International Journal of Geo-Information

MDPI

Article

A Multi-Scale Hybrid Scene Geometric Similarity Measurement Method Using Heterogeneous Graph Neural Network

Chongya Gong [1], Tinghua Ai [1,*], Shiyu Chen [2], Tianyuan Xiao [1] and Huafei Yu [1]

[1] School of Resource and Environmental Sciences, Wuhan University, Wuhan 430079, China; chongyagong@gmail.com (C.G.); xiaotianyuan@whu.edu.cn (T.X.); huafeiyu@whu.edu.cn (H.Y.)
[2] School of Geographical Sciences, Xinyang Normal University, Xinyang 464300, China; csy_hy@xynu.edu.cn
* Correspondence: tinghuaai@whu.edu.cn

Abstract: Geographic features in maps consist of a mixture of points, polylines, and polygons, generally including POIs, roads, buildings, and other geographic features. Due to the differing dimensionality of these various types of geographic data, traditional geometric similarity measurement methods that rely on a single type of feature are not applicable to mixed scenes. The traditional solution to this issue is to treat points as projections of polylines and polylines as projections of polygons. Through neural networks, projection matrices can be learned to convert points, polylines, and polygons into the same type of object, thereby enabling the use of single-scene geometric measurement methods (e.g., Graph Neural Networks) to solve the problem. However, the key challenge in using Graph Neural Networks for similarity measurement is learning the adjacency relationships between geometric features. It is evident that the adjacency relationships between different feature pairs, such as polyline–polygon, polyline–polyline, and polygon–polygon, require different approaches for measurement, and these diverse relationships cannot be captured by a simple GNN. Heterogeneous Graph Neural Networks (HGNNs) are suited to address this problem: different adjacency relationships between feature pairs can be learned using distinct embedded networks, the new node characteristics can be calculated through the information aggregation and propagation framework of HGNNs, and these new characteristics can be used for geometric similarity measurement. Finally, the effectiveness of the proposed method was verified through practical experiments.

Keywords: heterogeneous graph neural network; hybrid scene; similarity measurement; data-driven

Academic Editors: Wolfgang Kainz and Maria Antonia Brovelli

Received: 7 January 2025
Revised: 10 February 2025
Accepted: 12 February 2025
Published: 14 February 2025

Citation: Gong, C.; Ai, T.; Chen, S.; Xiao, T.; Yu, H. A Multi-Scale Hybrid Scene Geometric Similarity Measurement Method Using Heterogeneous Graph Neural Network. *ISPRS Int. J. Geo-Inf.* **2025**, *14*, 84. https://doi.org/10.3390/ijgi14020084

1. Introduction

Measuring the similarity between geographic objects is one of the central research topics in the geographical information science field [1–4]. For instance, in computer vision [5,6], image retrieval [7] and matching [8] are concerned with the similarity of image features [9]; in engineering technology, optimal transport problems [10] focus on the similarity of probability distributions; in bioengineering, phenotype similarity issues are related to genetic similarities [11]. Similarly, in geographic information science, the measurement of similarity is crucial. The similarity of geographical features at different scales is of great significance, as it can quantify the quality of spatial information transfer in maps and guide the expression of multi-scale map features [12–14]. The similarity of map features includes both semantic and geometric similarity [15], with geometric similarity providing the foundation for semantic similarity, while semantic similarity represents its outward expression.

For geospatial features, measuring geometric similarity between different objects plays a crucial role in various task scenarios [16], such as geospatial feature matching, spatial database retrieval, detection of specific spatial patterns, and evaluation of map generalization [17,18]. In the process of geometric similarity assessment, evaluations are generally conducted from a visual representation perspective, comprehensively measuring aspects such as shape characteristics, topological relationships, and spatial patterns of geospatial features. Research on geometric similarity is typically categorized into three types of geospatial features: points, polylines, and polygons. The study of point features usually involves abstracted building points and POI clusters, with similarity analysis mainly focusing on clustering pattern analysis of point groups [19,20]. Some studies also convert vector point data into images to assess the similarity of points in the map generalization process [21]. For polyline features, commonly used data include rivers, road networks, and contour lines. Similarity measurement research covers tasks such as identifying drainage patterns and measuring the similarity of contour lines [22,23]. Polygon features, as closed polyline features, are represented by datasets such as building groups and islands. Similarity research on these features involves identifying island patterns and classifying building patterns [24,25]. However, while existing studies have employed both traditional spatial methods and deep learning algorithms to evaluate the similarity of geospatial data from different perspectives, current algorithms can only process geospatial data of the same type. They are unable to achieve collaborative analysis and processing of multiple feature types, such as polyline–polygon features, in complex scenarios.

This study focused on the measurement of geometric similarity and presented a framework based on Heterogeneous Graph Neural Networks (HGNNs) [26] for calculating multi-scale geometric similarity in hybrid scenes, along with an example for measuring similarity in hybrid scenes. However, unlike simple problems such as image feature similarity, geometric similarity measurement under scale transformation does not have a unified computation model, and the complexity of mixed scenes (i.e., scenes that contain more than one single type of geometric object) further complicates the issue. Geometric similarity also involves psychological aspects [27], making it an intuitive and subjective cognitive judgment. This is an uncertainty that has long been a focus of geographic studies.

The uncertainty in geometric similarity measurement under scale transformation is expressed in three aspects: cognitive subjectivity, differences in the focused perspectives, and the diversity and complexity of parameters [4]. Although these uncertainties pose challenges for geometric similarity measurement, the objects under scale transformation are essentially defined, and these objects are represented by various geometric features [28]. By integrating these geometric features into a descriptive vector, geometric similarity can be defined based on the similarity between vectors [29]. Specifically, let x and x' represent the features of the geometric objects before and after transformation, respectively. Their similarity can be expressed as a function $s = f(x,\ x\prime)$, which f can be selected as any general vector similarity metric (e.g., Euclidean distance or cosine similarity). However, in scale transformations, geometric features typically involve more than one, and a comprehensive representation of these features is needed to form feature vectors x and x', after which the similarity can be evaluated using function f. The integration of geometric features can generally be achieved through pooling (e.g., average, maximum, etc.) [30], while the challenge lies in characterizing the features of individual geometric objects.

In a certain scene, as long as the adjacency relationships between geometric objects remain unchanged during any transformation, the geometric similarity of the scene will also remain relatively unchanged. Conversely, if the adjacency relationships between geometric objects stay relatively stable, the similarity of the scene before and after transformation will remain constant. Therefore, adjacency relationships can serve as a medium for evaluating

similarity. By incorporating adjacency relationships into the features of geometric objects, and ultimately using pooling methods [30], a geometric feature description of the entire scene can be obtained. It should be noted that when a scene contains only one type of geometric object (e.g., only points, polylines, or polygons), it is relatively easy to integrate adjacency information into node features using simple Graph Neural Networks (GNN) algorithms [31]. However, in mixed scenes, the relationships between polylines and polygons, polylines and polylines, polygons and polygons may be numerically similar but differ in their actual meaning. Therefore, in this study, the HGNN algorithm regards geometric objects as nodes and incorporates different types of adjacency relationships into the node features, thereby generating feature descriptions of geometric objects in mixed scenes through pooling methods, which can then be used to measure geometric similarity under scale transformations.

The remainder of this paper is organized as follows: Section 2 outlines the method used to construct our method based on heterogeneous graph neural network; Section 3 presents and discusses the experimental results; Section 4 concludes our work.

2. Methods

In scenes containing only a single type of geometric feature, geometric objects can be extracted using standard GNNs. However, in hybrid scenes, where multiple types of geometric features are present, standard GNNs are difficult to extract features from different types of edges. To address this problem, this study introduces the HGNN algorithm, which uses deep learning to extract features from each geometric object [26] (the workflow of the proposed method is presented in Figure 1). The input of the method in the cognitive parameters [4] is generated from a mix scene M, and then the parameters are treated as initial features and split into polyline features H_L and polygon H_P. These two types of features are subsequently processed by normalization, parallel HGNN, HAN and feature reduction, to generate the final feature H. At last, with a feature H' computed from another map, the similarity s of the two mix scenes can be obtained. This section first introduces the definition of heterogeneous graphs, then presents the feature extraction framework for geometric features in hybrid scenes based on HGNN and subsequently describes the construction of heterogeneous graphs and the implementation of geometric feature extraction methods based on this framework.

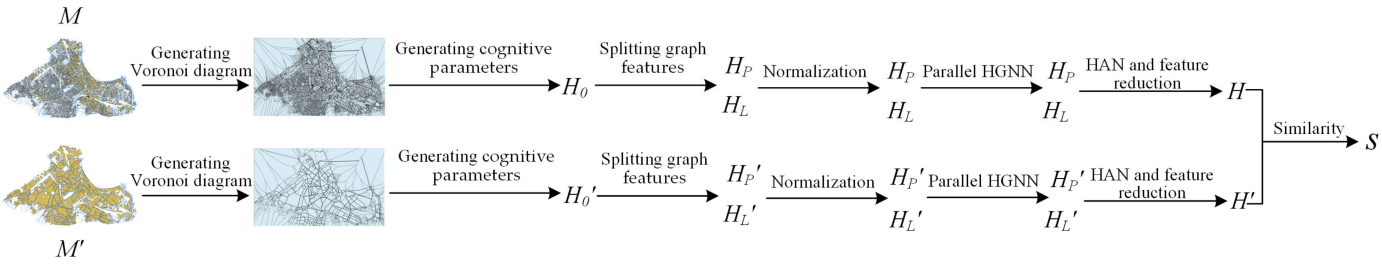


Figure 1. Workflow of the proposed method.

2.1. Framework

Heterogeneous Graph Neural Networks (HGNNs) are a type of deep learning model used to process heterogeneous graphs, which are graphs in which nodes and edges have different types. These are commonly found in fields such as social networks [32], bioinformatics [33], and recommendation systems [34]. HGNNs are able to capture complex relationships and structural information by considering the characteristics of different types of nodes and edges. HGNNs follow the same message-passing and aggregation pattern as standard GNNs [35]:

$$H_t^{\ell} = \underset{\forall s \in N(t),\ \forall e \in E(s,t)}{\text{Aggregate}} \left(\text{Extract}\left(H_s^{\ell-1};\ H_t^{\ell-1}, e \right) \right), \tag{1}$$

where s and t represent the nodes of the graph; e is the edge connecting node s and node t (in hypergraphs, there may be multiple edges between a pair of nodes); $H_s^{\ell-1}$ and $H_t^{\ell-1}$ are the features of nodes s and t from the ℓ^{th} layer network; $N(t)$ represents the neighborhood of node t, which can be defined based on geometric adjacency relationships [36] or topological relationships [37]. For example, neighborhoods can be defined by hops [38], or in heterogeneous graphs, they can be defined using meta-paths [39]; Extract$(\cdot)$ is the feature extraction function, typically a GNN such as GCN (Graph Convolution Network) [40], GAT (Graph Attention Network) [41], SGAT (Sparse GAT) [42], GraphSAGE [43], etc., with the core being message passing, i.e., extracting information from source nodes; Aggregate$(\cdot)$ is the message aggregation function, which aggregates information from neighboring node features and the edge weights connecting them, to form new features H_t^{ℓ}. In general GNNs, nodes s and t are of the same type, so there is only one type of edge in the entire graph. In heterogeneous graphs, however, nodes s and t can be of different types, meaning that there are multiple types of edges.

In an ordinary graph, since all the edges are of the same type, node messages can be aggregated using a network with shared weights. In a heterogeneous graph, however, edges of different types connect nodes of different types, so a network with shared parameters cannot be used to aggregate messages. In heterogeneous graphs, meta-paths are typically used to indicate which types of nodes the network will gather information from to update the target node features. A meta-path is defined as a sequence of nodes, represented as $A_1 \rightarrow A_2 \rightarrow \ldots \rightarrow A_i \rightarrow \ldots \rightarrow A_n$ (where A_i represents the i^{th} type of node i. Once the meta-paths for the heterogeneous graph are defined, the neighborhood $N(t)$ in Equation (1) is constructed based on the meta-path. Even if some nodes are directly adjacent, if they do not belong to the defined meta-path neighborhood, no information will be extracted from those nodes.

The spatial distribution of geometric objects in a hybrid scene is essentially a graph representation, and the types of geometric objects are diverse. Therefore, it is naturally suitable to represent such scenes using heterogeneous graphs: geometric objects are regarded as nodes in the graph, and adjacency relationships between geometric features are regarded as edges, with meta-paths defining the adjacency relationships between different types of nodes. However, when extracting information from geometric features in hybrid scenes using heterogeneous graphs, the primary challenge is the inconsistency in the feature dimensions and units of the nodes (i.e., geometric objects), as well as the different patterns of constructing neighborhood relationships between nodes which cannot be handled by the same model. To address these issues, this paper first introduces two common heterogeneous graph neural network structures: HGT (Heterogeneous Graph Transformer) [44] and HAN (Heterogeneous Graph Attention Network) [45]. Based on these, we proposed a new feature extraction method for geometric objects in hybrid scenes based on HAN. This method utilizes instance normalization (IN) [46] with a linear projection model to solve the problem of inconsistent units, while embedding a parallel GNN within the HAN framework to extract different semantics from different types of edges, thereby generating accurate geometric feature descriptions for hybrid scenes.

2.2. Heterogeneous Graph Transformer

The core of HGT (Heterogeneous Graph Transformer) relies on the learning framework of GAT. The GAT learning framework can be expressed as:

$$\begin{aligned} h_t^{\ell} = & \underset{\forall s \in N(t),\ \forall e \in E(s,t)}{\text{Aggregate}} \left(\text{Attention}(s,t) \cdot \text{Message}(s)\right) \\ & \text{Message(s)} = W h_{\text{s}}^{\ell-1} \\ & \text{Attention}(\cdot) = \sigma(\text{Mean}(\cdot)) \end{aligned} \tag{2}$$

where W is a projection matrix; $\text{Mean}(\cdot)$ and $\sigma(\cdot)$ represent the mean function and activation function (depending on the learning task, either the Sigmoid function [47] or SoftMax function [48] can be chosen); $\text{Message}(\cdot)$ is the message-passing function. HGT uses a simple linear mapping to project the features of different types of nodes into the same linear space; $\text{Attention}(s,t)$ is the core of the GAT network, which extracts attention from nodes s and t according to the predefined meta-path. Essentially, this is a weighting factor [41], which uses attention to weight the projected source node s, and finally, the network uses $\text{Aggregate}(\cdot)$ to perform message aggregation, resulting in updated node features h_t^{ℓ}. The general form of the $\text{Attention}(s,t)$ attention function can be expressed as:

$$\text{Attention(s, t)} = \text{Softmax}(\text{W}_1 \text{h}_{\text{s}}^{\ell-1} \times (\text{W}_2 \text{h}_{\text{t}}^{\ell-1})^{\text{T}}) \tag{3}$$

where W_1 and W_2 are the projection matrices for the features $\text{h}_{\text{s}}^{\ell-1}$ and $\text{h}_{\text{t}}^{\ell-1}$ of nodes s and t, respectively. In a general graph, these parameters can be shared due to the node types being identical.

In the HGT framework, the node types of s and t are different, therefore different projection matrices are used to project the node features, and multi-head attention is employed to extract as much information from the graph structure as possible to update the target node features:

$$\begin{aligned} Attention_{HGT}(s,t) = & \underset{\forall s \in N(t),\ \forall e \in E(s,t)}{\text{Softmax}} \left(\Big\Vert_{i \in [1,\varkappa]} AT^i(s,e,t) \right) \\ AT^i(s,e,t) = & \left(K_s^i W_{\phi(e)}^{AT} \left(Q_t^i\right)^T\right) \cdot \kappa \\ K_s^i = & W_{\tau(s)}^i h_s^{\ell-1} \\ K_t^i = & W_{\tau(t)}^i h_t^{\ell-1} \end{aligned} \tag{4}$$

where $\tau(\cdot)$ represents the node type; K_{s}^i and K_t^i represent the node features after linear projection. Since the node types are different, non-shared linear projection parameters are used to project the features $\text{h}_{\text{s}}^{\ell-1}$ and $\text{h}_{\text{t}}^{\ell-1}$ of nodes s and t. $AT^i(s,e,t)$ represents the attention generated by the i^{th} attention head, $W_{\phi(e)}^{AT}$ is the projection matrix, and $\phi(e)$ represents the type of edge e. Since the edge types in a heterogeneous graph are different, the projection matrices for different types of edges are also different, whereas in GAT, because all edges are of the same type, this matrix becomes the identity matrix. κ is a constant unrelated to learning; $\Vert$ is the concatenation operator, i.e., concatenating the attention from $\varkappa$ heads together to form the final attention $\text{Attention}_{HGT}(s,t)$.

By examining Equations (3) and (4), it can be observed that there are two main differences between the general graph attention network GAT and HGT: first, the use of non-shared parameters for the node feature projection matrices; second, the use of distinct projection matrices for different types of edges. Through these two operations, the features of different types of nodes can interact and integrate, thereby forming new node features according to the learning task.

2.3. Heterogeneous Graph Attention Network

The Heterogeneous Graph Attention Network (HAN), as with HGT, uses the graph attention framework for message aggregation and propagation, and HAN utilizes the graph attention generation method in Equation (4). Unlike HGT, HAN introduces a novel

semantic aggregation method to aggregate information from different types of meta-paths. By examining Equation (2), it is evident that that in HGT, the message aggregation function uses a simple Sigmoid function. This design aims to keep the parameters of the graph neural network minimal, making it suitable for large-scale graph data. However, nodes in different meta-paths should have different semantics. If only a simple projection followed by a Sigmoid function transformation is used, the semantic information related to the meta-paths cannot be fully captured in the node features. To address this issue, HAN also utilizes the attention mechanism and proposes a method for integrating node semantic features.

HAN first utilizes Equation (2) to extract the features of nodes under predefined meta-paths, represented as $\left\{h^{\ell}_{\Phi(e)}\right\}_{\Phi(e)\in[1,n]}$, where n denotes the number of predefined meta-path types. The final node features, which incorporate semantics, should be represented as the weighted average of the node features:

$$h^{\ell}_{s} = \sum_{\phi(e)=1}^{n} \beta_{\phi(e)} \cdot h^{\ell}_{\phi(e)} \tag{5}$$

where $\beta_{\phi(e)}$ represents the weight coefficient of the node features $h^{\ell}_{\phi(e)}$ constructed from the meta-path corresponding to edge e, which can be obtained utilizing an attention mechanism:

$$\begin{aligned} \beta_{\phi(e)} &= \frac{\exp\left(w_{\phi(e)}\right)}{\sum_{\phi(e)\in[1,n]} \exp\left(w_{\phi(e)}\right)} \\ w_{\phi(e)} &= \frac{1}{n}\sum_{\phi(e)=1}^{n} q^{T} \cdot \tanh\left(W_{h} \cdot h_{\phi(e)} + b\right) \end{aligned} \tag{6}$$

where $w_{\phi(e)}$ represents the attention extracted from the meta-path corresponding to edge e, while W_h, q, and b are the parameters to be learned.

By comparing Equations (2) and (5), it can be observed that after HGT learns node features from various types of meta-paths, it performs a simple linear projection on these features and then combines them into a single node feature through a Sigmoid function transformation. In contrast, HAN takes into account the semantic information defined under different meta-paths and employs an attention mechanism to integrate the semantic features from different meta-paths, resulting in a final node feature.

2.4. Parallel Network for Geometric Feature Extraction Based on HAN

As described in Section 2.3, HAN builds upon HGT and provides a heterogeneous graph node feature extraction framework. This framework first extracts node features based on meta-paths, and then extracts the semantic information related to these meta-paths from the features, thereby forming the final heterogeneous graph node features. However, there are two apparent issues with the HAN framework: (1) HAN uses a linear projection approach, attempting to learn a weight matrix to project the original features of different nodes into the same linear space. The fundamental assumption behind this operation is that there exists a linear transformation between the original feature vectors. However, since each dimension of different geometric feature vectors has a different physical meaning, the likelihood of a linear transformation existing between features of different types of geometric features is minimal. (2) Both HAN and HGT use the graph attention mechanism to aggregate node information and update node features, but they do not account for the fact that the attention extracted from the connection relationships between different geometric features carries different physical meanings. In light of these issues, this study proposed a parallel structure for the geometric feature extraction network. This network first uses normalization to eliminate the physical units of the original node features, enabling linear mapping to project the features of different nodes. The feature extraction network is then designed as a parallel structure to explore different adjacency

relationships between features. This structure allows the use of both a single node feature aggregation method and a mixed use of multiple feature aggregation methods.

2.4.1. Normalization of Initial Node Features

The initial node features must be normalized to eliminate dimensions in order to be used for subsequent model training; otherwise, the model will fail to converge during training (experimental results are detailed in the ablation study of Section 3.2.3). Initially, the IN method [46], which does not involve learnable parameters, is employed to normalize the initial vectors.

$$h_s^\ell = \frac{h_s^0 - E\left(h_s^0\right)}{\sqrt{\mathrm{Var}\left(h_s^0\right) + \varepsilon}} \tag{7}$$

where h_s^0 is the initial features of node s; $E\left(h_s^0\right)$ is the mean of all features in h_s^0; var is the variance of all features in h_s^0; ε is a small constant, typically set to 10^{-5}, to avoid division by zero. The essence of the instance normalization method is to center and then scale the vector, effectively eliminating dimensions while also addressing data offset errors.

After eliminating dimensions through instance normalization, a weight matrix can then be learned to project different types of node features into the same linear space:

$$h_s^\ell \leftarrow W_{\tau(s)} \times h_s^\ell \tag{8}$$

where $\tau(s)$ represents the node type corresponding to node s; therefore, different node types utilize different weight matrices for projection. It is important to note that GNNs typically learn using mini-batch samples as training data, and in this case, it is advisable to use batch normalization (BN) [49] to mitigate the differences between samples:

$$h_s^\ell \leftarrow \frac{h_s^\ell - E\left(\left\{ h_s^\ell \right\}_{s=1}^b\right)}{\sqrt{\mathrm{Var}\left(\left\{ h_s^\ell \right\}_{s=1}^b\right) + \varepsilon}} \tag{9}$$

where $\left\{h_s^\ell\right\}_{s=1}^b$ denotes the b sample data consisting of features from all nodes during a single training iteration; $\mathrm{E}(\cdot)$ and $\mathrm{Var}(\cdot)$ have the same meanings as in Equation (7). Batch normalization (BN) assumes that all training data have approximately the same probability distribution; therefore, the BN operation must be performed after instance normalization and linear projection.

2.4.2. Parallel Network for Geometric Feature Extraction

After the initial node features undergo normalization, they can be considered to reside in the same linear space. At this point, it is necessary to use the paradigm provided in Equation (1) for feature extraction and aggregation. The proposed parallel network does not rely on meta-paths; instead, it decomposes the meta-paths into edges that connect the various nodes. In an undirected graph, there exists an equivalence between meta-paths and edges. Let the predefined meta-path be $A_1 \to A_2 \to \ldots \to A_i \to \ldots \to A_n$ (where A_i represents the i^{th} type of node), then:

$$A_1 \to A_2 \to \ldots \to A_i \to \ldots \to A_n \Longleftrightarrow \{E(A_i, A_{i+1}), E(A_{i+1}, A_{i+2})\}_{i=1}^{n-2} \tag{10}$$

where $E(A_i, A_{i+1})$ denotes the edge connecting node types A_i, A_{i+1}. Based on different types of edges, the original graph can be decomposed into various subgraphs. For instance, in the subgraph defined by the edge $E(A_1, A_2)$, only nodes of type A_1 and A_2 are included,

and the subgraph defined by this edge can be represented as $G_{E(A_1,A_2)}$. Thus, the entire graph G can be represented as:

$$G \triangleq \left\{ G_{E(A_i,A_j)} \right\}_{i,j\in[1,n]} \tag{11}$$

where after decomposing the entire graph into subgraphs based on the edges connecting different nodes, the various subgraphs become isolated from one other. Therefore, parallel graph neural networks can be employed to process each subgraph, extracting node features from the individual subgraphs. Ultimately, the features obtained from each subgraph can be weighted according to Equation (5) to yield node features that incorporate semantics. The advantage of using parallel networks for feature extraction is that they do not exhibit significant bias toward any particular type of node during the feature extraction process. From Equation (11), it can be seen that for a complete graph G containing n types of nodes, the total number of decomposed subgraphs is n^2. For a specific node type A_i, it forms subgraphs with n types of nodes (including its own type), resulting in a total of n features, which can be represented as:

$$h_i \triangleq \left\{ h_{G_E(A_i,A_j)} \right\}_{j=1}^{n} \tag{12}$$

where the equation has the same mathematical form as the node features extracted by HAN; therefore, the form used in HAN for aggregating semantic information can be directly applied for semantic feature aggregation (i.e., Equations (5) and (6)).

2.4.3. Node Feature Extraction Method for Decomposed Subgraphs

As mentioned earlier, the normalized node feature vectors reside in the same linear space. Once the complete graph is divided into individual subgraphs based on edges, the subgraphs become isolated from one other, allowing for the extraction of node information within each subgraph using traditional graph neural network methods. There are many traditional graph neural network methods; the classic Graph Convolutional Network [11] is used as an example to detail the method for extracting node features from subgraphs.

The GCN considers the node feature matrix $H \in R^{m\times d}$ (where m is the number of nodes and d is the dimensionality of the node feature vectors) as the signal of the graph. The convolution operation on the graph can be expressed as:

$$g_w * H = U g_w U^T H \tag{13}$$

where $*$ denotes the convolution operation; g_w is the convolution kernel, where w represents the parameters to be learned. U is the feature matrix of the normalized graph Laplacian matrix $\tilde{L}$ [50] which can be expressed as:

$$\tilde{L} = I - D^{-1/2} M D^{-1/2} \tag{14}$$

where $M \in R^{m\times m}$ is the adjacency matrix of the graph (which can also be the edge weight matrix); $D \in R^{m\times m}$ is the degree matrix; $I \in R^{m\times m}$ is the identity matrix.

From Equation (14), it can be deduced that if the number of nodes is large, the number of features in $\tilde{L}$ grows quadratically. Calculating its feature matrix then becomes very time-consuming, and since the feature matrix U is the same size as $\widetilde{L}$, its multiplication with other matrices is also very computationally intensive. To improve computational efficiency, the GCN uses a first-order approximation of the Chebyshev polynomial [51,52] to

replace Equation (13), while employing a reparameterization strategy to enhance numerical stability. Thus, the GCN can be expressed as:

$$h_s^{\ell} = \sigma\left(\sum_{t \in N(s)} \frac{\tilde{D}_{i,i} \times \tilde{D}_{j,j}}{\tilde{M}_{i,j}} \times h_t^{\ell-1} W\right) \tag{15}$$

where $\tilde{M} = M + I$ is the adjacency matrix that includes self-connections, where $\tilde{M}_{i,j}$ is the feature in the i^{th} row and j^{th} column of $\tilde{M}$. $\tilde{D}$ is a diagonal matrix where $\tilde{D}_{i,i} = \sum_j \tilde{M}_{i,j}$; W represents the parameters to be learned.

From Equation (15), it can be observed that a single-layer GCN network contains only the parameter W, which is significantly fewer than other graph neural networks (such as GAT or GraphSAGE). Moreover, experiments have shown that its performance is generally comparable to other graph neural networks that include more parameters, making it commonly used for processing graph-structured data.

2.5. Feature Reduction and Similarity Measurement

After extracting the features of graph nodes using HGNN, a feature reduction process is required, which involves aggregating the features of all nodes to generate a vector that serves as an algebraic representation of the entire graph. This vector should theoretically encompass all information regarding the graph's nodes and edges. Common approaches to feature reduction can be categorized into supervised (e.g., DiffPool) [53] and unsupervised methods. This paper adopts an unsupervised approach for the reduction of feature vectors. The average pooling method can be expressed as:

$$h = \frac{1}{m} \sum_{\forall s} h_s \tag{16}$$

which essentially computes the average of all graph nodes. Since the parallel HGNN algorithm proposed in this paper has already integrated the adjacency information of nodes into the graph node features during training, the average pooling method encompasses information from the entire heterogeneous graph. Experimental results demonstrate that the feature reduction method based on average pooling is not only simple to implement but can also accurately describe the characteristics of the graph.

After feature extraction and reduction, the geometric features of mixed scenes can be represented as a feature vector of dimension d. Under scale transformation, each mixed scene corresponds to a feature vector. The similarity between two scenes under scale transformation can be measured using the similarity of their corresponding feature vectors:

$$sim(g, h) = \frac{g^T \times h}{|g| \times |h|} \tag{17}$$

where g and h represent the feature representations of the geometric features of mixed scenes at two different scales. Essentially, Equation (17) represents the cosine similarity of vectors, which is unaffected by the length and rotational transformations of the features. Consequently, it is widely used in the measurement of vector similarity.

3. Experimental Results and Analysis

This section focuses on multi-scale maps, constructing heterogeneous graphs using line and polygon features in the maps as heterogeneous nodes. These heterogeneous graphs serve as the learning data, and an unsupervised learning approach is employed to train the parallel network based on HAN to obtain optimal parameters. To validate the effectiveness

of the algorithm, both quantitative comparisons and qualitative questionnaire surveys are conducted.

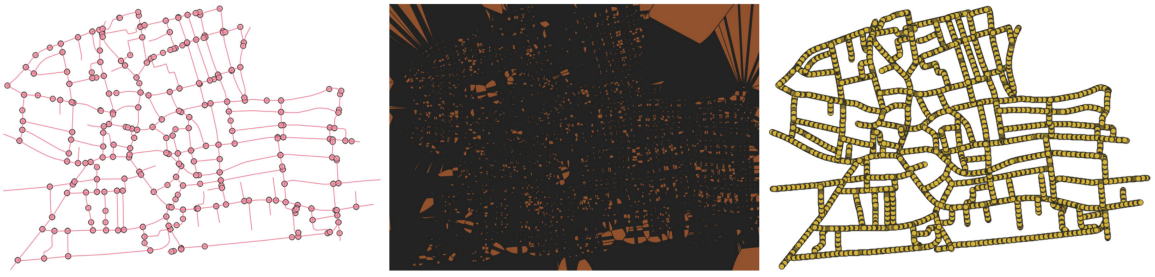
3.1. Experimental Data

The original data for the experiment come from the Federal Office of Topography Swisstopo (FOTS) in Switzerland (https://www.swisstopo.admin.ch/en/national-maps) (accessed on 26 October 2024). The original data consist of map data at various scales. After preprocessing, they are divided into training and testing datasets for model training and evaluation.

3.1.1. Data Preprocessing

This study evaluates the proposed algorithm using polyline and polygon geometric features in the map as instances of mixed scenes. For a single map, the polygons (buildings) and polylines (roads) in the map are converted into nodes of a heterogeneous graph while simultaneously capturing the characteristic information of these graph nodes. The heterogeneous graph and its node features are constructed based on the Voronoi diagram. Firstly, the lines representing buildings and roads are segmented at their intersections. For each polyline segment, points are generated at fixed intervals to create denser points along the polylines, with these points serving as the nodes of the heterogeneous graph. If a polyline segment is obtained from interrupting a road, the corresponding dense points are classified as nodes of type "polyline". Similarly, if a dense point originates from a polyline that constitutes a building, it is classified as a node of type "polygon". Based on these dense points, a Voronoi diagram can ultimately be constructed.

Figure 2 illustrates the process of segmenting a road at intersections, generating dense points, and constructing the Voronoi diagram. After the Voronoi diagram is created, its vertices become the nodes of the heterogeneous graph, and the adjacency relationships between the graph nodes correspond to the adjacency relationships of the vertices in the Voronoi diagram.



(**a**) (**b**) (**c**)

Figure 2. Construction process of the Voronoi diagram for roads. (**a**) Segmentation of the road into polyline segments at intersection points; (**b**) densification of each polyline segment; (**c**) Voronoi diagram composed of dense points.

After establishing the nodes and node relationships of the heterogeneous graph, it is also necessary to assign initial features to each node. The construction of initial node features is based on cognitive parameters [54]. The parameters used to construct the initial features for "line vertices" and "polygon vertices" are shown in Table 1.

Following the aforementioned approach, heterogeneous graphs corresponding to all maps can be constructed, thereby establishing a mapping relationship between the maps and the heterogeneous graphs, denoted as $\Phi : \mathrm{M} \rightarrow \mathrm{G}\left(\mathrm{V}, \mathrm{H}^0, \mathrm{M}\right)$. Here, M and G represent the map and its corresponding heterogeneous graph after mapping, while V,

H, and M signify the nodes of the graph, the initial feature vectors of the nodes, and the adjacency relationship matrix of the nodes, respectively.

Table 1. Cognitive parameters used for constructing initial features of Heterogeneous Graph Nodes.

Vertex Type	Selected Parameters
Polyine Vertex	Number of Neighbors, Direction of Minimum Bounding Rectangle, Length Ratio of Minimum Bounding Rectangle, Polyine Length, Curvature, Spatial Compactness
Polygon Vertex	Number of Neighbors, Direction of Minimum Bounding Rectangle, Length Ratio of Minimum Bounding Rectangle, Polygon Area, Polygon Perimeter, Number of Points, Spatial Sparsity

3.1.2. Dataset for Deep Learning

The dataset is derived from 21 map data sets available on the FOTS website, with scales ranging from 1:10K to 1:500K. Due to the large size of individual maps, it becomes impractical to convert the entire map into a heterogeneous graph that can fit into a graphics processing unit (GPU) for training. Additionally, since the majority of the content in the maps consists of roads, training on a heterogeneous graph dominated by a single type of node can easily lead to class imbalance issues. Therefore, in creating the learning dataset, this study cropped local regions from each map that contained a rich presence of both roads and buildings, constructing heterogeneous graphs based on these local maps. Ultimately, a total of 403 local maps were cropped from the 21 complete maps to build the heterogeneous graphs (examples of local maps used for training can be seen in Figure 3), with each heterogeneous graph containing approximately 1000 to 17,000 nodes. These 403 heterogeneous graphs were then split into training and validation sets in a ratio of 9:1.

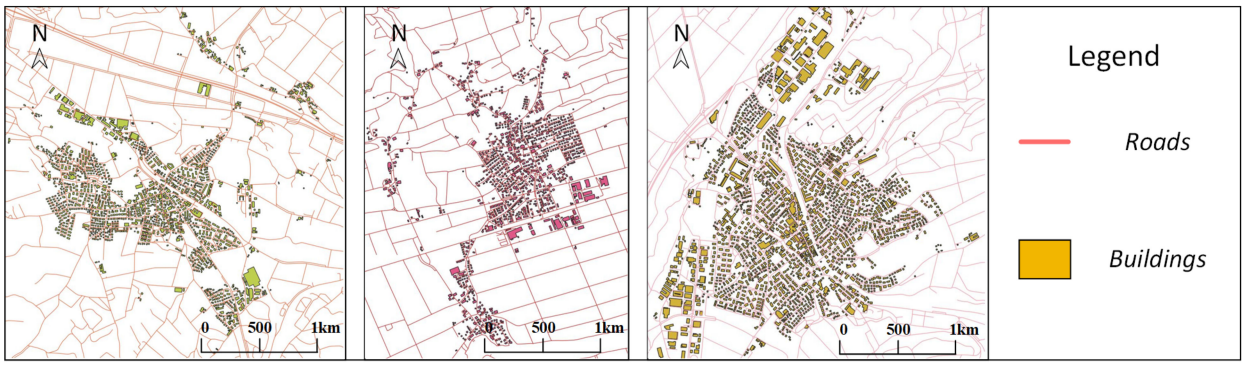


Figure 3. Examples of local maps for training.

The training set is used to train the network parameters, while the validation set is utilized to compute key performance metrics, which guide the adjustment of hyperparameters during training to enhance network performance. The basic metrics computed on the validation set include six indicators: Loss, ROC (Receiver Operating Characteristic), AP (Average Precision), Recall, Precision, and F1 score. Among these six metrics, the most critical are Loss and Recall. When the network model exhibits the lowest Loss and the highest Recall on the validation set, it is considered to be the optimal network model.

The test data consists of 10 complete maps, which were used to construct heterogeneous graphs employing the same method as that used for creating the learning dataset. Detailed information regarding the map scales and the number of nodes in the constructed heterogeneous graphs is provided in Table 2.

Table 2. Scale and number of vertices in the test dataset.

Region	Map Scale	Number of Vertices
Basel	1:10K 1:15K 1:25K 1:50K 1:100K 1:250K 1:500K	16,627 14,075 11,413 7758 4804 2177 331
Bern	1:10K 1:15K 1:25K 1:50K 1:100K 1:250K 1:500K	9845 8137 6360 4529 2974 502 149
Genève	1:10K 1:15K 1:25K 1:50K 1:100K 1:250K 1:500K	14,942 11,586 6118 3575 582 156
Lausanne	1:10K 1:15K 1:25K 1:50K 1:100K 1:250K 1:500K	13,693 10,938 9236 5323 3016 472 136
Zurich	1:10K 1:15K 1:25K 1:50K 1:100K 1:250K 1:500K	13,993 10,939 8479 6181 4210 1135 366
AaRau	1:15K 1:25K 1:50K 1:150K 1:500K	9666 5265 3017 639 115
Bulle	1:15K 1:25K 1:50K 1:150K 1:500K	7491 3877 2406 864 142
Chur	1:15K 1:25K 1:50K 1:150K 1:500K	6727 4224 2552 580 216
Sion	1:15K 1:25K 1:50K 1:150K 1:500K	3888 2261 1590 354 127
Zug	1:15K 1:25K 1:50K 1:150K 1:500K	9829 5759 3694 404 172

For each region, the reduced features are calculated based on the original scale map (i.e., the map with the largest scale in each group). These reduced features are then compared with the reduced features of other scale maps to compute geometric similarity according to Equation (17) in order to evaluate the effectiveness of the proposed algorithm (the distribution of test data and data samples can be seen in Figure 4 and Table 3).

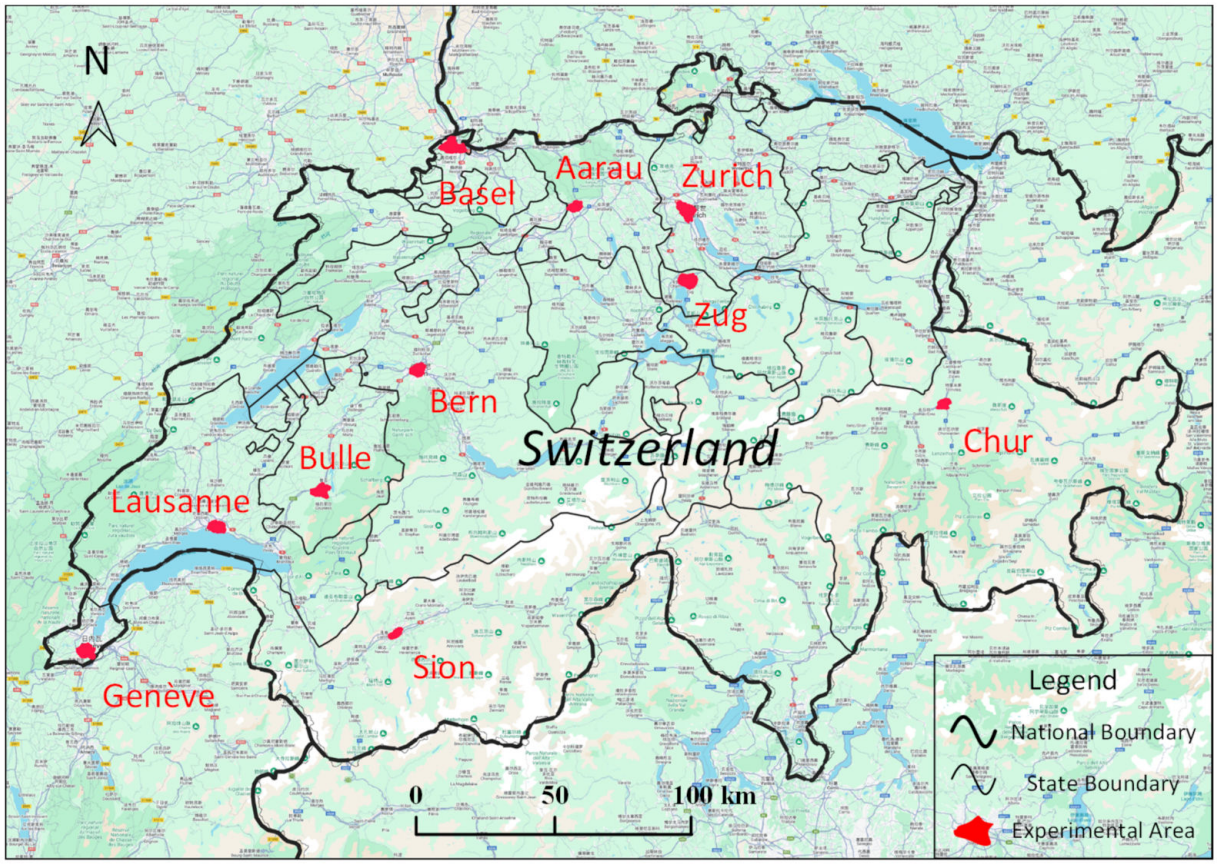


Figure 4. The regions of test dataset.

Table 3. Test data samples from the Zurich and Chur regions.

	Origin Scale	Target Scale					
	1:10K	**1:15K**	**1:25K**	**1:50K**	**1:100K**	**1:250K**	**1:500K**
Zurich							
	Original Scale	**Target Scale**					
	1:10K		**1:25K**	**1:50K**	**1:150K**		**1:500K**
Chur							

3.2. HGNN Algorithm Details

This study implements a parallel network based on the HAN framework using the PyTorch deep learning framework [55] and the torch geometric library for graph neural networks [56]. The dataset used for training the network is derived from Section 3.1.2. The optimization objective is to minimize the negative log-likelihood, resulting in optimal network parameters and completing the HGNN training process.

During the inference phase, scene data at different scales are used as inputs to the network, which output the feature reduction vector for the map scene data. By calculating the similarity between these vectors, the geometric similarity of the features at different scales can be measured. This section will provide a detailed overview of the implementation details of the proposed algorithm.

3.2.1. Network Structure

The structure of the parallel network based on the HAN framework is shown below:

The proposed framework consists of four modules: feature normalization, parallel HGNN, HAN, and feature reduction (Figure 5). Following the data preprocessing method outlined in Section 3.1.1, the map data can be converted into a heterogeneous graph G, while obtaining the initial features H^0 of the heterogeneous graph. Subsequently, the heterogeneous graph G is split into two subgraphs G_L (containing only "polyline" objects) and G_P(containing only "polygon" objects), with their initial node features denoted as $H_{P,0}$, and $H_{L,0}$, respectively. The feature normalization network consists of two nonparametric operations (i.e., IN and BN) and a linear projection. The first two operations are utilized to de-dimensionalize the initial features and the projection ensures that the normalized features share the same dimension D. The following module is the parallel HGNN in which four HGNN modules are parallelized. GNN_L and GNN_P are designed to learn features from the single polyline and polygon objects, and $GNN_{P,L}$ and $GNN_{L,P}$ try to learn features from the mixed polyline and polygon objects. The four modules output features with the same dimension D and then they are modulated by the HAN module to give a d dimension feature for every node. Finally, the feature reduction module averagely pools all the features of a mix scene and generates a single d dimension vector, i.e., the comprehensive feature

H of the heterogeneous graph G. This feature vector d also represents the generalized characteristics of the map's geometric features.

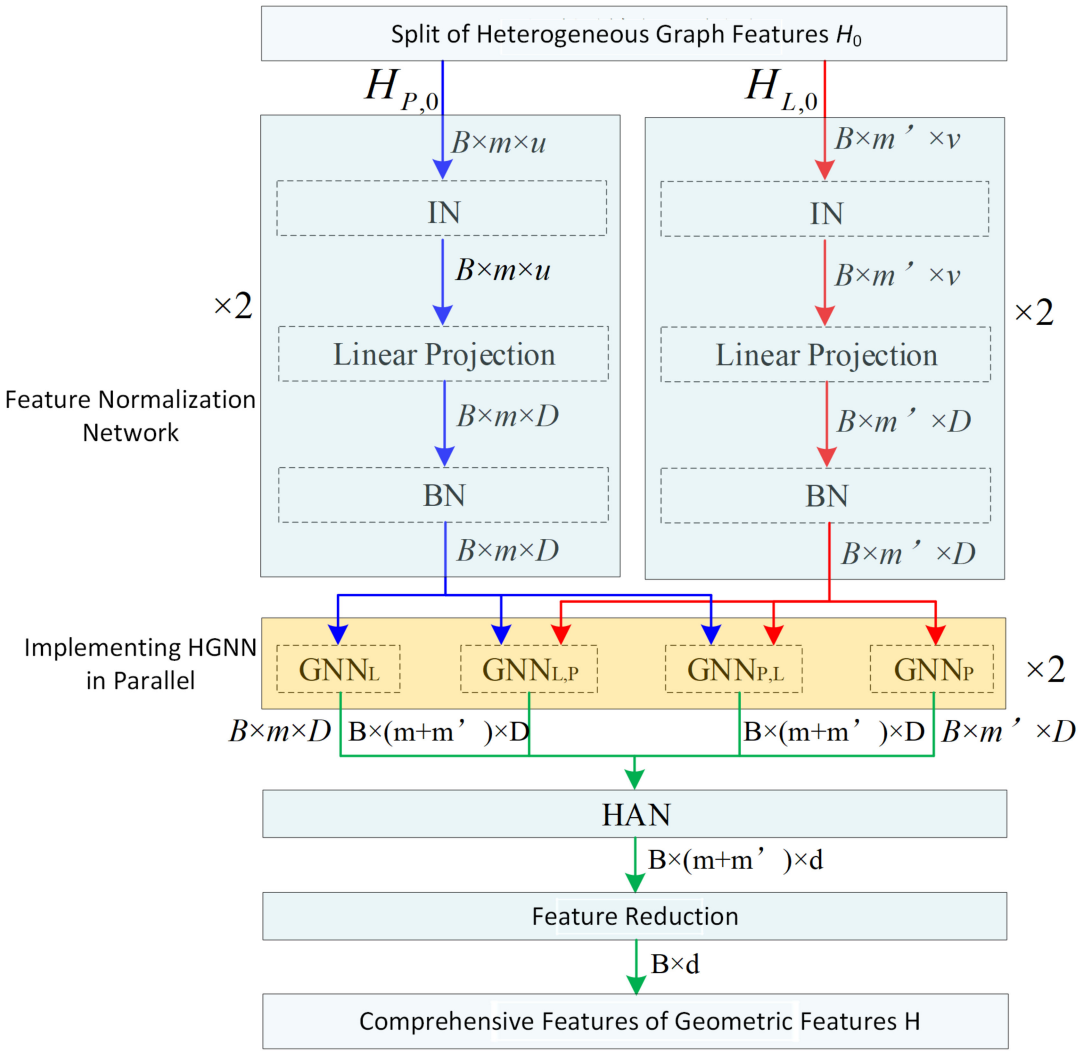


Figure 5. Parallel GNN feature extraction network for geometric features based on the HAN Framework. B represents the number of small batch samples during training; m and m' denote the number of features; u, v, d, and D refer to the feature dimensions; ×2 indicates a two-layer structural network.

The parallel HGNN consists of four sub-GNN modules, and the types of these sub-modules can be freely selected (for example, GCN, GraphSAGE, GAT, etc.). The relational patterns among different geometric objects should vary; therefore, it is necessary to use GNN modules with different parameters for their integration. Utilizing these four sub-modules with distinct learning parameters, different types of geometric relationships can be learned. Among them, the GNNL module primarily learns the relationships between geometric features composed of "polyline" objects, while the GNNP module learns geometric relationships from "polygon" objects. The $GNN_{P,L}$ and $GNN_{L,P}$ modules learn geometric relationships from mixed "polyline–polygon" features. In the ablation study presented in Section 3.2.3, we shall see that employing four parallel sub-GNN modules can better learn complex geometric relationships, thereby facilitating the generation of accurate generalized features for map geometry. To identify the most effective types of sub-GNN modules, this study integrated four different types of GNN models—GCN, GraphSAGE, GAT, and SGAT—into the parallel GNN module. The ablation experiment revealed that the four types of sub-GNN blocks exhibit roughly comparable performance, with GCN

performing slightly better than GraphSAGE, followed by the others; specific details can be found in Section 3.2.3.

Using the geometric relationships learned from the four sub-GNN modules, the normalized features can be weighted to obtain new graph node features. Since each module outputs features for the graph nodes, a single node will contain three features of length D from the GNN modules. For example, for a node of type "polyline," it will respectively input to the GNN_L, $GNN_{P,L}$, and $GNN_{L,P}$ modules, whereby each of these three modules will output a new feature of length D. These features can be regarded as the representation of this "polyline" node across three different graphs, and the final feature of this "polyline" node should be the generalization of these three feature vectors. A common method for feature generalization is to concatenate the three feature vectors end-to-end; however, this simple approach does not effectively compress the redundant information within the feature vectors. Therefore, this study employs the HAN method described in Section 3.3 for feature generalization.

3.2.2. Training Details

Similar to the training paradigm of Variational Graph Autoencoders (GAE) [57], this study adopted a self-supervised approach for training the network model. The loss function for model training is the negative log-likelihood, expressed as:

$$L = -log\left(1 - Sigmoid\left(Sum\left(H^T H\right) \bigodot (I - M)\right)\right) \tag{18}$$

where $\bigodot$ denotes the Hadamard product; I represents a matrix with all features equal to 1; $\mathrm{Sum}(\cdot)$ indicates the summation of all features within the matrix; $\mathrm{Sigmoid}(\cdot)$ refers to the Sigmoid function. During the training process, it is only necessary to input the initial features H^0 of the heterogeneous graph nodes and the adjacency matrix M, which represent the fundamental information of the heterogeneous graph. Thus, the proposed parallel network can be trained using the heterogeneous graph alone without requiring additional supervisory signals.

In Equation (18), the first term represents the feature similarity between adjacent nodes, while the second term denotes the similarity of features between non-adjacent nodes. In the training process, an "adversarial relationship" is formed between adjacent and non-adjacent nodes, meaning that after minimizing L, the features of adjacent nodes become highly similar, while the features of non-adjacent nodes remain dissimilar. This approach integrates the adjacency relationships of the heterogeneous graph into the node feature representation.

For training the network, the optimizer used was Adam with an initial learning rate of 0.005. A learning rate decay technique was employed, reducing the learning rate to 0.9 of the current value every 150 training epochs. The mini-batch size for the training samples was set to 4, and the total number of training epochs was 2000. An early stopping strategy was also implemented; if the rate of change in Loss is less than 0.01 over 100 training epochs, indicating that the model has converged, the training is terminated. To identify the best training hyperparameters and to avoid network overfitting, the six performance evaluation metrics used in Section 3.1.2 were employed to record the changes in these metrics, with the results presented in Figure 6.

As shown in Figure 6, during the model training process, the losses on both the learning dataset and the validation dataset steadily decreased, with only a small gap between the two. This indicates that the model extracted in this study is convergent and shows no signs of overfitting. Comparing the performance of the four embedded sub-GNN models on the validation set, SGAT performed slightly worse than the other three methods, which may be attributed to its smaller number of parameters.

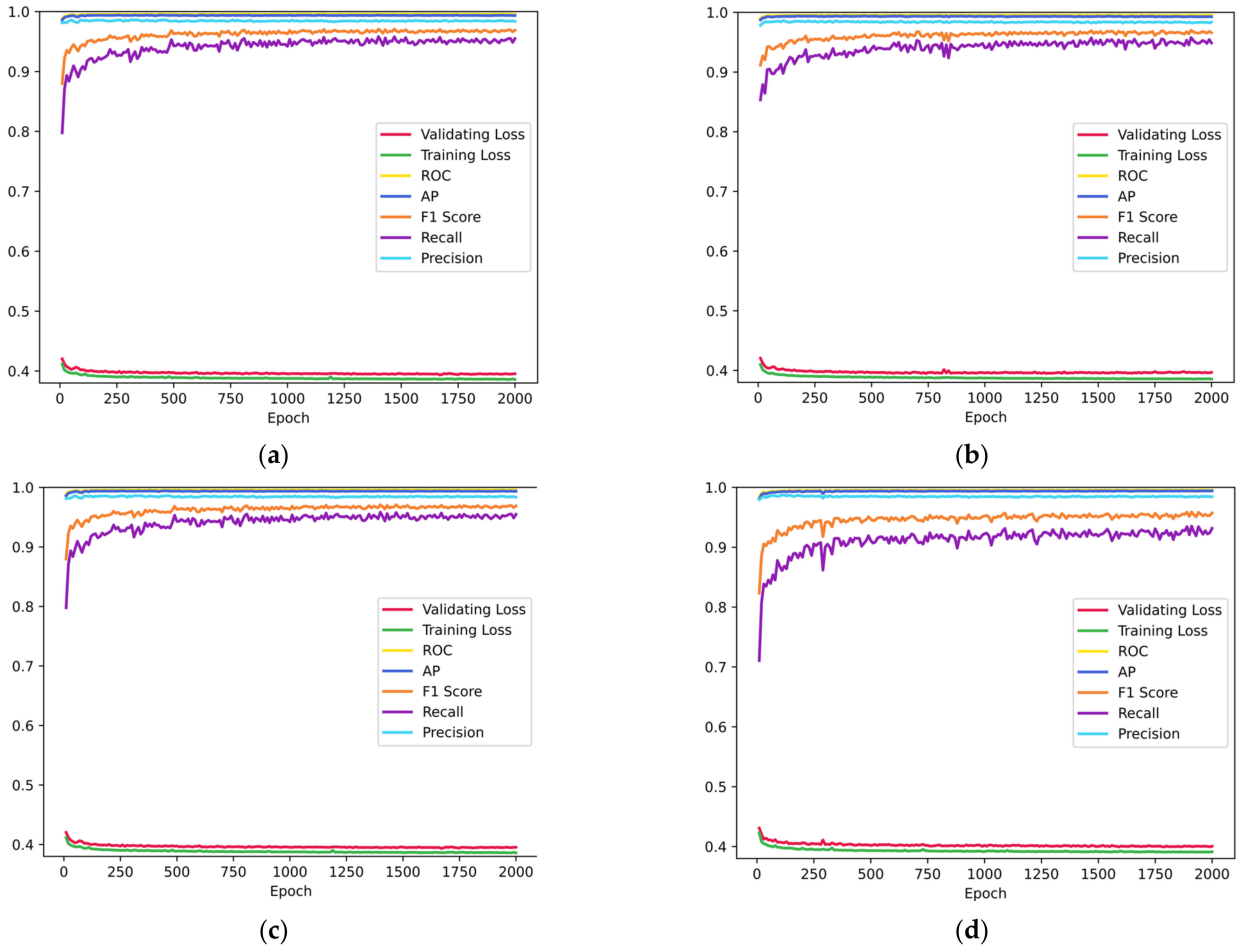


Figure 6. Variations of metrics during model training. (**a**) GCN, (**b**) GAT, (**c**) SGAT, (**d**) GraphSAGE.

Additionally, all four sub-GNN models achieved Recall and F1 scores exceeding 94% on the validation set, demonstrating the model's ability to accurately reconstruct the adjacency relationships of the majority of nodes. This ensures that the final feature reduction vector accurately represents the properties and adjacency characteristics of the heterogeneous graph. Other metrics, such as the final values of the ROC curve, Precision curve, and AP curve, approach 100%, indicating that the model fits the training data well and possesses a certain degree of generalization potential, making it suitable for subsequent experiments on multi-scale geometric similarity measurement in mixed scene testing.

3.2.3. Ablation Experiment

To validate the necessity of the normalization network and the parallel network extracted in this study, an ablation experiment was conducted by removing the normalization network module or replacing the four sub-GNN modules with different parameters in the parallel module with a single GNN model. By comparing the aforementioned metrics, the necessity of these two modules can be verified.

- **Removing the Normalization Network**: In the experiment with the normalization network removed, the processes of instance normalization and batch normalization were eliminated. Instead, only the features of different types of nodes were linearly projected without performing the operations described in Equations (7) and (9). Instead, the operation defined in Equation (8) was directly applied, mapping the feature vectors of all nodes to the same length, and then training the model according to the procedure outlined in Figure 5. During the training process, the hyperparameters

described in Section 3.2.2 were utilized, and the curves of various metrics are shown in the figure below:

As shown in Figure 7, regardless of which sub-GNN module is embedded, when the normalization module is removed, the Precision of the model on the validation set is consistently 0.5. This indicates that the model is unable to accurately determine adjacency relationships and therefore cannot integrate the heterogeneous graph node features based on these relationships. Consequently, the generated features fail to describe the geometric features and relationships within the map.

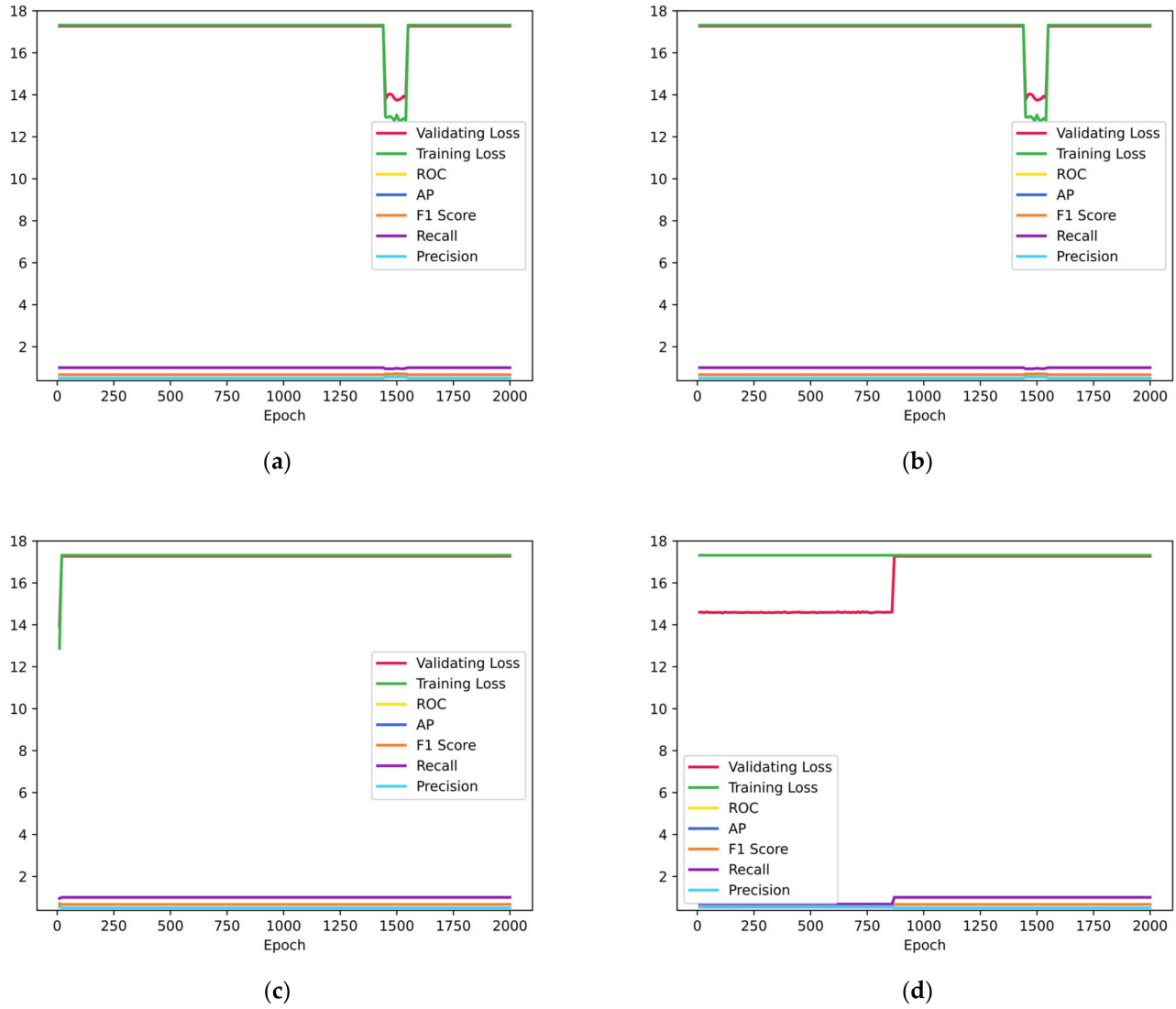


Figure 7. Variations of metrics during training after removing the normalization network model. (**a**) GCN, (**b**) GAT, (**c**) SGAT, (**d**) GraphSAGE.

The underlying reason for this is that the initial input features have different dimensions, resulting in significant disparities in the values between each feature component. This prevents the neural network from learning the transformation relationships among the data, leading to an inability to correctly differentiate whether nodes are adjacent. It should be noted that if the normalization module is also removed from the methods under the VGE framework, similar results will occur, indicating that the model fails to converge at an optimal position and cannot accurately distinguish adjacency relationships between nodes.

- **Replacing Parallel Sub-GNNs with a Single GNN**: In this experiment, the normalization network is retained while the four parallel sub-GNN networks are replaced by a single GNN network. Since the model parameters of the four sub-GNN networks exceed those of a single GNN, to ensure consistent parameter scale, the four individual sub-GNN networks are concatenated together, resulting in a four-layer stacked GNN network structure. The training parameters continue to utilize the hyperparameters described in Section 3.2.2, and the curves of the various metrics are shown in the figure below:

As is shown in Figure 8, after replacing the parallel model with a four-layer stacked model, the training loss curve remains unchanged after 1000 epochs of training, indicating that the model begins to converge thereafter. However, the loss on the validation set shows an upward trend, and the gap between the two loss values gradually widens, indicating that the model starts to exhibit signs of overfitting. Additionally, the recall curve fluctuates consistently at a low level (generally below 40%), indicating that the model is currently unable to accurately distinguish whether nodes are adjacent or not. As a result, the model cannot effectively integrate the node feature information, preventing it from generating precise reduced features for the heterogeneous graph.

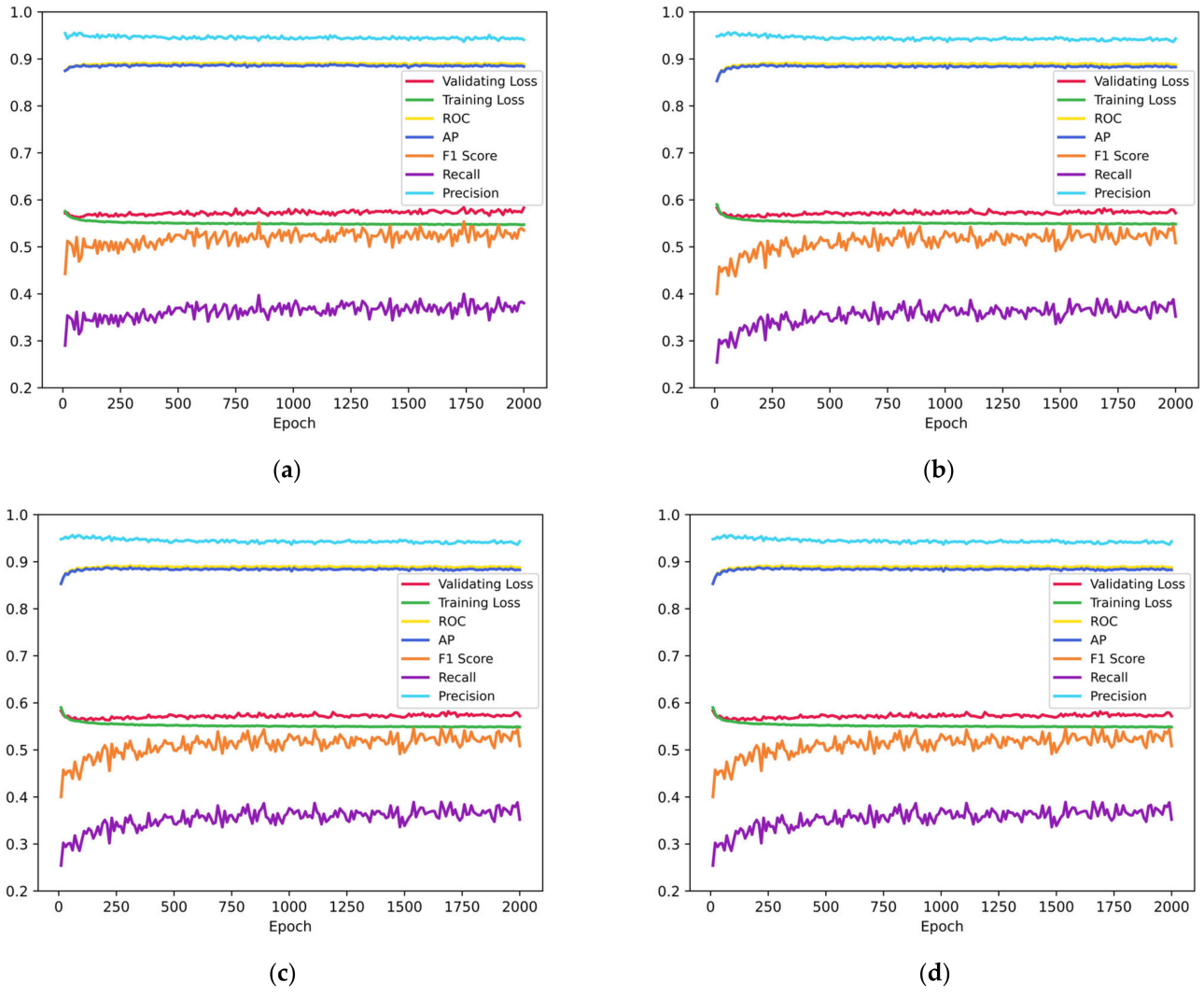


Figure 8. Variation of metrics during Ttraining after replacing the parallel model with a four-layer stacked model. (**a**) GCN, (**b**) GAT, (**c**) SGAT, (**d**) GraphSAGE.

This experiment also verified that HGNN based framework cannot be replaced by GNN base framework, and more precisely, simply replacing the parallel network with a stacked one does not work. This conclusion can be drawn not only from the comparison between Figures 6 and 8, but also from a theoretical analysis. Stacked GNN views the upstream outputs as homogeneous data and tries to mine features from them, while for the proposed parallel network, it applies different networks to process different types of data, thus it has the awareness of different data structures, and consequently learns more informative features from geometric objects and outperforms stacked methods.

3.3. Model Evaluation

To validate the effectiveness of the proposed parallel model measurement method based on the HAN framework, this section conducted two comparative experiments: one is a quantitative experiment, comparing it with other geometric similarity measurement models, and the other is a qualitative comparison, examining the reliability of the geometric similarity measurement results of this model through a questionnaire survey. In the quantitative experiment, traditional non-deep learning methods were unable to handle heterogeneous data; therefore, this study selected GNN-based methods as the comparison targets.

3.3.1. Quantitative Evaluation

This study primarily compared two types of graph neural networks: GNN and HGNN. To enable GNN to process heterogeneous data, the GAE framework has been implemented, along with the normalization method described in Section 3.2.1. Additionally, GCN, GAT, SGAT, and GraphSAGE were embedded into the GAE framework, and each network was trained using the learning and validation data described in Section 3.1.2. For the HGNN network, the HGT framework was implemented, similarly training this network with the learning and validation data in Section 3.1.2.

Upon completion of training, the test data were inputted into the network. After obtaining the feature vectors for each node in the heterogeneous graph, feature reduction was performed according to Equation (16) to obtain the descriptive vector of the graph, which corresponds to the geometric relationship descriptive vectors of the various maps. After measuring the geometric similarity between two maps, the map with the largest scale in each region was taken as the baseline, and the similarity of other scale maps was calculated in relation to this baseline. Table 4 illustrates the geometric similarity measurement results across different scales for four typical regions using various methods.

Table 4. Geometric similarity measurement results at various scales for four typical regions using different methods. Note: SAGE is an abbreviation for GraphSAGE, with the same meaning used throughout the text. (The parameters in bold in the table represent the minimum values in the similarity comparison).

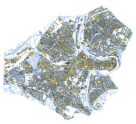
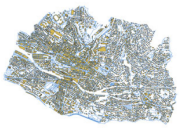
		1:10K	**1:15K**	**1:25K**	**1:50K**	**1:100K**	**1:250K**	**1:500K**
Bern								
HAN	GCN	1.0000	0.7767	0.7524	0.6996	0.6739	0.4573	0.2927
	GAT	1.0000	**0.7133**	**0.7328**	**0.6740**	**0.6528**	**0.3396**	**0.2147**
	SGAT	1.0000	0.8194	0.7906	0.7043	0.6882	0.5601	0.3093
	SAGE	1.0000	0.8883	0.8630	0.8588	0.8421	0.6823	0.5466

Table 4. *Cont.*

GAE	GCN	1.0000	0.9831	0.9407	0.8532	0.7611	0.5296	0.3492
	GAT	1.0000	0.9278	0.8226	0.7323	0.6843	0.3746	0.2843
	SGAT	1.0000	0.9344	0.8653	0.7973	0.7328	0.4513	0.3210
	SAGE	1.0000	0.8447	0.8009	0.7154	0.6959	0.5559	0.3474
HGT		1.0000	0.9920	0.9918	0.9891	0.9861	0.7792	0.8686
		1:15K	**1:25K**		**1:50K**	**1:150K**		**1:500K**
Aarau								
HAN	GCN	1.0000	**0.6972**		**0.5066**	0.3567		0.1181
	GAT	1.0000	0.7334		0.5442	**0.2805**		**0.0212**
	SGAT	1.0000	0.8179		0.7964	0.4972		0.2753
	SAGE	1.0000	0.6769		0.5999	0.3963		0.1048
GAE	GCN	1.0000	0.7518		0.5484	0.3988		0.0335
	GAT	1.0000	0.7661		0.6656	0.4438		0.3227
	SGAT	1.0000	0.8127		0.6272	0.5041		0.3242
	SAGE	1.0000	0.8337		0.7930	0.6897		0.2387
HGT		1.0000	0.9917		0.9296	0.7313		0.7211
		1:15K	**1:25K**		**1:50K**	**1:150K**		**1:500K**
Bulle								
HAN	GCN	1.0000	0.5484		0.3988	**0.1301**		0.1094
	GAT	1.0000	0.5133		**0.3429**	0.2532		**0.0722**
	SGAT	1.0000	**0.3933**		0.3769	0.2941		0.0843
	SAGE	1.0000	0.6542		0.5745	0.2981		0.1053
GAE	GCN	1.0000	0.6549		0.3964	0.2546		0.1867
	GAT	1.0000	0.4561		0.3206	0.3389		0.3871
	SGAT	1.0000	0.6127		0.3393	0.0640		0.1499
	SAGE	1.0000	0.5697		0.2070	0.1132		0.1820
HGT		1.0000	0.9913		0.9321	0.6915		0.6374
		1:10K	**1:15K**		**1:25K**	**1:50K**		**1:100K**
Lausanne								
HAN	GCN	1.0000	0.8260		0.8161	**0.6431**		**0.4781**
	GAT	1.0000	0.8406		0.8257	0.7472		0.5713
	SGAT	1.0000	0.8530		0.8331	0.7624		0.6583
	SAGE	1.0000	**0.8005**		**0.7604**	0.6619		0.5707
GAE	GCN	1.0000	0.8854		0.8620	0.6982		0.5658
	GAT	1.0000	0.8791		0.8606	0.7739		0.6751
	SGAT	1.0000	0.8892		0.8780	0.7421		0.6946
	SAGE	1.0000	0.8803		0.8559	0.7908		0.6831
HGT		1.0000	0.9975		0.9957	0.9776		0.8895

As shown in Table 4, the HGT method yields geometric similarity measurement results above 0.6 across all scales. Although the similarity measurement results in most evaluations exhibiting a downward trend, which aligns with objective cognitive principles, the outcomes from the HGT method are noticeably inconsistent with objective reality. For instance,

in the experimental results for the Lausanne region, both the GAE and HAN frameworks indicate that the geometric similarity lies between 0.45 and 0.70, whereas the HGT method reports a value exceeding 0.85. This is unrealistic, as comparing the original scale map with the map at a scale of 1:500K reveals a significant discrepancy; thus, a similarity of 0.8895 is evidently implausible, violating subjective spatial cognition principles.

The four methods under the HAN framework demonstrate an objective trend where similarity decreases as the scale decreases, with most measurement results being lower compared to other methods. Higher similarity is observed at larger scales, while lower similarity is noted at smaller scales. The similarity measurement results for GAE exhibit a pattern generally consistent with those of the HAN framework, showing a decline in similarity as the scale diminishes. However, GAE often reports excessively high similarity values for large-scale maps that diverge from actual conditions. For example, in the Bern region experiment, the GCN, GAT, and SGAT methods under the GAE framework suggest that the map at a scale of 1:15K remained more than 0.90, similar to the original map (at a scale of 1:10K), with the GCN method reporting a striking similarity of 0.9831, which clearly does not reflect objective reality. In contrast, most methods under the HAN framework provide realistic similarity calculations, indicating the effectiveness of this approach.

3.3.2. Subjective Evaluation

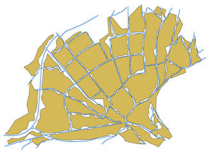
Human cognition of geometric shapes is a result of the integration of logical and visual thinking; therefore, the results of the similarity measurements must also align with human cognition, necessitating evaluation based on subjective human perceptions. The measurement of similarity in mixed scenes under scale transformation falls within the realm of spatial cognition related to maps. Due to differences in cognitive levels arising from educational backgrounds and professional fields, it is challenging to reach a consensus on such uncertain issues.

To obtain a professional evaluation of the method from individuals with a background in cartography, this study employed a questionnaire survey, inviting respondents to provide subjective assessments of the similarity measurement results from different methods. A total of 28 questionnaires were distributed, with 24 returned, of which 20 were valid. The participants included 8 males and 12 females, aged between 20 and 35 years. Among them, 15 individuals were professionals in the geographic information industry, while the remaining 5 were Master's or doctoral students in cartography, geographic information systems, cartography and geographic information engineering, or surveying engineering.

The qualitative comparison methods consist of three categories. The first category is the parallel model based on the HAN framework proposed in this study, which integrated four sub-GNN modules: GCN, GAT, SGAT, and GraphSAGE. The second category comprised methods based on the GAE framework, which similarly incorporated the four sub-modules: GCN, GAT, SGAT, and GraphSAGE. The third category consisted of methods using the HAN framework. All three categories of methods were trained using the same data, selecting the model with the minimum loss value on the validation set as the optimal model.

The questionnaire covered ten regions from the test set (detailed information for each region can be found in Table 2). As shown in the scene evaluation cases in Table 5 the questionnaire included two main aspects: the respondents' agreement with the similarity measurement results for the mixed scene (with options including "Agree," "Disagree," and "Unsure"). When evaluating the same region, respondents were required to select the method they believe would deliver the best evaluation. Note that a method could be selected as the best option a maximum of 200 times, meaning that all 20 respondents believed this method to be optimal across the 10 regions.

Table 5. Questionnaire case for evaluating similarity measurement of mixed scenes.

		Original Scale	Target Scale			
		1:10K	1:25K	1:50K	1:150K	1:500K
Bern						
HAN	GCN	1.0000	0.0345 ☐	0.7094 ☐	0.1532 ☐	0.0761 ☐
	GAT	1.0000	0.0140 ☐	0.6076 ☐	0.3637 ☐	0.3607 ☐
	SGAT	1.0000	0.0964 ☐	0.6829 ☐	0.4069 ☐	0.1649 ☐
	SAGE	1.0000	0.3245 ☐	0.7452 ☐	0.5135 ☐	0.3128 ☐
GAE	GCN	1.0000	0.0455 ☐	0.5244 ☐	0.3270 ☐	0.2822 ☐
	GAT	1.0000	0.0710 ☐	0.2915 ☐	0.2356 ☐	0.2068 ☐
	SGAT	1.0000	0.2268 ☐	0.0926 ☐	0.3495 ☐	0.1720 ☐
	SAGE	1.0000	0.0205 ☐	0.4139 ☐	0.2357 ☐	0.1548 ☐
HGT		1.0000	0.8927 ☐	0.9590 ☐	0.8049 ☐	0.7416 ☐

If you agree with this rating, please mark "√"; if you disagree, mark "×"; if you are unsure, mark "○".

Considering that respondents might have gradually adapted to the questionnaire, three regions were randomly selected for initial surveys and the questionnaires were collected. If there were significantly differing responses in the results, the answers from that questionnaire would be discussed to reach a consensus; responses that showed little variation would not be discussed. After respondents became more familiar with the questions, the order of the questions in the original questionnaire would be shuffled, and the questionnaires for the 10 scenarios would be reissued and collected.

According to the evaluation model described above, each evaluation method will yield 50 similarity measurement results (for a total of 10 scenarios, each containing 4 or 5 different scales), with 20 respondents assessing the unified quantitative results, resulting in 1000 evaluation outcomes for each method. Therefore, the satisfaction level for each method can be represented as Y/1000 (where Y is the number of times the method is rated as "satisfactory"). Additionally, the number of times a certain method's evaluation is deemed the optimal solution can also be statistically recorded, with results presented in Tables 6 and 7.

Table 6. Evaluation statistics of respondents on 9 methods applied to 10 scenarios.

Method		Agree	Disagree	Unsure	Satisfaction Rate
HAN	GCN	817	177	6	0.817
	GAT	797	197	6	0.797
	SGAT	754	241	5	0.754
	SAGE	802	186	8	0.805
GAE	GCN	414	582	4	0.414
	GAT	335	664	1	0.335
	SGAT	328	672	0	0.328
	SAGE	341	656	15	0.336
HGT		112	885	3	0.115

Table 7. Statistics of the number of times selected as the optimal method.

Method	HAN				GAE				HGT
	GCN	GAT	SGAT	SAGE	GCN	GAT	SGAT	SAGE	
Optimal Count	93	56	12	22	9	3	1	4	0

Combining the similarity measurement evaluation heatmap with the data from Tables 6 and 7, it is evident that the proposed parallel network model based on the HAN framework significantly outperforms both the HGT method and the GAE framework methods in terms of satisfaction rate and optimal count. Among the methods under the HAN framework, the GCN method performs the best (with a satisfaction rate of 0.817), followed by GraphSAGE and GAT. On the other hand, methods under the GAE framework have an average satisfaction rate of less than 0.4. Although GCN in this framework achieved a certain number of "optimal" ratings (9 times), its overall satisfaction rate is only about half that of the HAN framework methods, and the count of "optimal" ratings is less than one-tenth of those under the HAN framework. The HGT method has the lowest satisfaction rate, and the main issue with this method is its overestimation of similarity. For example, in the example provided in Table 4, the minimum and maximum scales exhibit considerable differences, yet this method still reports a similarity of up to 0.7, which clearly does not align with reality.

The superior performance of the proposed method can be attributed to two key aspects. First, the normalization module can eliminate dimensions, normalizing the selected geometric parameters into the same feature space while also eliminating offsets between features. Additionally, the four parallel GNN modules can extensively exploit spatial relationships, integrating the adjacency semantics between nodes into the node features. This enables the final generated reduced features to accurately describe the information of the heterogeneous graph, resulting in objective similarity measurement outcomes.

3.3.3. Discussion of Parallel Network Results

From the quantitative and subjective qualitative evaluations, it can be observed that the proposed method consistently provides objective evaluation results in the majority of cases. However, abnormal situations arose in the experimental results for the two regions shown in the table below. In Table 8, the areas marked in red indicate scenes where the vast majority of respondents disagreed with the quantitative results.

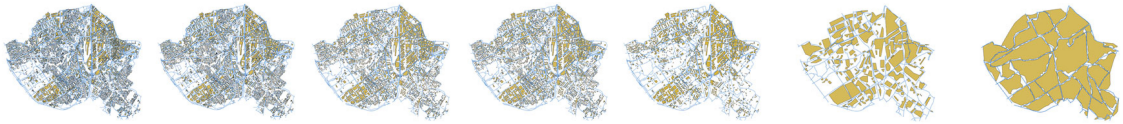
It can be observed that when measuring the similarity between the 1:15K topographic map and the original scale (1:10K) map, both the GAE and HAN framework methods yield similarity measurement values below 0.5, which clearly contradicts objective reality: there should be a high geometric similarity between maps with such a small difference in scale. However, when the scale is less than 1:25K, the similarity measurement results from the methods under the HAN framework return to normal.

Furthermore, an analysis of data results from other regions reveals that this type of error occurs only when measuring the similarity between 1:10K and 1:15K scale maps. When the target scale differs from the original scale by more than a factor of 2, these errors do not appear. Additionally, with the exception of the HGT algorithm, the algorithms under the other two frameworks exhibit similar performance.

Analysis of the data reveals that this issue is likely caused by the cartographic generalization method. When generalizing roads in maps, it is generally possible to abstract and summarize the features of the original scale map, meaning that road features in the map are gradually reduced. However, when generalizing buildings, a large number of new geometric features emerge. In the example given in Figure 9, the original scale map shows

a closed or semi-closed building, while after generalization, the building is represented as a complete polygonal geometric unit.

Table 8. Example of abnormal similarity evaluation results under the HAN framework.

		1:10K	1:15K	1:25K	1:50K	1:100K	1:250K	1:500K
Genève								
HAN	GCN	1.0000	0.0013	0.8076	0.7813	0.6211	0.3160	0.2990
	GAT	1.0000	0.0051	0.7861	0.7799	0.6418	0.4441	0.2884
	SGAT	1.0000	0.1234	0.8537	0.8313	0.7270	0.4731	0.3660
	SAGE	1.0000	0.3427	0.8863	0.8486	0.7299	0.5236	0.1589
GAE	GCN	1.0000	0.0701	0.9325	0.8605	0.5006	0.2494	0.0084
	GAT	1.0000	0.2575	0.7500	0.5556	0.2965	0.1801	0.1424
	SGAT	1.0000	0.0609	0.7854	0.7027	0.3204	0.2192	0.1172
	SAGE	1.0000	0.4959	0.8275	0.7376	0.0998	0.1434	0.2063
HGT		1.0000	0.9975	0.9957	0.9776	0.8895	0.8787	0.8940
		1:10K	**1:15K**	**1:25K**	**1:50K**	**1:100K**	**1:250K**	**1:500K**
Basel								
HAN	GCN	1.0000	0.0955	0.8728	0.8199	0.6105	0.2063	0.1898
	GAT	1.0000	0.0211	0.8398	0.7420	0.5810	0.2617	0.2766
	SGAT	1.0000	0.0920	0.8522	0.8001	0.7226	0.4552	0.4030
	SAGE	1.0000	0.2994	0.8709	0.8504	0.8321	0.5322	0.5071
GAE	GCN	1.0000	0.0274	0.9631	0.7765	0.6001	0.3436	0.3456
	GAT	1.0000	0.2029	0.8818	0.6022	0.5128	0.2379	0.1999
	SGAT	1.0000	0.3921	0.9069	0.7326	0.6404	0.0578	0.2845
	SAGE	1.0000	0.0499	0.8768	0.4869	0.1774	0.0071	0.0474
HGT		1.0000	0.9505	0.9979	0.9913	0.9534	0.8900	0.9173

When constructing features for the same geometric object at different scales using the parameters listed in Table 1, the differences in their parameters (such as the number of neighbors, orientation of the minimum bounding rectangle, aspect ratio of the minimum bounding rectangle, area, perimeter, etc.) are substantial. This leads to significant differences in the features of the same geometric object, meaning that the feature vectors constructed from these parameters can no longer adequately describe the geometric objects, resulting in incorrect similarity measurement outcomes.

On the other hand, from a visual perspective, in the example shown in Figure 9, although the scale difference is only 1.5 times, the generalization algorithm completely ignores the internal details of the building when faced with such a complex scenario. The degree of generalization is too large, causing the learning algorithm to mistakenly treat the target map as a large-scale map. As a result, the algorithm computes similarity based on the pattern for comparing large-scale and small-scale maps, leading to incorrect measurement results.

Figure 9. Cartographic generalization case. The first column shows the overlays of two scale maps; the second column presents the original scale map (1:10K); the third column displays the generalized map at a scale of 1:15K.

4. Conclusions and Future Work

Measuring the similarity between geographic objects lies the heart of geographical information science. While measuring the similarity of mixed scenes is still a challenging work since multiple types of geometric objects in the mixed scenes are heterogeneous. Traditionally non-deep-learning-based methods are not capable of dealing with heterogeneous data structures for it to typically concentrate on single-type features which originate from single scene. Deep-learning-based methods (such as GNN embedded GAE) are the alternative and have the ability to cope with heterogeneous data, while their performances are not desirable for the lack of expressivity in extracting features from different geometric objects. This paper proposed a HGNN-based framework which not only can handle heterogeneous data from mixed scenes but also surpass traditionally used methods in performance. The proposed framework can embed many other GNN modules in a parallel mode and thus is more expressive in compositing features from different types of geometric objects. Meanwhile, we conducted ablation experiments to show that normalization is a prerequisite for heterogeneous preprocessing, and without the parallel network, the training fails at a local optimum. We also selected road and building data from 10 regions in Switzerland for experimentation, and the results indicate that the proposed model effectively accomplished the task of evaluating map generalization quality in mixed scenes, presenting potential benefits for map quality assessment research.

The main drawback of the proposed method is that its performances may be influenced by the cartographic generalization methods, especially for the scenes that the new emerging geometric objects are generating from the generalization process. The issue may result from the self-supervised learning mode in which only affinity relationships are the constraints. If we give more supervised signals (i.e., the scales of two maps), this issue may be alleviated. In the future, we will present a semi-supervised training mode on the proposed framework and try to give a thoroughly explanation on how and why the proposed HGNN-based model works.

Author Contributions: Conceptualization, Tinghua Ai; data curation, Chongya Gong; formal analysis, Chongya Gong and Shiyu Chen; funding acquisition, Tinghua Ai; methodology, Chongya Gong; software, Chongya Gong, Tinghua Ai, Shiyu Chen and Tianyuan Xiao; supervision, Tinghua Ai and Huafei Yu; validation, Chongya Gong, Shiyu Chen and Tianyuan Xiao; writing—original draft, Chongya Gong; writing—review and editing, Tinghua Ai, Tianyuan Xiao and Huafei Yu. All authors have read and agreed to the published version of the manuscript.

Funding: This article was supported by the National Natural Science Foundation of China [grant number 42394065]; the Fundamental Research Funds for the Central Universities, China (Grant No. 2042022dx0001).

Data Availability Statement: The data that support the findings of this study are available with a DOI at https://doi.org/10.6084/m9.figshare.28151696 (Accessed on 7 January 2025).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Austin, R.F. Measuring and Comparing Two-Dimensional Shapes. In *Spatial Statistics and Models*; Springer: Dordrecht, The Netherlands, 1984; pp. 293–312.
2. Bruns, H.T.; Egenhofer, M. Similarity of spatial scenes. In Proceedings of the Seventh International Symposium on Spatial Data Handling, Delft, The Netherlands, 12–16 August 1996; pp. 31–42.
3. Zhao, Z.; Stough, R.R. Measuring similarity among various shapes based on geometric matching. *Geog. Anal.* **2005**, *37*, 410–422. [CrossRef]
4. Yu, H.; Ai, T.; Yang, M.; Huang, W.; Harrie, L. A graph autoencoder network to measure the geometric similarity of drainage networks in scaling transformation. *Int. J. Digit. Earth* **2023**, *16*, 1828–1852. [CrossRef]
5. Hartley, R.; Zisserman, A. *Multiple View Geometry in Computer Vision*; Cambridge University Press: Cambridge, UK, 2003.
6. Szeliski, R. *Computer Vision: Algorithms and Applications*; Springer Nature: Berlin/Heidelberg, Germany, 2022.
7. Rui, Y.; Huang, T.S.; Chang, S.F. Image retrieval: Past, present, and future. *J. Vis. Commun. Image Represent.* **1999**, *10*, 1–23.
8. Jiang, X.; Ma, J.; Xiao, G.; Shao, Z.; Guo, X. A review of multimodal image matching: Methods and applications. *Inf. Fusion* **2021**, *73*, 22–71. [CrossRef]
9. Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
10. Villani, C. *Optimal Transport: Old and New*; Springer: Berlin/Heidelberg, Germany, 2009.
11. Hedrick, P.W. A new approach to measuring genetic similarity. *Evolution* **1971**, *25*, 276–280. [CrossRef]
12. Yan, H. Description and Generalization of River Networks. In *Description Approaches and Automated Generalization Algorithms for Groups of Map Objects*; Springer: Singapore, 2019; pp. 109–148.
13. Liu, P.; Huang, X.; Ma, H.; Yang, M. Fourier Descriptor-based neural network method for high-precision shape recognition of building polygon. *Acta Geod. Cartogr. Sin.* **2022**, *51*, 1969.
14. Yan, H. Quantifying spatial similarity for use as constraints in map generalisation. *J. Spat. Sci.* **2024**, *69*, 23–42. [CrossRef]
15. Jacobson, J.Z.; Rhinelander, G. Geometric and semantic similarity in visual masking. *J. Exp. Psychol. Hum. Percept. Perform.* **1978**, *4*, 224. [CrossRef]
16. Yan, H. Theory of Spatial Similarity Relations and Its Applications in Automated Map Generalization. Ph.D. Thesis, University of Waterloo, Waterloo, ON, Canada, 2014.
17. Ai, T.; Shuai, Y.; Li, J. A spatial query based on shape similarity cognition. *Acta Geod. Cartogr. Sin.* **2009**, *38*, 356–362.
18. Yan, X.; Ai, T.; Yang, M.; Tong, X. Graph convolutional autoencoder model for the shape coding and cognition of buildings in maps. *Int. J. Geogr. Inf. Sci.* **2021**, *35*, 490–512. [CrossRef]

19. Sadahiro, Y.; Liu, Y. A scale-sensitive approach for comparing and classifying point patterns. *J. Spat. Sci.* **2020**, *65*, 281–306. [CrossRef]
20. Zhang, Y.; Yu, W.; Chen, Z. An improved method for generalisation of point features with consideration of reinforcing relationships. *J. Spat. Sci.* **2022**, *67*, 41–60. [CrossRef]
21. Xiao, T.; Ai, T.; Yu, H.; Yang, M.; Liu, P. A point selection method in map generalization using graph convolutional network model. *Cartogr. Geogr. Inf. Sci.* **2023**, *51*, 20–40. [CrossRef]
22. Yu, H.; Ai, T.; Yang, M.; Huang, L.; Yuan, J. A recognition method for drainage patterns using a graph convolutional network. *Int. J. Appl. Earth Obs. Geoinf.* **2022**, *107*, 102696. [CrossRef]
23. Liu, P.; Li, X.; Liu, W.; Ai, T. Fourier-based multi-scale representation and progressive transmission of cartographic curves on the internet. *Cartogr. Geogr. Inf. Sci.* **2016**, *43*, 454–468. [CrossRef]
24. Yan, X.; Ai, T.; Yang, M.; Yin, H. A graph convolutional neural network for classification of building patterns using spatial vector data. *ISPRS J. Photogramm. Remote Sens.* **2019**, *150*, 259–273. [CrossRef]
25. Steiniger, S.; Burghardt, D.; Weibel, R. Recognition of island structures for map generalization. In Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems, Arlington, Virginia, 10–11 November 2006; pp. 67–74.
26. Zhang, C.; Song, D.; Huang, C.; Swami, A.; Chawla, N.V. Heterogeneous graph neural network. In Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, Anchorage, AK, USA, 4–8 August 2019; pp. 793–803.
27. Hoffman, W.C. Subjective geometry and geometric psychology. *Math. Model.* **1980**, *1*, 349–367. [CrossRef]
28. Gal, R.; Cohen-Or, D. Salient geometric features for partial shape matching and similarity. *ACM Trans. Graph. (TOG)* **2006**, *25*, 130–150. [CrossRef]
29. Zhang, Y.; Yu, W. Detecting common features from point patterns for similarity measurement using matrix decomposition. *Cartogr. Geogr. Inf. Sci.* **2024**, *51*, 462–485. [CrossRef]
30. Boureau, Y.L.; Ponce, J.; LeCun, Y. A theoretical analysis of feature pooling in visual recognition. In Proceedings of the 27th International Conference on Machine Learning (ICML-10), Haifa, Israel, 21–24 June 2010; pp. 111–118.
31. Wu, Z.; Pan, S.; Chen, F.; Long, G.; Zhang, C.; Philip, S.Y. A comprehensive survey on graph neural networks. *IEEE Trans. Neural Netw. Learn. Syst.* **2020**, *32*, 4–24. [CrossRef] [PubMed]
32. Khan, B.; Wu, J.; Yang, J.; Ma, X. Heterogeneous hypergraph neural network for social recommendation using attention network. *ACM Trans. Recomm. Syst.* **2023**. [CrossRef]
33. Yan, B. Using Heterogeneous Graph Neural Networks (hGNN) to Predict Cell-Cell Communication. Ph.D. Thesis, Massachusetts Institute of Technology, Cambridge, MA, USA, 2024.
34. Wang, X.; Ma, W.; Guo, L.; Jiang, H.; Liu, F.; Xu, C. HGNN: Hyperedge-based graph neural network for MOOC course recommendation. *Inf. Process. Manag.* **2022**, *59*, 102938. [CrossRef]
35. Li, D.; Shi, G.; Wu, Y.; Yang, Y.; Zhao, M. Multi-scale neighborhood feature extraction and aggregation for point cloud segmentation. *IEEE Trans. Circuits Syst. Video Technol.* **2020**, *31*, 2175–2191. [CrossRef]
36. Pei, H.; Wei, B.; Chang, K.C.C.; Lei, Y.; Yang, B. Geom-gcn: Geometric graph convolutional networks. *arXiv* **2020**, arXiv:2002.05287.
37. Lux, L.; Berger, A.H.; Weers, A.; Stucki, N.; Rueckert, D.; Bauer, U.; Paetzold, J.C. Topograph: An efficient Graph-Based Framework for Strictly Topology Preserving Image Segmentation. *arXiv* **2024**, arXiv:2411.03228.
38. Wang, G.; Ying, R.; Huang, J.; Leskovec, J. Multi-hop attention graph neural network. *arXiv* **2020**, arXiv:2009.14332.
39. Meng, C.; Cheng, R.; Maniu, S.; Senellart, P.; Zhang, W. Discovering Meta-Paths in Large Heterogeneous Information Networks. In Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015; pp. 754–764.
40. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
41. Veličković, P.; Cucurull, G.; Casanova, A.; Romero, A.; Lio, P.; Bengio, Y. Graph attention networks. *arXiv* **2017**, arXiv:1710.10903.
42. Ye, Y.; Ji, S. Sparse graph attention networks. *IEEE Trans. Knowl. Data Eng.* **2021**, *35*, 905–916. [CrossRef]
43. Hamilton, W.; Ying, Z.; Leskovec, J. Inductive representation learning on large graphs. *Adv. Neural Inf. Process. Syst.* **2017**, *30*.
44. Hu, Z.; Dong, Y.; Wang, K.; Sun, Y. Heterogeneous graph transformer. In Proceedings of the Web Conference, Taipei Taiwan, 20–24 April 2020; pp. 2704–2710.
45. Wang, X.; Ji, H.; Shi, C.; Wang, B.; Ye, Y.; Cui, P.; Yu, P.S. Heterogeneous graph attention network. In Proceedings of the World Wide Web Conference, San Francisco, CA, USA, 13–17 May 2019; pp. 2022–2032.
46. Huang, X.; Belongie, S. Arbitrary style transfer in real-time with adaptive instance normalization. In Proceedings of the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 1501–1510.
47. Waoo, A.A.; Soni, B.K. Performance analysis of sigmoid and relu activation functions in deep neural network. In *Intelligent Systems: Proceedings of SCIS 2021*; Springer: Singapore, 2021; pp. 39–52.
48. Liu, W.; Wen, Y.; Yu, Z.; Yang, M. Large-margin softmax loss for convolutional neural networks. *arXiv* **2016**, arXiv:1612.02295.
49. Ioffe, S. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv* **2015**, arXiv:1502.03167.
50. Merris, R. Laplacian matrices of graphs: A survey. *Linear Algebra Appl.* **1994**, *197*, 143–176. [CrossRef]

51. Shuman, D.I.; Vandergheynst, P.; Frossard, P. Chebyshev polynomial approximation for distributed signal processing. In Proceedings of the International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS), Barcelona, Spain, 27–29 June 2011; IEEE: Piscataway, NJ, USA, 2011; pp. 1–8.
52. Shuman, D.I.; Vandergheynst, P.; Kressner, D.; Frossard, P. Distributed signal processing via Chebyshev polynomial approximation. *IEEE Trans. Signal Inf. Process. Netw.* **2018**, *4*, 736–751. [CrossRef]
53. Ying, Z.; You, J.; Morris, C.; Ren, X.; Hamilton, W.; Leskovec, J. Hierarchical graph representation learning with differentiable pooling. *Adv. Neural Inf. Process. Syst.* **2018**, *31*.
54. Ai, T. Maps adaptable to represent spatial cognition. *Natl. Remote Sens. Bull.* **2008**, *12*, 347–354.
55. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Chintala, S. Pytorch: An imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **2019**, *32*.
56. Bielak, P.; Kajdanowicz, T.J. Pytorch-geometric edge-a library for learning representations of graph edges. In Proceedings of the The First Learning on Graphs Conference, Virtual, 9–12 December 2022.
57. Kipf, T.N.; Welling, M. Variational graph auto-encoders. *arXiv* **2016**, arXiv:1611.07308.