Geometrically Invariant and Equivariant Graph Neural Networks for TSP Algorithm Selection and Hardness Prediction

Ya Song$^{1[0000-0001-6378-2212]}$ and Laurens Bliek$^{1[0000-0002-3853-4708]}$ Yingqian Zhang$^{1[0000-0002-5073-0787]}$

Eindhoven University of Technology, Eindhoven 5612 AZ, Netherlands
{y.song,l.bliek,yqzhang}@tue.nl

Abstract. Algorithm selection and instance hardness prediction are important tasks in combinatorial optimization, as different algorithms perform optimally on different instances, and accurately predicting instance difficulty enables more efficient problem-solving strategies. While researchers have explored feature-based machine learning models for these tasks, there remains a lack of general and effective instance representations, even for fundamental problems like the Euclidean Traveling Salesman Problem (TSP). Recent studies have leveraged Convolutional Neural Networks (CNNs) to learn TSP representations, but they still require intermediate image-based representations, introducing additional preprocessing steps. We tackle algorithm selection and hardness prediction problems for TSP, but instead treat instances as geometric graphs. We propose four Geometrically Invariant and Equivariant Graph Neural Networks (GIE-GNNs), based on conventional GNN models. The proposed GIE-GNNs incorporate a novel message-passing mechanism to capture geometric information effectively. Evaluations on two algorithm selection and two hardness prediction datasets demonstrate that our GNNs outperform feature-based, CNN-based, and standard GNN approaches designed for non-geometric graphs and point clouds. Moreover, our analysis highlights that our GNNs uniquely achieve geometrically invariant predictions. Code and datasets are available at GitHub.

Keywords: Traveling Salesperson Problem · Algorithm Selection · Instance Hardness Prediction · Graph Representation Learning · Graph Neural Network · Geometrically Equivariant · Geometrically Invariant.

1 Introduction

The Euclidean Traveling Salesman Problem (TSP) is one of the most intensely studied NP-hard combinatorial optimization problems. It relates to many real-world applications and has significant theoretical value. TSP can be described as follows. Given a list of cities with known positions, find the shortest route to visit each city and return to the origin city. Researchers have developed various exact, heuristic, and learning-based algorithms to solve this routing problem [1]. As these algorithms' performance is highly variable depending on the characteristics of the problem instances, predicting the algorithms' performance and selecting algorithms for each instance helps to improve the overall efficiency [2]. The algorithm selection problem was proposed in [3] and developed further in Instance Space Analysis (ISA) [4,5], where it is framed as a classification task establishing mappings between Problem Space and Algorithm Space. Researchers also utilize ISA to investigate instance hardness, employing a process consistent with algorithm selection [6]. Traditionally, domain experts have hand-engineered features [7] to capture the characteristics of TSP instances. A machine learning classifier is then trained on these features to select the most suitable algorithm. However, this feature-based approach suffers from several drawbacks [8]: it demands extensive domain expertise, the features themselves may be insufficiently expressive, and a separate feature selection step is often required. Moreover, handcrafted features do not transfer well to other optimization problems, and for complex or less-studied problems than TSP, it becomes especially challenging for humans to design effective features in the first place.

Deep learning models, especially Convolutional Neural Networks (CNN), have recently been applied to select TSP algorithms. By converting TSP instances into images and applying CNN to recognize them, the algorithm selection problem is transformed into a computer vision problem. Since CNN has sufficient automatic feature learning capability, this approach no longer requires handcrafted features. In [8], the authors generate three images: a point image, a Minimum Spanning Tree (MST) image, and a k-Nearest-Neighbor-Graph (kNNG) image, to represent each TSP instance. Then an 8-layer CNN architecture was applied to predict which algorithm is better. The authors of [9] use a gridding method to transform TSP instances into density maps and then apply ResNet [10] to do the prediction. Similarly, [2] applies a CNN model to predict algorithms' temporal performance at different time steps.

Although [2,8,9] show CNNs outperform traditional handcrafted feature-based ML models for TSP algorithm selection, the CNN based approaches have the following drawbacks. (1) *They need to generate intermediate representations (images) from the instances.* This step is often labor-intensive. In [8], producing MST and kNNG images for each instance requires significant computation, while in [9], multiple upscaling steps are performed to improve image resolution. Moreover, data augmentation techniques such as random rotations and flips are widely used to enhance CNN generalization [2,8,9], meaning that multiple images must be generated for each TSP instance. (2) *They may introduce problem-irrelevant parameters.* In [8], cities are represented by solid dots and connected by solid lines in MST and kNNG images, however, the dot size and line width are unrelated to the inherent properties of the TSP instance. Similarly, when using a gridding method, one must specify parameters such as image size or number of grids [9], thereby adding complexity and increasing the burden of parameter tuning. (3) *They potentially lose problem-relevant information.* Dividing a TSP instance into grids, with each cell reflecting the number of cities it includes [2,9], can lead to the loss of certain local structural details. Moreover, imposing a maximum value for each grid cell [2] introduces further distortion and reduces the fidelity of the representation. (4) *Their generalizability to other routing problems could be limited.* It is relatively straightforward to convert TSP instances into images in a 2D Euclidean space. However, for variants such as the Asymmetric TSP (ATSP) or the Vehicle Routing Problem (VRP), this procedure becomes much less direct. In these cases, a graph-based representation with node and edge features is often more suitable than an image-based approach.

To address the above limitations of CNN-based approaches, we propose to represent TSP instances as geometric graphs and employ Graph Neural Networks (GNNs) to address both algorithm selection and instance hardness prediction. We integrate geometric invariance and equivariance into GNN to robustly capture the underlying geometric information. Our end-to-end framework directly processes cities' coordinates and pairwise distances, ensuring invariance to rotations and translations in the 2D Euclidean space. Our main contributions are:

- We design geometrically invariant and equivariant GNNs (GIE-GNN for short) to learn the representation of TSP instances for algorithm selection and hardness prediction, outperforming the existing feature-based and CNN-based approaches on public TSP datasets.
- With the adopted graph representation, we show that there is no need to generate intermediate representations, such as handcrafted features or images.
- The proposed GNNs not only exceed the performance of traditional GNNs on non-geometric graphs and point clouds but also demonstrate the capability to offer geometrically invariant predictions. This makes them more reliable and robust, as shown in our analysis.

2 GIE-GNNs as TSP algorithm selector

The TSP algorithm selection problem is defined as follows: given a TSP instance set $I = \{I_1, I_2, ..., I_l\}$, a TSP algorithm set $A = \{A_1, A_2, ..., A_m\}$, and a certain algorithm performance metric, the goal is to identify a per-instance mapping from I to A that maximizes its performance on I based on the given metric. A related task, instance hardness prediction, is a special case where the algorithm space consists of a single algorithm and each instance is labeled as either easy or hard. Traditionally, TSP instances have been represented by handcrafted features or images that serve as inputs to supervised models like SVMs and CNNs. In contrast, we use geometric graphs $G = (S, P, E)$ to represent TSP instances, where S are scale node features invariant to rotation or translation, P contains the geometric vectors (coordinates) of nodes, and E are edges computed via the k-Nearest Neighbor method. Given a set of TSP graphs $\{G_1, G_2, ..., G_l\}$ and their algorithm performance labels $\{y_1, y_2, ..., y_l\}$, the task of selecting TSP algorithms can be converted to a graph-level classification task.

We propose four GIE-GNN models: GINE-GI, MP-GI, GINE-GE, and MP-GE, by adapting from conventional GNN architectures and standard message passing mechanisms.

Design Geometrically Invariant Node Features In contrast to previous work [11] that processes TSP instances as non-geometric graphs by treating city coordinates as node features, our GNNs treat coordinates as positional vectors while restricting node features to scalar values, preserving invariance under geometric transformations. This distinction ensures that geometric attributes propagate via equivariant operations rather than contaminating invariant feature channels. Since TSP nodes inherently lack distinguishable attributes beyond coordinates, we design scalar features for our GNN architectures to preserve geometric invariance. Positional Encoding enhances graph representation learning by embedding node spatial relationships through trainable geometric priors, with Random Walk Positional Encoding (RWPE) [12] particularly constructing invariant node representations for TSP geometric graphs via k-step return probabilities:

$$s_i = \left[\mathrm{RW}_{ii}^{(1)},\ \mathrm{RW}_{ii}^{(2)},\ \ldots,\ \mathrm{RW}_{ii}^{(k)} \right] \in \mathbb{R}^k \tag{1}$$

where k denotes the preset maximum walk steps, and $\mathrm{RW}_{ii}^{(t)}$ represents the t-step return probability from node i to itself with $t \in \{1, ..., k\}$. Once a geometric graph is constructed, the relationships between points become fixed. Since the edge indices remain unchanged under rotations or translations, the positional encodings derived from random walks on these edges are invariant. These geometrically invariant features thus serve as robust inputs for our proposed GNNs.

Geometrically Invariant GINE (GINE-GI) We adapt the conventional GINE model [13] to be geometrically invariant. Our strategy involves assigning both a scalar feature and geometric vector (coordinates) to each node, while ensuring we propagate only the invariant scalar features of nodes and edges. The invariant scalar features of edges are the relative distances between nodes. The message-passing formulation of GINE-GI is shown as follows:

$$s_i^{(\ell+1)} = \mathrm{MLP}\left((1+\epsilon) \cdot s_i^{(\ell)} + \mathit{aggregate}_{j \in \mathcal{N}(i)} \mathrm{ReLU}\left(s_j^{(\ell)} + \|\mathbf{p}_j - \mathbf{p}_i\| \right) \right), \tag{2}$$

where $s_i^{(\ell)}$ is the scalar node features, with the initial features $s_i^{(0)} = s_i$ derived from the RWPE encoding defined in Eq. (1). $\|\mathbf{p}_j - \mathbf{p}_i\|$ is the scalar edge features and is equal to the Euclidean distance between the two connected nodes, and *aggregate* indicates the selected aggregator: it can be either the MAX aggregator or the SUM aggregator.

Geometrically Invariant Message-Passing (MP-GI) Similar to the setting of GINE-GI, we construct another geometrically invariant GNN with a different powerful message-passing mechanism. In this new message-passing approach, instead of adding node features and edge features to compute the message as in GINE, we use a simple concatenation operation to aggregate information and employ an MLP to project the concatenated features into a unified feature space. Concatenation allows for a richer representation of the combined node and edge features compared to simple addition. Besides, by using an MLP after concatenation, the model gains more flexibility in transforming and learning complex relationships between node and edge features. The message-passing formulation of MP-GI is shown as follows:

$$s_i^{(\ell+1)} = \mathrm{MLP}\left(\mathrm{Concat}\left(s_i^{(\ell)}, \mathit{aggregate}_{j\in\mathcal{N}(i)}\mathrm{Concat}(s_i^{(\ell)}, s_j^{(\ell)}, \|\mathbf{p}_j - \mathbf{p}_i\|)\right)\right), \tag{3}$$

where Concat is concatenation operation. The first Concat combines the node and edge features to provide a richer representation, and the second one concatenates the feature of node i with the aggregated features from its neighbors.

Geometrically equivariant Message-Passing (MP-GE) In [14], the authors designed EGNN and demonstrated that it is geometrically equivariant. The EGNN aggregates and updates node embeddings and performs geometrically equivariant updates on nodes' coordinates. Our MP-GE framework is built upon the foundations of EGNN. However, it diverges from EGNN in three significant aspects. First, we calculate the Euclidean distance between nodes for message construction instead of using the squared relative distance. This decision is informed by the fact that Euclidean distance is better suited to the characteristics of TSP instances and results in smoother gradients during training, which enhances the model's numerical stability. Second, we adopt a different aggregator to gather task-relevant information from the graph. Finally, we employ a distinct information integration method, for example, updating node features by concatenating the current features with the aggregated message, allowing for more flexible information propagation. The formulation of MP-GE is as follows:

$$\mathbf{m}_{i,j} = \mathrm{MLP}_m\left(\mathrm{Concat}\left(\mathbf{s}_i^{(\ell)}, \mathbf{s}_j^{(\ell)}, \|\mathbf{p}_j - \mathbf{p}_i\|\right)\right), \tag{4}$$

$$\mathbf{m}_i = \mathrm{aggregate}_{i\neq j}\, \mathbf{m}_{i,j}\,, \tag{5}$$

$$\mathbf{p}_i^{(\ell+1)} = \mathbf{p}_i^{(\ell)} + \sum_{i\neq j}(\mathbf{p}_j - \mathbf{p}_i)\cdot \mathrm{MLP}\left(\mathbf{m}_{i,j}\right), \tag{6}$$

$$\mathbf{s}_i^{(\ell+1)} = \mathrm{MLP}_u\left(\mathrm{Concat}\left(\mathbf{s}_i^{(\ell)}, \mathbf{m}_i\right)\right). \tag{7}$$

The message-passing mechanism in MP-GE involves both message aggregation and geometric updates of the scalar node features and coordinates. Eq. (4) defines the message passed between two nodes, computed via an MLP using concatenated node features and their Euclidean distance. Eq. (5) formalizes the message aggregation process via permutation-invariant operators. For coordinate updates, Eq. (6) applies a weighted sum of relative positions with neighbors, where the weights are learned via an MLP from the message. Meanwhile, Eq. (7) updates node scale features by concatenating current features with the aggregated message, processed by an MLP for the next layer's features. Together, these equations equip MP-GE with the ability to capture spatial structures while preserving equivariance to rotations and translations, ensuring robust feature learning in geometric graphs.

Geometrically equivariant GINE (GINE-GE) We adopt the framework of MP-GE and replace the message-passing part with GINE. The formulation of GINE-GE is shown as follows:

$$\mathbf{m}_{i,j} = \mathrm{ReLU}\left(\mathbf{s}_j^{(\ell)} + \|\mathbf{p}_j - \mathbf{p}_i\|\right), \tag{8}$$

$$\mathbf{m}_i = aggregate._{i \neq j}\mathbf{m}_{i,j}, \tag{9}$$

$$\mathbf{p}_i^{(\ell+1)} = \mathbf{p}_i^{(\ell)} + \sum_{i \neq j} (\mathbf{p}_j - \mathbf{p}_i) \cdot \mathrm{MLP}(\mathbf{m}_{i,j}), \tag{10}$$

$$\mathbf{s}_i^{(\ell+1)} = \mathrm{MLP}\left(\mathbf{s}_i^{(\ell)} + \mathbf{m}_i\right). \tag{11}$$

In contrast to MP-GE, where the message is derived from the concatenation of both nodes' scalar features $\mathbf{s}_i^{(\ell)}$ and $\mathbf{s}_j^{(\ell)}$ along with their positional distance and processed by an MLP, GINE-GE simplifies the message to the sum of the neighboring node's scalar features $\mathbf{s}_j^{(\ell)}$ and the distance between their positions, followed by a ReLU activation. This simplification avoids the use of an MLP at this step and focuses on the influence of the neighboring information and positional distance directly. Besides, GINE-GE directly sums the current state $\mathbf{s}_i^{(\ell)}$ with the aggregated message $\mathbf{m}_i$ before passing it through an MLP. This results in a simpler and more direct update rule, potentially reducing the computational complexity associated with the concatenation operation.

After adopting geometrically invariant node features and the message-passing procedure, we construct a lightweight GNN architecture as shown in Figure 1. In Geometrically equivariant GNNs, although geometric vectors P are not directly incorporated into the final classifier, their role in maintaining geometric equivariance and supporting the message passing process indirectly improves the feature representation of scalars, thereby impacting the final prediction result. We adopt two GNN layers to extract salient information from TSP graphs and apply PairNorm [15] after each GNN layer to address the over-smoothing problem. Then we apply the Jumping Knowledge strategy [16], which means concatenating each layer's graph embedding and feeding it into the following graph-level pooling layer. We employ the Global Maximum Pooling technique here, as it is proficient in capturing and preserving the most prominent features, remaining unaffected by variations in point density. This enables us to maximize the use of the representation acquired across the two GNN layers.

3 Experiments

We evaluate our GNNs on four TSP datasets: two for algorithm selection and two for hardness prediction. All datasets derive from ISA [17] and CNN-based

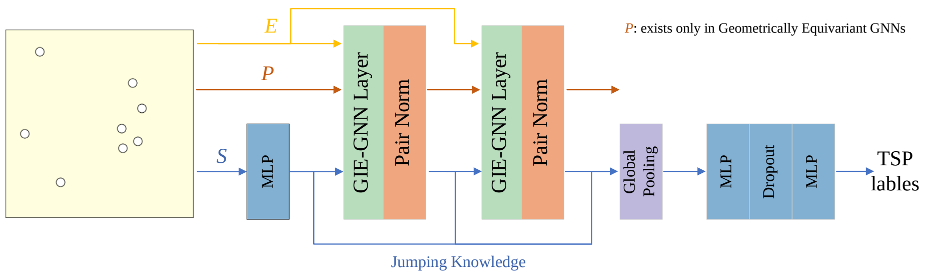

Fig. 1. The proposed GIE-GNN architecture

selection studies [8], differing in instance size (100 vs. 1000 cities) and target algorithms.

Algorithm Selection Datasets: **TSP-ISA** (100-city) contains 1,330 instances categorized into seven CLK/LKCC performance groups. The binary task predicts which algorithm outperforms per instance. The Single-Best-Solver (LKCC) achieves 71.43% accuracy; we balance the imbalanced training set via oversampling. **TSP-CNN** (1000-city) includes 1,000 instances comparing EAX and LKH algorithms. With a balanced distribution, the Single-Best-Solver (EAX) reaches only 49% accuracy, indicating comparable algorithm competitiveness.

Hardness Prediction Datasets: We construct two hardness prediction datasets from the TSP-ISA collection by redefining labels according to algorithm-specific difficulty criteria. The CLK hardness dataset contains 760 balanced instances, with classes defined as follows: Easy instances combine the original CLKeasy and easyCLK-hardLKCC groups, while hard instances comprise CLKhard and hardCLK-easyLKCC categories. Similarly, the LKCC hardness dataset consists of 760 balanced samples where easy instances merge LKCCeasy with hardCLK-easyLKCC, and hard instances correspond to LKCChard combined with easyCLK-hardLKCC.

3.1 Baseline model

In addition to comparing with the model proposed in other articles, we create several baseline models to be TSP selectors. For traditional feature-based models, we adopt Random Forest (RF) as the classifier following its proven effectiveness in TSP algorithm selection [8]. In our experiments, we set the maximum tree depth to 5, a configuration that balances performance and overfitting risk on our datasets. We evaluate four feature groups extracted via the salesperson R package [7]: (1) **All140**: Complete set of 140 features categorized into 10 groups, including Minimum Spanning Tree (MST), kNNG, and Angle features. (2) **Top15**: Optimal 15 features identified through feature selection for the TSP-CNN dataset [8], predominantly comprising kNNG connectivity statistics along with MST/Angle

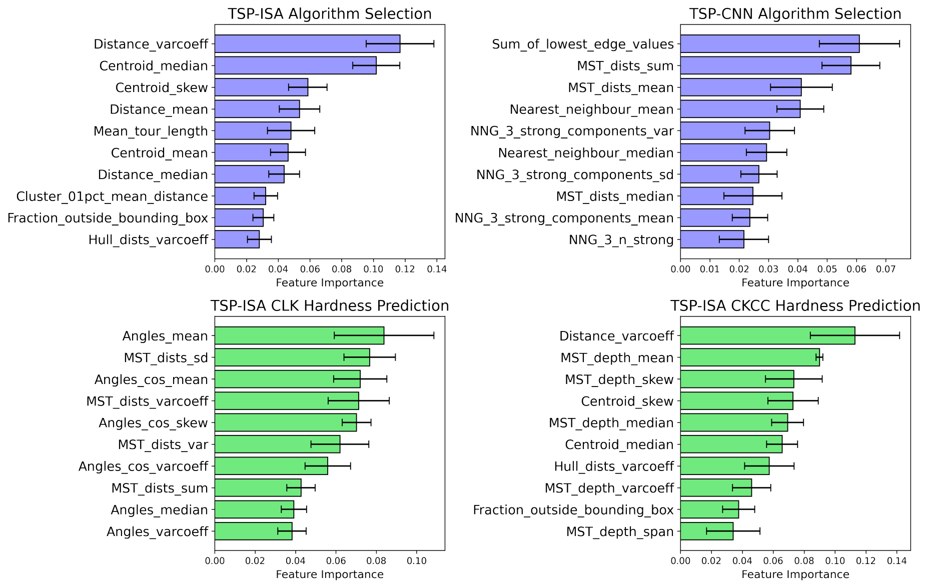

Fig. 2. Top 10 importance features for algorithm selection datasets (TSP-ISA and TSP-CNN), hardness prediction datasets (TSP-ISA CLK and TSP-ISA LKCC).

features. (3) **MST19**: All 19 MST-related features capturing distance/depth statistics. MST is prioritized due to its established theoretical connection to TSP approximation [8]. (4)**kNNG51**: Comprehensive 51-feature set describing kNNG distance distributions and connectivity components.

We evaluate two categories of baseline GNNs for TSP algorithm selection: (1) GCN [18] and GINE [13] for non-geometric graphs, and (2) PointNet [19] and DGCNN [20] for point clouds. To our knowledge, this is the first successful application of these GNNs to TSP algorithm selection. All models share the same architecture and parameters, differing only in GNN layer types. Following prior work [17,8], we adopt identical dataset processing methods. For evaluation, we use 10-fold cross-validation on TSP-ISA and TSP-CNN (replicating the original grouping), and 5-fold cross-validation for CLK/LKCC hardness prediction. All datasets are balanced, with classification accuracy as the metric. Models are trained for 150 epochs using the Adam optimizer (learning rate 0.001) and Cross Entropy loss. TSP instances are represented via kNNG (k=10) to balance computational efficiency. Random seeds are fixed to 41 for reproducibility [11].

3.2 Results of algorithm selection and hardness prediction

Table 1 presents the classification accuracy on the TSP-ISA algorithm selection dataset. Random forest (RF) with full 140 features achieves the best performance among feature-based methods, while feature reduction significantly degrades

Table 1. Overall performance comparison on the TSP-ISA algorithm selection dataset (bold: best; underline: runner-up).

Category	Models	Input data	Parameters	Accuracy
Feature-based models	RF	All140 features	–	$95.94 \pm 1.88\%$
		Top15 features		$87.22 \pm 2.54\%$
		MST19 features		$87.82 \pm 2.54\%$
		kNNG51 features		$74.21 \pm 3.69\%$
Standard GNNs	GCN	Non-geometric graphs	1,122	$97.52 \pm 1.26\%$
	GINE	Non-geometric graphs	1,266	$98.65 \pm 0.94\%$
Point cloud GNNs	PointNet	Point clouds	962	$\underline{99.33 \pm 0.53}\%$
	DGCNN	Point clouds	6,370	$99.03 \pm 0.48\%$
Proposed GIE-GNNs	GINE-GI	Geometric graphs	1,266	$98.87 \pm 0.97\%$
	MP-GI	Geometric graphs	2,546	$99.25 \pm 0.75\%$
	GINE-GE	Geometric graphs	1,622	$\mathbf{99.48 \pm 0.59}\%$
	MP-GE	Geometric graphs	2,614	$99.17 \pm 1.03\%$

accuracy, particularly revealing that MST features exhibit greater importance than kNNG features. This observation aligns with Figure 2, where TSP-ISA and TSP-CNN show markedly different key feature distributions, justifying the necessity of automatic feature learning in deep models. All GNNs surpass manual feature-based approaches through automatic feature extraction. While standard GNNs (GCN, GINE) designed for non-geometric graphs relatively underperform, point cloud-based architectures like PointNet and DGCNN excel by leveraging point coordinates to discern TSP instance patterns. Our geometrically invariant and equivariant GNNs further advance performance, with GINE-GE achieving state-of-the-art results. This underscores the importance of incorporating geometric awareness into model design. Specifically, these models achieve superior accuracy without TSP-specific domain knowledge, demonstrating their potential as promising approaches to TSP algorithm selection. TSP-CNN outcomes presented in Table 2 indicate that RF with MST features consistently outperforms its kNNG-based alternatives. Although CNN with Points+MST images outperforms other vision-based approaches, it lags behind RF. Notably, point cloud-based GNNs underperform compared to standard GNNs here due to larger instance complexity, yet our geometric GNN variants still surpass all baselines. Compared to CNNs in [8], our GNNs achieve higher accuracy with fewer parameters and direct raw data processing, emphasizing their efficiency advantage.

Although the hardness prediction instances originate from the TSP-ISA algorithm selection dataset, the key features shift significantly due to the change in prediction targets. Moreover, as shown in Figure 2, the top 10 most important features differ substantially between the two hardness measures, CLK and LKCC. Table 3 presents the overall results for CLK and LKCC hardness prediction. While CNN performs comparably to GCN on LKCC, it performs poorly on CLK, revealing its lack of robustness across tasks. In contrast, our proposed GNN models

Table 2. Overall performance comparison on the TSP-CNN algorithm selection dataset (bold: best; underline: runner-up).

Category	Models	Input data	Parameters	Accuracy
Feature-based models	RF	All140 features	–	$73.30 \pm 5.10\%$
		Top15 features		$73.40 \pm 5.66\%$
		MST19 features		$73.90 \pm 4.81\%$
		kNNG51 features		$72.80 \pm 5.86\%$
CNNs	CNN [8]	Points+MST+kNNG images	1,174,690	$70.50 \pm 7.55\%$
		Points+MST images	1,174,402	$72.00 \pm 4.96\%$
		Points images	1,174,114	$71.80 \pm 6.63\%$
Standard GNNs	GCN	Non-geometric graphs	4,290	$71.80 \pm 4.21\%$
	GINE	Non-geometric graphs	2,874	$72.10 \pm 3.56\%$
Point cloud GNNs	PointNet	Point clouds	962	$68.60 \pm 3.26\%$
	DGCNN	Point clouds	6,370	$64.10 \pm 4.64\%$
Proposed GIE-GNNs	GINE-GI	Geometric graphs	2,874	$\underline{74.30 \pm 4.45\%}$
	MP-GI	Geometric graphs	9,698	$74.20 \pm 4.83\%$
	GINE-GE	Geometric graphs	5,798	$73.80 \pm 4.28\%$
	MP-GE	Geometric graphs	9,830	$\mathbf{75.40 \pm 3.35}\%$

consistently achieve the highest accuracy across both tasks while maintaining lower model complexity. Consistent with our findings in algorithm selection, GNNs tailored for non-geometric graphs underperform compared to those designed for point clouds. Our GIE-GNNs are the only methods that outperform traditional feature-based models in all settings. From a statistical perspective, our models exhibit significant improvement over feature-based baselines on the CLK task, confirming that the observed gains are not due to random variation. For the LKCC task, our models also perform better on average, though the difference is not statistically conclusive. Nonetheless, the overall trend, combined with substantial and consistent margins over CNN, supports the robustness and generalization ability of our geometry-aware GNN framework for TSP hardness prediction.

3.3 Ablation study

This section evaluates the effectiveness of different parts of the proposed GNNs. First, we compare the performance of GINE, GINE-GI, and GINE-GE on various tasks. GINE is a GNN designed for non-geometric graphs, and its information propagation is not geometric invariant or equivariant. On the other hand, GINE-GI and GINE-GE are our geometric invariant GNN and geometric equivariant GNN, respectively. As shown in Table 4, by transforming GINE into a geometrically invariant GNN, we observe a significant improvement in prediction accuracy across all tasks. Although geometrically equivariant GNNs are theoretically more powerful than geometrically invariant GNNs, the performances of GINE-GE are slightly worse than GINE-GI in some cases. There could be two possible reasons for this. Firstly, both models already achieve close to 100% prediction

Table 3. CLK and LKCC hardness prediction overall performance comparison.

Models	Input data	Parameters	CLK Acc	LKCC Acc
RF	All140 features	–	$98.82 \pm 0.49\%$	$97.50 \pm 0.97\%$
	Top15 features		$97.76 \pm 0.53\%$	$87.24 \pm 2.84\%$
	MST19 features		$98.29 \pm 1.07\%$	$95.66 \pm 0.53\%$
	kNNG51 features		$96.45 \pm 0.67\%$	$70.53 \pm 5.60\%$
CNN	Points images	132,466	$77.50 \pm 3.39\%$	$92.37 \pm 1.22\%$
GCN	Non-geometric graphs	1,122	$85.79 \pm 2.06\%$	$92.63 \pm 3.42\%$
GINE	Non-geometric graphs	1,266	$97.11 \pm 0.89\%$	$95.13 \pm 1.79\%$
PointNet	Point clouds	962	$91.71 \pm 1.48\%$	$95.92 \pm 2.33\%$
DGCNN	Point clouds	6,370	$93.68 \pm 0.67\%$	$95.66 \pm 0.98\%$
GINE-GI	Geometric graphs	1,266	$\mathbf{99.61 \pm 0.32}\%$	$97.76 \pm 1.07\%$
MP-GI	Geometric graphs	2,546	$97.89 \pm 1.83\%$	$\underline{97.63 \pm 1.07\%}$
GINE-GE	Geometric graphs	1,622	$98.82 \pm 1.47\%$	$\mathbf{98.29 \pm 0.89}\%$
MP-GE	Geometric graphs	2,614	$\underline{98.95 \pm 1.22\%}$	$96.97 \pm 1.22\%$

Table 4. The prediction accuracy of GINE, GINE-GI, and GINE-GE on different algorithm selection and hardness prediction tasks.

Models	Geometrically Invariant	Geometrically Equivariant	ISA-TSP Algorithm selection	ISA-TSP CLK Hardness	ISA-TSP LKCC Hardness	CNN-TSP Algorithm selection
GINE	✗	✗	$98.65 \pm 0.94\%$	$97.11 \pm 0.89\%$	$95.13 \pm 1.79\%$	$72.10 \pm 3.56\%$
GINE-GI	✓	✗	$\underline{98.87 \pm 0.97\%}$	$\mathbf{99.61 \pm 0.32}\%$	$\underline{97.76 \pm 1.07\%}$	$\mathbf{74.30 \pm 4.45}\%$
GINE-GE	✗	✓	$\mathbf{99.47 \pm 0.59}\%$	$\underline{98.82 \pm 1.47\%}$	$\mathbf{98.29 \pm 0.89}\%$	$\underline{73.80 \pm 4.28\%}$

accuracy, leaving little room for improvement. Additionally, the GINE-GE model is more complex and thus has a higher risk of over-fitting. Next, we investigate the impact of the adopted positional encoding method named RWPE on the model performance on two hardness prediction tasks and summarize it in Table 5. When RWPE is not applied, we set the point features to a zero tensor of the same dimension. In most cases, using RWPE helps to distinguish graph attributes, but there are also some cases where directly setting the features to zero yields better results. We found that when using GINE as the message-passing mechanism, RWPE consistently improves performance. However, when using MP to propagate information, RWPE has a negative impact. One possible reason is that in MP, directly concatenating RWPE with different information and performing multiple linear transformations may introduce instability.

3.4 Geometrically invariant analysis

The inherent hardness and optimal solutions of TSP instances are fundamentally geometrically invariant; they remain constant under coordinate transformations like rotation or translation. This necessitates prediction models to maintain

Table 5. The performance comparison of applying RWPE on the TSP-ISA CLK and LKCC hardness prediction dataset.

	CLK hardness prediction		LKCC hardness prediction	
Positional Encoding	None	RWPE	None	RWPE
GINE-GI	$98.68 \pm 1.25\%$	$99.61 \pm 0.32\%\uparrow$	$97.11 \pm 0.67\%$	$97.76 \pm 1.07\%\uparrow$
MP-GI	$98.82 \pm 0.64\%$	$97.89 \pm 1.83\%\downarrow$	$96.97 \pm 2.23\%$	$97.63 \pm 1.07\%\uparrow$
GINE-GE	$98.55 \pm 0.77\%$	$98.82 \pm 1.47\%\uparrow$	$97.76 \pm 1.22\%$	$98.29 \pm 0.89\%\uparrow$
MP-GE	$99.08 \pm 0.53\%$	$98.95 \pm 1.22\%\downarrow$	$97.24 \pm 1.13\%$	$96.97 \pm 1.22\%\downarrow$

Table 6. The prediction accuracy on rotated training and test data of the TSP-ISA CLK hardness prediction dataset.

Data / Model	Training Data				Test Data			
	Orig.	45°	90°	180°	Orig.	45°	90°	180°
GCN	95.07	80.69↓	84.70↓	83.75↓	85.79	81.58↓	86.32↑	83.03↓
GINE	99.31	96.32↓	96.25↓	95.92↓	97.11	96.45↓	95.79↓	96.18↓
PointNet	99.70	91.05↓	91.25↓	90.62↓	91.71	92.11↑	89.21↓	92.37↑
DGCNN	100.00	89.18↓	91.64↓	92.53↓	93.68	90.13↓	91.97↓	92.11↓
GINE-GI	99.67	99.67↔	99.67↔	99.67↔	99.61	99.61↔	99.61↔	99.61↔
MP-GI	99.93	99.93↔	99.93↔	99.93↔	97.89	97.89↔	97.89↔	97.89↔
GINE-GE	99.93	99.93↔	99.93↔	99.93↔	98.82	98.82↔	98.82↔	98.82↔
MP-GE	100.00	100.00↔	100.00↔	100.00↔	98.95	98.95↔	98.95↔	98.95↔

geometric invariance, a property that is unattainable without explicitly treating TSPs as geometric graphs through invariant/equivariant GNN architectures. To validate this critical property, we evaluated model robustness under coordinate rotations applied to both training and test data. Table 6 compares prediction consistency across three rotation configurations. Traditional GNNs for non-geometric graphs and point clouds show significant performance degradation, demonstrating their coordinate system dependency. In contrast, our geometric GNNs consistently maintain unchanged prediction results. Our approach achieves this through two principled designs: 1) Geometrically invariant GNNs naturally maintain invariance by only propagating scalar features; 2) Geometrically equivariant GNNs maintain equivariant intermediate representations while enforcing invariance through graph-level pooling. This geometric-aware learning focuses on essential structural patterns rather than coordinate system artifacts, ensuring reliable predictions. Figure 3 uses t-SNE to compare the graph-level embeddings of GINE, GINE-GI, and GINE-GE on both original and rotated test instances, illustrating that GINE-GI and GINE-GE maintain more stable and separable representations compared to GINE.

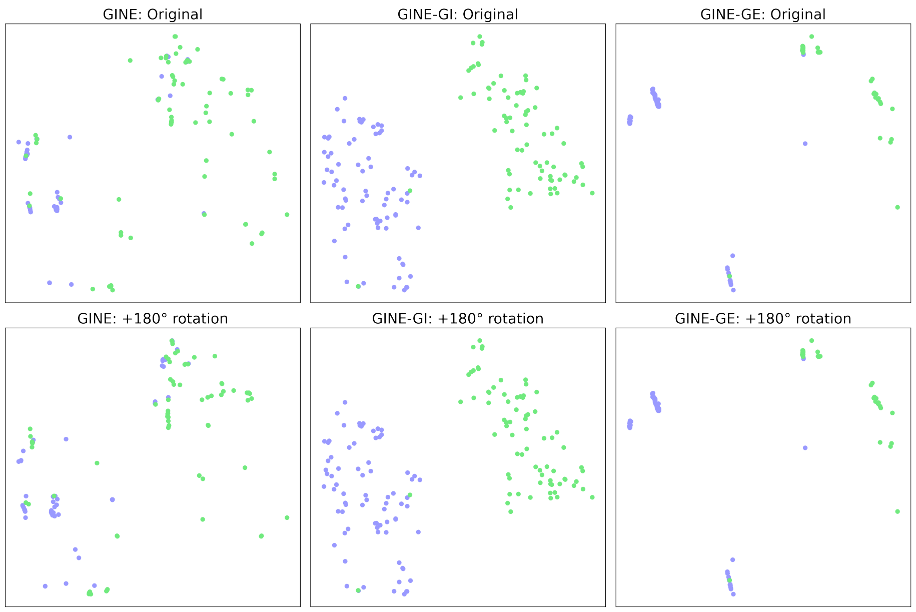

Fig. 3. The graph embeddings of GINE, proposed GINE-GI, and proposed GINE-GE before and after rotation on the TSP-ISA CLK hardness prediction dataset.

3.5 Model properties comparison

Table 7 provides an overview of the characteristics of feature-based models, CNNs, conventional GNNs used for non-geometric graphs and point clouds, as well as the proposed GNNs in relation to the task of selecting algorithms and predicting hardness for TSP. The feature-based approach suffers from heavy reliance on domain expertise and poor feature generalizability across datasets. While CNNs automate feature learning, they require computationally intensive image conversions with sensitive hyperparameters that lack theoretical justification. More critically, their image-based paradigm becomes impractical for complex routing problems such as VRP with heterogeneous nodes. Our geometrically invariant and equivariant GNNs overcome these limitations by directly processing coordinate and distance matrices, eliminating manual feature engineering and image generation. Unlike traditional GNNs that ignore spatial relationships, we explicitly model geometric invariances through dedicated message propagation mechanisms. This architecture achieves superior prediction accuracy while maintaining robustness across problem scales - particularly crucial as TSP instance sizes vary. The coordinate native representation also facilitates effortless extension to various routing problems by straightforwardly augmenting features, such as identifying depot nodes for VRPs.

Table 7. Properties comparison between Feature-based model, CNN, traditional GNNs, and Our GNNs for algorithm selection and hardness prediction.

Properties	Feature-based	CNN[8]	Traditional GNNs	Our GNNs
Instance representations	Handcrafted features	Points, MST, kNNG images	Non-geometric graphs Point clouds	Geometric graphs
Feature engineering required	Yes	No	No	No
Problem-irrelevant parameters	None	Image size Dot/line params	None	None
Data Augmentation	None	Random rotation Random flipping	None	None
Information loss	Domain knowledge dependent	Resolution dependent	None	None
Generalize to VRP	Hard	Hard	Easy to add node features	Easy to add node features
Geometrically invariant prediction	No	No	No	Yes

4 Conclusion

We propose a group of geometrically invariant and equivariant GNNs to do algorithm selection and hardness prediction for TSP. Our proposed GNNs can learn useful spatial information of TSP instances and outperform traditional feature-based models, CNNs, and GNNs designed for non-geometric graphs or point clouds on four public datasets. Our GNNs only take cities' coordinates and distance matrices as input. Thus, no intermediate representations for problem instances, such as features or images, need to be designed and generated prior to model training. In contrast to converting TSP instances to images, the graph representation is more natural and efficient, as it neither introduces problem-irrelevant parameters nor loses problem-relevant information. We have shown the proposed GNNs are promising as they can provide geometrically invariant predictions, which are more reliable and robust. Besides, they are easy to generalize to other routing problems. For example, we can distinguish nodes and routes in the problem instances by adding node and edge features.

In this work, we use a well-studied problem, TSP, to demonstrate our proposed approaches to algorithm selection and instance hardness prediction, as benchmark datasets and methods (handcrafted feature-based and CNN-based) for TSP are public available, which allows us to do a fair comparison. In the future, we will test our architectures for more complex variants of routing problems.

References

1. Cheng, H., Zheng, H., Cong, Y., Jiang, W., Pu, S.: Select and optimize: Learning to solve large-scale tsp instances. In: International Conference on Artificial Intelligence and Statistics. pp. 1219–1231. PMLR (2023)
2. Huerta, I.I., Neira, D.A., Ortega, D.A., Varas, V., Godoy, J., Asín-Achá, R.: Improving the state-of-the-art in the traveling salesman problem: an anytime automatic algorithm selection. Expert Systems with Applications **187**, 115948 (2022)
3. Rice, J.R.: The algorithm selection problem. In: Advances in computers, vol. 15, pp. 65–118. Elsevier (1976)

4. Alipour, H., Muñoz, M.A., Smith-Miles, K.: Enhanced instance space analysis for the maximum flow problem. European Journal of Operational Research **304**(2), 411–428 (2023)
5. Pereira, J.L.J., Smith-Miles, K., Muñoz, M.A., Lorena, A.C.: Optimal selection of benchmarking datasets for unbiased machine learning algorithm evaluation. Data Mining and Knowledge Discovery **38**(2), 461–500 (2024)
6. Smith-Miles, K., Baatar, D., Wreford, B., Lewis, R.: Towards objective measures of algorithm performance across instance space. Computers & Operations Research **45**, 12–24 (2014)
7. Bossek, J.: Salesperson: computation of instance features and r interface to the state-of-the-art exact and inexact solvers for the traveling salesperson problem (2017)
8. Seiler, M., Pohl, J., Bossek, J., Kerschke, P., Trautmann, H.: Deep learning as a competitive feature-free approach for automated algorithm selection on the traveling salesperson problem. In: Parallel Problem Solving from Nature–PPSN XVI: 16th International Conference, PPSN 2020, Leiden, The Netherlands, September 5-9, 2020, Proceedings, Part I. pp. 48–64. Springer (2020)
9. Zhao, K., Liu, S., Yu, J.X., Rong, Y.: Towards feature-free tsp solver selection: A deep learning approach. In: 2021 International Joint Conference on Neural Networks (IJCNN). pp. 1–8. IEEE (2021)
10. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 770–778 (2016)
11. Dwivedi, V.P., Joshi, C.K., Laurent, T., Bengio, Y., Bresson, X.: Benchmarking graph neural networks (2020)
12. Dwivedi, V.P., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X.: Graph neural networks with learnable structural and positional representations. arXiv preprint arXiv:2110.07875 (2021)
13. Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., Leskovec, J.: Strategies for pre-training graph neural networks. arXiv preprint arXiv:1905.12265 (2019)
14. Satorras, V.G., Hoogeboom, E., Welling, M.: E (n) equivariant graph neural networks. In: International conference on machine learning. pp. 9323–9332. PMLR (2021)
15. Zhao, L., Akoglu, L.: Pairnorm: Tackling oversmoothing in gnns. arXiv preprint arXiv:1909.12223 (2019)
16. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.i., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In: International conference on machine learning. pp. 5453–5462. PMLR (2018)
17. Smith-Miles, K., van Hemert, J.: Discovering the suitability of optimisation algorithms by learning from evolved instances. Annals of Mathematics and Artificial Intelligence **61**, 87–104 (2011)
18. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907 (2016)
19. Qi, C.R., Su, H., Mo, K., Guibas, L.J.: 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 (2017)
20. Wang, Y., Sun, Y., Liu, Z., Sarma, S.E., Bronstein, M.M., Solomon, J.M.: Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (tog) **38**(5), 1–12 (2019)