage A, Agustsson E, Timofte R, Van Gool L (2018) Logo synthesis and manipulation with clustered generative adversarial networks. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 5879–5888
27. Radford A, Metz L, Chintala S (2016) Unsupervised representation learning with deep convolutional generative adversarial networks. In: Bengio Y, LeCun Y (eds) *4th international conference on learning representations, ICLR 2016, San Juan, Puerto Rico, May 2–4, 2016, conference track proceedings*. [arXiv:1511.06434](https://arxiv.org/abs/1511.06434)
28. Odena A, Olah C, Shlens J (2017) Conditional image synthesis with auxiliary classifier gans. In: *International conference on machine learning*, PMLR, pp 2642–2651
29. Ha D, Eck D (2018) A neural representation of sketch drawings. In: *International conference on learning representations*
30. Carlier A, Danelljan M, Alahi A, Timofte R (2020) Deepsvg: A hierarchical generative network for vector graphics animation. *Adv Neural Inf Process Syst* 33:16351–16361
31. Bruna J, Zaremba W, Szlam A, LeCun Y (2014) Spectral networks and locally connected networks on graphs. In: *ICLR*
32. Defferrard M, Bresson X, Vandergheynst P (2016) Convolutional neural networks on graphs with fast localized spectral filtering. *Adv Neural Inf Process Syst* 29
33. Velickovic P, Cucurull G, Casanova A, Romero A, Lio P, Bengio Y et al (2017) Graph attention networks. *Stat* 1050(20):10–48550
34. Huang Z, Khoshelham K, Tomko M (2024) Contrastive graph autoencoder for shape-based polygon retrieval from large geometry datasets. *Trans Mach Learn Res*
35. Liu C, Hu Y, Li Z, Xu J, Han Z, Guo J (2021) Triangleconv: A deep point convolutional network for recognizing building shapes in map space. *ISPRS Int J Geo Inf* 10(10):687

36. Wang Y, Sun Y, Liu Z, Sarma SE, Bronstein MM, Solomon JM (2019) Dynamic graph cnn for learning on point clouds. *Acm Trans Graphics (tog)* 38(5):1–12
37. Battaglia PW, Hamrick JB, Bapst V, Sanchez-Gonzalez A, Zambaldi V, Malinowski M, Tacchetti A, Raposo D, Santoro A, Faulkner R et al (2018) Relational inductive biases, deep learning, and graph networks. arXiv preprint [arXiv:1806.01261](https://arxiv.org/abs/1806.01261)
38. (2003) OpenGeospatialConsortium: OpenGIS simple features specification For SQL. OpenGeospatial-Consortium. <http://www.opengis.org/>
39. Ma X, Qin C, You H, Ran H, Fu Y (2022) Rethinking network design and local geometry in point cloud: A simple residual mlp framework. In: International conference on learning representations
40. Fan X, Gong M, Wu Y, Qin AK, Xie Y (2021) Propagation enhanced neural message passing for graph representation learning. *IEEE Trans Knowl Data Eng* 35(2):1952–1964
41. Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4700–4708
42. Zhang Z, Sabuncu M (2018) Generalized cross entropy loss for training deep neural networks with noisy labels. *Adv Neural Inf Process Syst* 31
43. Kingma DP, Ba J (2015) Adam: A method for stochastic optimization. In: ICLR (Poster)
44. Google (2010) Google. <https://fonts.google.com/>
45. Duckham M, Kulik L, Worboys M, Galton A (2008) Efficient generation of simple polygons for characterizing the shape of a set of points in the plane. *Pattern Recogn* 41(10):3224–3236
46. Yan X, Ai T, Zhang X (2017) Template matching and simplification method for building features based on shape cognition. *ISPRS Int J Geo Inf* 6(8):250
47. Douglas DH, Peucker TK (1973) Algorithms for the reduction of the number of points required to represent a digitized line or its caricature. *Cartographica Int J Geogr Inf Geovisualization* 10(2):112–122

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.