ARCX: A UNIFIED RELATION ENCODER FOR VECTOR SPATIAL REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Vector spatial representation learning (VSRL) has primarily focused on representing individual spatial entities, such as their locations, geometries, and attributes, while paying limited attention to pairwise spatial entity relations. Current methods face two limitations in spatial relation representation. First, they often rely on centroid distance, which can discard geometric details. Second, relations are typically inferred only after the two entities are encoded independently, resulting in limited spatial interaction. We introduce pair-level VSRL as a complementary representation perspective and propose ArcX, a unified VSRL encoder that generates dedicated spatial relation embeddings for entity pairs. ArcX introduces an arc-wise cross-entity modulation mechanism, allowing each entity to respond to the other before embedding readout. We evaluate ArcX across five datasets spanning pair-, entity-, and scene-level VSRL tasks. The results show that ArcX can capture multiple dimensions of spatial relations, outperforming the second-best method by an average of 10.8% on pairwise tasks. ArcX can also improve entity-level neighborhood aggregation by an average of 4.4% over the second-best method and scene-level information propagation by an average of 16.5%, demonstrating consistent utility across diverse task settings. ArcX introduces a relation-centric perspective to VSRL, complementing existing entity- and scene-level perspectives. The code and data are available at `https://anonymous.4open.science/r/ArcX-6BB3`.

1 INTRODUCTION

Vector spatial representation learning (VSRL) maps vector-based spatial objects into machine-readable embeddings (Mai et al., 2022; Zhang et al., 2025). As vector data are increasingly used to model real-world environments, VSRL quality largely determines how effectively spatial information can support downstream tasks.

VSRL is usually studied at two levels. Entity-level VSRL learns embeddings for individual entities to capture characteristics such as shape, location, and attributes (Mai et al., 2020; Jiang et al., 2019; Mai et al., 2023). Scene-level VSRL aggregates multiple entity embeddings to describe large-scale spatial characteristics (Gao et al., 2020; Li et al., 2023). These two VSRL levels naturally support each other: entity-level embeddings provide basic information for scene-level representation, while scene-level propagation mechanisms can, in turn, enable individual entities to incorporate neighborhood information (Kipf & Welling, 2017; Gilmer et al., 2017). However, research at these two levels remains largely separate, without fully leveraging the complementary capabilities of one another.

We argue that this separation stems from an important but underexplored problem: representing spatial relations between entities, which we term pair-level VSRL (Figure 1(a)). The objective of pair-level VSRL is to shift the representational focus from entity-centric to relation-centric by generating dedicated embeddings for spatial relations. The resulting relation embeddings are expected to jointly capture multiple aspects of spatial relations, including distance, direction, and topology.

However, pair-level VSRL has received limited dedicated study. Existing methods that support pairwise spatial reasoning exhibit two major limitations. First, spatial affinity is often described using centroid distance. While useful for measuring overall affinity, centroid distance ignores the spatial extent and geometric configuration of entities. Actually, entity pairs with similar centroid distances can differ substantially in both directional and topological relations, including relative

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

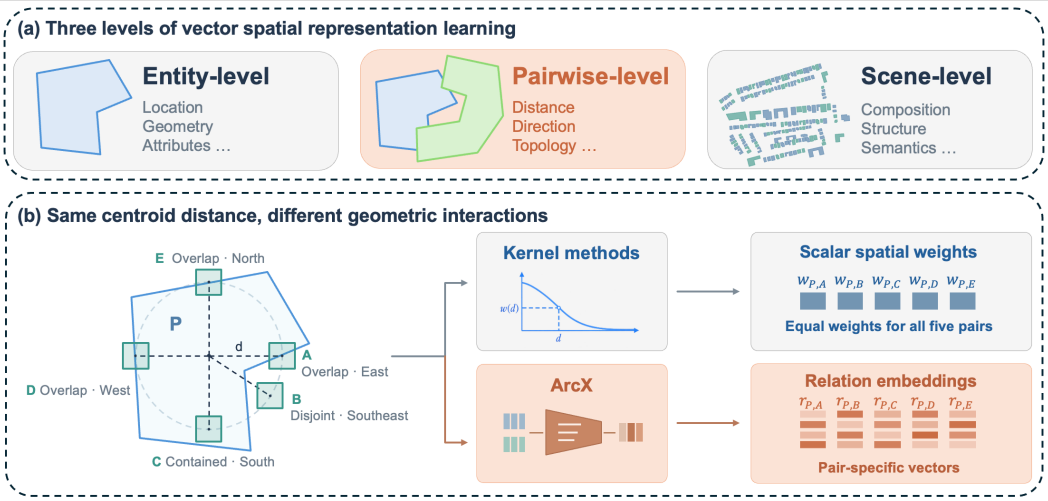

Figure 1: Overview of VSRL levels. (a) Entity-, pair-, and scene-level VSRL emphasize different aspects. (b) Entity pairs with the same centroid distance can differ in spatial relations. A distance kernel assigns them the same weight, whereas ArcX encodes pair-specific embeddings.

orientation, overlapping, and containment. Second, existing methods provide insufficient cross-entity interaction during the encoding stage. By the time interaction occurs at later stages, the entity representations may already have been compressed.

To address these limitations, we shift VSRL from an entity-centric perspective to a relation-centric perspective and propose ArcX, a unified VSRL encoder specially designed to generate spatial relation embeddings for entity pairs. First, both entities are expressed in a shared pair-local coordinate system. Then, ArcX decomposes each entity into arc tokens and conditions these tokens on a pooled summary of the other entity. This introduces cross-entity conditioning before embedding readout, rather than combining independently encoded representations only afterward. We conduct experiments on five datasets, covering pairwise VSRL tasks as well as entity- and scene-level tasks. Our goal is to not only evaluate ArcX's ability to represent pairwise spatial relations, but also to demonstrate that such relation embeddings can support multiple VSRL scenarios, thereby bridging different VSRL levels within a common framework.

Our main contributions are:

- We introduce ArcX, a spatial relation encoder for entity pairs. ArcX uses arc-wise cross-entity modulation to jointly represent multiple aspects of spatial relations between two spatial entities.
- We use ArcX embeddings to modulate inter-entity message passing, connecting pair-level VSRL with entity-level neighborhood aggregation and scene-level information propagation.
- We evaluate ArcX on five datasets that span all three levels of VSRL tasks. Our results demonstrate that ArcX can capture spatial relations and provide consistent utility across diverse task scenarios and spatial scales.

Together, the three VSRL levels constitute a more complete VSRL framework: entity-level addresses what information each entity should have; scene-level addresses which entities should communicate; pair-level bridges the two by addressing how these entities should exchange messages.

2 Related Work

Existing VSRL methods, while accounting for spatial relations to some extent, are predominantly entity-centric. Polygon-centric methods, such as NUFTspec (Mai et al., 2023) and PolygonGNN (Yu et al., 2024), focus on preserving the geometric characteristics of polygons; more unified methods, such as Poly2Vec (Siampou et al., 2025) and Geo2Vec (Chu & Shahabi, 2026), extend VSRL to heterogeneous geometry types. These methods often support pairwise spatial reasoning tasks, such

as distance estimation and topological classification (Mai et al., 2023; Siampou et al., 2025; Chu & Shahabi, 2026). However, their primary focus remains the individual entities rather than their relations; pairwise reasoning is typically performed only after the two entities have been independently encoded.

At larger scales, spatial relations are commonly represented as scalar affinities that are generated using distance heuristics or kernel methods (Appleby et al., 2020; Zhang et al., 2026). Classical spatial models, such as kriging, characterize spatial relatedness through distance covariance (Rasmussen & Williams, 2006). Recent deep spatial models follow a similar principle. KCN (Appleby et al., 2020) defines node adjacency using a Gaussian spatial kernel, while SPONGE (Abémgnigni Njifon & Schuhmacher, 2024) further introduces learnable convolution kernels. Despite differences in design details, these methods remain strongly centered on centroid distance: spatial entities are approximated as points, and their relationship is reduced to a distance-based scalar. Such approximations are effective, but overlook the actual geometric and spatial configuration. Consequently, spatial relations beyond proximity, such as direction and topology, are difficult to represent explicitly.

These two lines of research leave a complementary gap. Entity-level VSRL preserves increasingly rich individual information, but spatial interactions are often limited during the encoding process. Scene-level VSRL uses pairwise dependencies for aggregation, but typically reduces them to scalar affinities. To our knowledge, few existing methods explicitly target relation-centric VSRL while simultaneously supporting heterogeneous geometry types and representing multiple dimensions of spatial relations. ArcX addresses this gap by treating pairwise spatial relations as a representation target in its own right. Through cross-entity interaction, it produces dedicated relation embeddings that capture multiple spatial relation characteristics and can be reused across pair-level reasoning, entity-level neighborhood aggregation, and scene-level information propagation.

3 PRELIMINARIES

3.1 DEFINITIONS

Spatial entity. A spatial entity E_i is an object characterized by its spatial position. It is typically recorded in coordinates in $\mathbb{R}^2$. Common entity types include point, polyline, polygon, multi-polyline and multi-polygon.

Spatial entity pair. An entity pair (E_i, E_j) is a pair of two complete spatial entities. The two entities are not required to be of the same entity type.

Spatial entity representation. The task of spatial entity representation is to learn a mapping

$$f_{\text{ent}} : \mathcal{E} \to \mathbb{R}^{d_e}, \qquad E_i \mapsto \mathbf{e}_i,$$

where $\mathcal{E}$ denotes the domain of spatial entities and $\mathbf{e}_i$ is a d_e-dimensional embedding that captures the individual characteristics of E_i, such as its location, shape, and attributes.

Spatial relation representation. The task of spatial relation representation is to learn a mapping

$$f_{\text{rel}} : \mathcal{E} \times \mathcal{E} \to \mathbb{R}^{d_r}, \qquad (E_i, E_j) \mapsto \mathbf{r}_{ij},$$

where $\mathcal{E} \times \mathcal{E}$ denotes the domain of entity pairs and $\mathbf{r}_{ij}$ is a d_r-dimensional representation that captures the pairwise characteristics between E_i and E_j, such as distance, direction, and topology (Egenhofer & Franzosa, 1991; Siampou et al., 2025).

3.2 ARCSET

ArcSet is an entity-level VSRL encoder that uses arcs, rather than conventional vertices, as the tokens of a spatial entity. It first decomposes each entity into arc tokens and then applies self-attention to enable information exchange and feature updates among the arc tokens. The final output is a set of arc embeddings whose cardinality equals the number of arcs in the entity:

$$\mathbf{A}_i = \text{ArcSet}_{\text{token}}(E_i) \in \mathbb{R}^{n_i \times d_a},$$

where n_i denotes the number of arcs in E_i and d_a is the arc-token dimension. Each row of $\mathbf{A}_i$ corresponds to one arc and is subsequently used for cross-entity interaction in ArcX.

We treat each point as a zero-length arc. The detailed process of ArcSet is provided in Appendix A.

3.3 Triple-Distance Spatial Kernel

The Triple-Distance Spatial Kernel (TDSK) computes a scalar affinity between two spatial entities from three complementary geometric distances. Given an entity pair (E_i, E_j) in a metric coordinate system, the minimum, centroid, and maximum distances are defined as

$$\begin{aligned} d_{ij}^{\min} &= \min_{\mathbf{x}\in E_i,\, \mathbf{y}\in E_j} \|\mathbf{x}-\mathbf{y}\|_2, \\ d_{ij}^{\text{cent}} &= \|\operatorname{centroid}(E_i) - \operatorname{centroid}(E_j)\|_2, \\ d_{ij}^{\max} &= \max_{\mathbf{x}\in E_i,\, \mathbf{y}\in E_j} \|\mathbf{x}-\mathbf{y}\|_2. \end{aligned} \tag{1}$$

The minimum distance measures the closest separation between the two entities and becomes zero when they intersect. The centroid distance measures their overall separation. The maximum distance reflects their spatial extents. Each distance is converted into an affinity using an exponential kernel:

$$k_{ij}^r = \exp\left(-\frac{d_{ij}^r}{\tau_r}\right), \qquad \tau_r = \tau_r^{(0)} \exp(\delta_r), \qquad r \in \mathcal{R} = \{\min, \text{cent}, \max\}. \tag{2}$$

Here, $\tau_r^{(0)}$ is initialized to the median of the corresponding distances over the training-set entity pairs. The trainable parameter δ_r is initialized to zero, allowing each scale parameter to remain positive during training. TDSK combines the three affinities through learnable weights:

$$\lambda_r = \frac{\exp(\theta_r)}{\sum_{q\in\mathcal{R}} \exp(\theta_q)}, \qquad s_{ij} = \sum_{r\in\mathcal{R}} \lambda_r k_{ij}^r. \tag{3}$$

The logits θ_r are initialized equally, giving each affinity an initial weight of $1/3$. Both the mixture weights and spatial scale parameters are shared across entity pairs and optimized by downstream tasks. The resulting affinity $s_{ij} \in (0, 1]$ determines the spatial association between the two entities.

4 ArcX

ArcX encodes the spatial relation between an entity pair into a fixed-dimensional embedding (Figure 2). Given an entity pair (E_i, E_j), it first places both entities in a shared local coordinate system, then uses ArcSet to convert each entity into a set of arc tokens and enables arc-level cross-entity information exchanges. Finally, the two arc sets are aggregated into a relation embedding $\mathbf{r}_{i\to j} \in \mathbb{R}^{d_r}$.

4.1 Pair-Local Coordinate Transformation

A spatial relation depends on the relative configuration of two entities. To reduce the influence of their absolute positions while retaining this relative configuration, we translate both entities into a pair-local coordinate system. Let $\mathbf{v}_{ij}^{\min}, \mathbf{v}_{ij}^{\max} \in \mathbb{R}^2$ be the lower and upper corners of their joint axis-aligned bounding box. We define

$$\mathbf{o}_{ij} = \frac{\mathbf{v}_{ij}^{\min} + \mathbf{v}_{ij}^{\max}}{2}, \qquad \widetilde{E}_i = \{\mathbf{v} - \mathbf{o}_{ij} : \mathbf{v} \in E_i\}, \qquad \widetilde{E}_j = \{\mathbf{v} - \mathbf{o}_{ij} : \mathbf{v} \in E_j\}. \tag{4}$$

The pair-local coordinate transformation involves no scaling nor rotation, thereby preserving the relative distance, orientation, and topology between the two entities, as well as their original scales.

4.2 Arc-Token Encoding and Interaction

We apply a shared ArcSet encoder to each pair-local entity, obtaining arc-token sets

$$\mathcal{A}_i = \{\mathbf{a}_{ik}\}_{k=1}^{n_i}, \qquad \mathcal{A}_j = \{\mathbf{a}_{jk}\}_{k=1}^{n_j}, \qquad \mathbf{a}_{ik}, \mathbf{a}_{jk} \in \mathbb{R}^{d_a}, \tag{5}$$

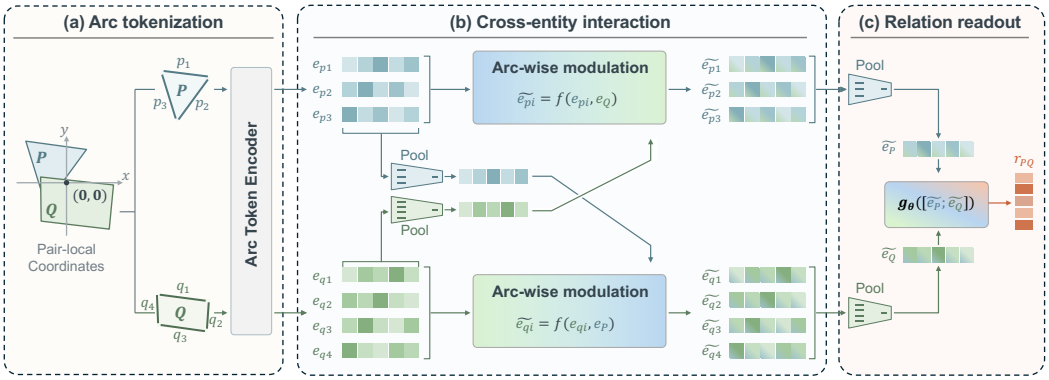

Figure 2: Overview of ArcX. (a) Both entities are expressed in a shared local coordinate system and encoded into arc tokens using a shared ArcSet. (b) Each entity's arc tokens are conditioned on the other entity. (c) The conditioned token sets are combined to obtain a relation embedding.

where n_i and n_j denote the numbers of valid arcs. Point entities are represented by single degenerate tokens without length and angle features.

To let internal entity geometry features respond to the other entity, we introduce a compact cross-entity branch. We first project each contextualized arc token into a lower-dimensional interaction space:

$$q_{uk} = W_c a_{uk} \in \mathbb{R}^{d_c}, \qquad c_u = \frac{1}{n_u} \sum_{k=1}^{n_u} q_{uk}, \quad u \in \{i, j\}, \tag{6}$$

where W_c is a learnable linear projection and c_u summarizes entity E_u in the interaction space.

Each projected arc token of E_i is then modulated using its own features together with the summary of E_j:

$$\alpha_{ik|j} = \sigma(f_{\text{gate}}([q_{ik}; c_j])), \qquad b_{ik|j} = \alpha_{ik|j} q_{ik}. \tag{7}$$

Here, f_{gate} is an MLP with a scalar output. The gate assigns different weights to arc tokens according to their relatedness to the other entity E_j. Applying the same operation with i and j exchanged gives the reverse updated arc tokens $b_{jk|i}$.

The resulting arc sets $\mathcal{B}_{i|j} = \{\mathbf{b}_{ik|j}\}_{k=1}^{n_i}$ and $\mathcal{B}_{j|i} = \{\mathbf{b}_{jk|i}\}_{k=1}^{n_j}$ retain arc-level resolution while incorporating cross-entity context. Interaction occurs through entity summaries, avoiding the $n_i n_j$ token comparisons in dense cross-attention (Vaswani et al., 2017). Under fixed feature dimensions, this cross branch scales linearly with the total token count.

4.3 Relation Readout

The relation readout combines each entity's features before and after cross-entity interaction. The intra-entity branch retains features prior to entity interaction, and the cross-entity branch supplies their relation-conditioned responses. For the intra-entity branch, we reuse ArcSet's entity readout on the original contextualized arc tokens:

$$\mathbf{h}_u^{\text{intra}} = \text{Readout}_{\text{intra}}(\mathcal{A}_u), \qquad u \in \{i, j\}, \tag{8}$$

where $\text{Readout}_{\text{intra}}$ includes the set-level processing described in Section 3.2. This readout differs from the compact summary $\mathbf{c}_u$: the latter conditions cross-entity gates, whereas $\mathbf{h}_u^{\text{intra}}$ retains the entity features for final information fusion. For the cross-entity branch, we use max pooling to retain the largest response in each feature channel:

$$\mathbf{h}_{i|j}^{\text{cross}} = \text{Readout}_{\text{cross}}(\mathcal{B}_{i|j}), \qquad \mathbf{h}_{j|i}^{\text{cross}} = \text{Readout}_{\text{cross}}(\mathcal{B}_{j|i}), \tag{9}$$

where $\text{Readout}_{\text{cross}}$ denotes element-wise pooling over the cross-entity arc tokens. We then concatenate these representations to obtain the final relation embedding:

$$\mathbf{r}_{i \to j} = \rho_{\text{rel}}\left([\mathbf{h}_i^{\text{intra}}; \mathbf{h}_j^{\text{intra}}; \mathbf{h}_{i|j}^{\text{cross}}; \mathbf{h}_{j|i}^{\text{cross}}]\right), \tag{10}$$

where ρ_{rel} is an MLP. The reverse embedding $\mathbf{r}_{j\to i}$ uses the same MLP with the entity order exchanged. Thus, both directions share the same information, while the ordered readout allows them to output different embeddings.

4.4 Relation-Conditioned Information Propagation

To incorporate the learned relation embeddings into downstream VSRL tasks, we use them to modify message flows between neighboring entities (Gilmer et al., 2017; Simonovsky & Komodakis, 2017). TDSK provides the spatial affinity between entities, while ArcX embeddings process the exchanged information.

Let $\mathbf{x}_i^{(\ell)}$ be the features of E_i at layer ℓ, and let $\mathcal{N}(i)$ denote its neighbors. Given the TDSK affinity s_{ij} defined in Section 3.3, we first normalize it over $\mathcal{N}(i)$ to obtain the propagation coefficient

$$\hat{s}_{ij} = \frac{s_{ij}}{\sum_{k\in\mathcal{N}(i)} s_{ik}}, \qquad j \in \mathcal{N}(i). \tag{11}$$

For messages flowing from E_j to E_i, let $\mathbf{r}_{j\to i}$ denote the relation embedding of the entity pair. We transform it into an element-wise gate to modulate raw projected messages from E_j (Perez et al., 2018):

$$\Delta\boldsymbol{\gamma}_{j\to i}^{(\ell)} = \tanh\left(W_\gamma^{(\ell)}\mathbf{r}_{j\to i}\right), \tag{12}$$

$$\mathbf{m}_{j\to i}^{(\ell)} = \left(\mathbf{1} + \Delta\boldsymbol{\gamma}_{j\to i}^{(\ell)}\right) \odot W_m^{(\ell)}\mathbf{x}_j^{(\ell)}. \tag{13}$$

We aggregate the relation-conditioned message flows from $\mathcal{N}(i)$ using the TDSK propagation coefficients and update $\mathbf{x}_i^{(\ell)}$ through a residual connection:

$$\mathbf{x}_i^{(\ell+1)} = \text{LayerNorm}\left(\mathbf{x}_i^{(\ell)} + W_o^{(\ell)} \sum_{j\in\mathcal{N}(i)} \hat{s}_{ij}\mathbf{m}_{j\to i}^{(\ell)}\right). \tag{14}$$

We initialize $W_\gamma^{(\ell)}$ to zero. Thus, the raw projected messages are preserved at initialization. Together with TDSK, the relation embeddings control the strength and routing of message flows. This design supports both neighborhood aggregation for downstream entity-level tasks and inter-entity information propagation for downstream scene-level tasks.

5 Experiments

5.1 Datasets

We evaluate ArcX on five datasets spanning pair-, entity-, and scene-level VSRL tasks.

VecCity (Zhang et al., 2025) contains urban entities, including POIs, road segments, and polygonal regions, from Beijing and Xi'an. We sample real-world entity pairs from VecCity and construct topology classification, direction estimation, and distance estimation tasks.

UUKG (Ning et al., 2023) contains urban entities, including POIs, road segments, road junctions, and urban areas, from Chicago, and organizes these entities into urban knowledge graphs. We evaluate bike (BIKE) and human mobility (HUMOB) prediction tasks, which require predicting the mobility demand of road segments and POIs from their surrounding urban environments.

HydroSHEDS (Lehner & Grill, 2013) contains vectorized drainage networks from watersheds across the Tibetan Plateau. We evaluate a flow direction classification task that requires identifying the dominant flow direction of each watershed from the geometric structure of its drainage network.

Local Climate Zones (LCZ) (Demuzere et al., 2022) contains building footprints grouped into rasterized urban regions. We evaluate an LCZ classification task that requires distinguishing different urban morphologies from the spatial distribution of building footprints.

Building Fragments contains artificially generated building fragments derived from real-world building footprints (Yan et al., 2021). The fragmentation procedure is described in the appendix. We evaluate a building shape classification task that requires identifying the shape category of each original building from the spatial pattern of its fragments.

5.2 BASELINES

We compare ArcX against three categories of baselines.

At the pair level, we compare against four entity encoders: TILE (Mai et al., 2020), THEORY (Mai et al., 2020), Poly2Vec (Siampou et al., 2025), and ArcSet.

At the entity and scene levels, we compare against multiple aggregation methods, including non-learnable spatial kernels (Uniform, Gaussian, and Spectral-Mixture (SM) (Wilson & Adams, 2013)), learnable spatial weighting methods (Learnable Gaussian (L-Gaussian), Embedding Self-Attention (ESA) (Vaswani et al., 2017), and Distance-Embedding Self-Attention (DESA)), and graph neural networks (Graph Convolutional Network (GCN) (Kipf & Welling, 2017), Graph Attention Network (GAT) (Veličković et al., 2018)). For the tasks derived from UUKG (Ning et al., 2023), we also compare against the baselines provided by these benchmarks.

5.3 PAIRWISE TASKS

We evaluate ArcX on direction estimation, distance estimation, and topology classification to assess its ability to represent pairwise relations. We jointly train all three tasks with a shared encoder to test whether a single embedding can capture multiple spatial relation dimensions. We use the Beijing dataset for training, parameter selection, and testing, while the Xi'an dataset is used only as an external test set to assess ArcX's cross-city generalization ability. All baselines independently encode the two entities in the same pair-local coordinate system and combine their entity representations using the same prediction head. In particular, the ArcSet baseline shares the ArcSet backbone with ArcX but omits cross-entity modulation before readout.

The results in Tables 1 and 2 show that ArcX achieves the best performance across all evaluated metrics in both cities. On the Beijing test set, ArcX improves topology Macro-F1 and direction accuracy over the second-best methods by 6.5% and 3.7%, while reducing direction and distance MAE by 23% and 13%. Similar advantages are retained on the zero-shot Xi'an test set, where ArcX again ranks first across all metrics. In particular, ArcX's improvement over ArcSet provides evidence for the benefit of arc-wise cross-entity modulation. These results indicate that a single ArcX embedding can jointly capture multiple dimensions of spatial relations, with strong performance maintained on the external Xi'an test set.

Table 1: Pairwise task results on the Beijing test set. Topology is evaluated using Macro-F1, direction using Accuracy (8 classes) and MAE, and distance using centroid distance MAE. The best result in each column is bold, and the second-best result is underlined.

	Topology	Direction		Distance
Method	Macro-F1 ↑	Acc ↑	MAE (°) ↓	MAE (m) ↓
TILE	0.3495 ± 0.0356	0.5522 ± 0.0061	12.89 ± 0.21	124.64 ± 18.19
THEORY	0.7677 ± 0.0078	0.8393 ± 0.0120	4.09 ± 0.25	$\underline{31.94 \pm 2.25}$
Poly2Vec	0.7465 ± 0.0144	0.8218 ± 0.0021	5.05 ± 0.08	41.20 ± 2.82
ArcSet	$\underline{0.8017 \pm 0.0141}$	$\underline{0.8454 \pm 0.0065}$	$\underline{3.65 \pm 0.24}$	37.06 ± 5.50
ArcX (Ours)	$\mathbf{0.8540 \pm 0.0042}$	$\mathbf{0.8771 \pm 0.0055}$	$\mathbf{2.81 \pm 0.11}$	$\mathbf{27.93 \pm 1.49}$

5.4 ENTITY-LEVEL TASKS

We evaluate ArcX on bike trip and human mobility prediction tasks to assess whether relation-conditioned neighborhood aggregation can improve entity representations. We first construct distance and direction estimation tasks from the UUKG training split to pretrain the parameters of

ArcX. The initial entity embeddings are provided by the UUKG benchmark (Ning et al., 2023), and aggregation methods are applied to update them using neighborhood information.

As shown in Table 3, ArcX achieves the best performance on both entity-level tasks. Compared with the second-best methods, ArcX reduces RMSE by approximately 5.3% on bike trip prediction and 3.5% on human mobility prediction. ArcX also consistently outperforms conventional graph- and attention-based methods, suggesting that relation-conditioned processing of neighborhood information can improve updates of the central entity nodes. These results demonstrate that ArcX embeddings can support entity-level VSRL tasks beyond pairwise tasks.

Table 2: Pairwise task results on the Xi'an test set. Topology is evaluated using Macro-F1, direction using Accuracy (8 classes) and MAE, and distance using centroid distance MAE. The best result in each column is bold, and the second-best result is underlined.

	Topology	Direction		Distance
Method	Macro-F1 ↑	Acc ↑	MAE (°) ↓	MAE (m) ↓
TILE	0.3541 ± 0.0424	0.5461 ± 0.0045	13.36 ± 0.22	128.70 ± 17.42
THEORY	0.7698 ± 0.0079	0.8090 ± 0.0196	4.44 ± 0.40	$\underline{30.49 \pm 2.21}$
Poly2Vec	0.7240 ± 0.0049	0.7949 ± 0.0064	5.55 ± 0.15	45.18 ± 2.62
ArcSet	$\underline{0.8112 \pm 0.0105}$	$\underline{0.8516 \pm 0.0140}$	$\underline{3.64 \pm 0.34}$	37.13 ± 5.40
ArcX (Ours)	$\mathbf{0.8643 \pm 0.0028}$	$\mathbf{0.8689 \pm 0.0082}$	$\mathbf{2.81 \pm 0.14}$	$\mathbf{27.63 \pm 1.56}$

Table 3: Entity-level task results. All tasks are evaluated using RMSE. The best result in each column is bold, and the second-best result is underlined.

Method	BIKE RMSE ↓	HUMOB RMSE ↓	Method	BIKE RMSE ↓	HUMOB RMSE ↓
UUKG	1.6543	2.3736	Gaussian	1.9692	2.8676
Uniform	1.9783	2.8756	SM	1.9582	2.9210
L-Gaussian	1.7996 ± 0.0034	2.8750 ± 0.0235	ESA	$\underline{1.5055 \pm 0.0347}$	2.3863 ± 0.0421
DESA	1.6177 ± 0.0481	$\underline{2.2357 \pm 0.0292}$	GAT	1.6884 ± 0.0522	2.4145 ± 0.0272
GCN	1.5902 ± 0.0607	2.2497 ± 0.0540	ArcX	$\mathbf{1.4260 \pm 0.0172}$	$\mathbf{2.1578 \pm 0.0227}$

5.5 Scene-Level Tasks

We evaluate ArcX on LCZ classification, flow direction classification, and building pattern classification tasks to assess whether relation-conditioned entity embeddings improve scene representations. Before each task, we construct distance and direction estimation tasks from the corresponding training split to pretrain the parameters of ArcX. We initialize entity embeddings using Geo2Vec (Chu & Shahabi, 2026) and apply aggregation methods to propagate messages between entities.

As shown in Table 4, ArcX consistently achieves the best performance across all three scene-level tasks, suggesting that relation-conditioned message propagation can improve scene-level representations. Table 5 further shows that TDSK learns task-dependent weights over the three distance components rather than relying on a single measure of spatial proximity. Notably, the centroid-distance kernel, which is commonly used in existing spatial aggregation methods, does not have the highest weight in any of the three tasks. The learned weights favor minimum distance for flow direction and building pattern classification, and maximum distance for LCZ classification, highlighting the value of modeling complementary geometric distances.

5.6 Ablation Study

We conduct ablation experiments on the building pattern classification task to examine the contribution of different components in our framework. Specifically, we compare the full model against variants that remove TDSK, ArcX, or ArcX pretraining, as well as variants pretrained using only distance or direction estimation. TDSK ablation evaluates the effect of explicitly modeling mul-

tiple geometric distances during spatial aggregation, while the ArcX pretraining ablation evaluates whether initializing ArcX with pretrained spatial relation representations benefits downstream tasks.

Table 6 shows that TDSK alone improves accuracy from 51.30% to 52.59%, while adding ArcX without pretraining further raises it to 56.92%, higher than all external baselines. ArcX without relation pretraining already outperforms all external baselines, indicating that the improvement is not solely attributable to auxiliary relation supervision. The full model achieves 70.36% accuracy. Distance-only and direction-only pretraining improve accuracy to 61.21% and 63.71%, respectively, but remain below the full model. These results support learning multiple aspects of spatial relations during pretraining to better inform downstream tasks.

Table 4: Scene-level task results. All tasks are evaluated using Accuracy. The best result in each column is bold, and the second-best result is underlined.

Method	Flow Direction Acc ↑	LCZ Acc ↑	Building Pattern Acc ↑
Uniform	0.5882	0.6552	0.4052
Gaussian	0.5882	0.6149	0.3842
SM	0.5714	0.6207	0.3782
L-Gaussian	0.6088 ± 0.0058	0.5747 ± 0.0205	0.4990 ± 0.0065
ESA	0.6116 ± 0.0115	0.6207 ± 0.0081	0.4907 ± 0.0033
DESA	0.6032 ± 0.0092	0.6034 ± 0.0124	0.4927 ± 0.0055
GAT	$\underline{0.6293 \pm 0.0115}$	0.6207 ± 0.0188	0.5020 ± 0.0078
GCN	0.6041 ± 0.0172	$\underline{0.6628 \pm 0.0187}$	$\underline{0.5213 \pm 0.0125}$
ArcX	$\mathbf{0.6592 \pm 0.0087}$	$\mathbf{0.7280 \pm 0.0222}$	$\mathbf{0.7036 \pm 0.0079}$

Table 5: TDSK weights learned on the three tasks. The highest weight on each task is bold.

Task	$\lambda_{\min}$	λ_{cent}	$\lambda_{\max}$
Flow Direction	$\mathbf{0.6003 \pm 0.1152}$	0.1832 ± 0.0485	0.2165 ± 0.0668
LCZ Classification	0.1698 ± 0.0024	0.4055 ± 0.0026	$\mathbf{0.4247 \pm 0.0050}$
Building Pattern	$\mathbf{0.7130 \pm 0.0418}$	0.0877 ± 0.0151	0.1993 ± 0.0344

Table 6: Ablation results on the Building Pattern task. All variants are evaluated using accuracy. The best result in each column is bold, and the second-best result is underlined.

Variant	Acc ↑	Variant	Acc ↑
w/o TDSK & ArcX	0.5130 ± 0.0093	Distance pretraining only	0.6121 ± 0.0070
TDSK only (w/o ArcX)	0.5259 ± 0.0079	Direction pretraining only	$\underline{0.6371 \pm 0.0092}$
w/o ArcX pretraining	0.5692 ± 0.0074	ArcX	$\mathbf{0.7036 \pm 0.0079}$

6 Conclusion

In this work, we introduce pair-level VSRL as a complementary perspective to existing entity- and scene-level VSRL, and propose ArcX to explicitly focus on spatial relations between entity pairs. ArcX performs arc-wise cross-entity modulation and produces dedicated relation embeddings that jointly capture multiple aspects of spatial relations, including distance, direction, and topology. These embeddings are further integrated with TDSK to modulate information exchange between neighboring entities, allowing the same relation representation to support pairwise reasoning, entity-level neighborhood aggregation, and scene-level information propagation.

We evaluate ArcX across five datasets covering all three levels of VSRL. The results show that ArcX can represent spatial relations and provide consistent benefits across diverse entity types, task settings, and spatial scales. Our findings suggest that explicitly modeling spatial relations can serve as an effective bridge between entity- and scene-level VSRL, enabling spatial representations to better exploit the interactions among entities rather than treating them in isolation.

AI USE STATEMENT

Generative AI tools were used to assist with manuscript editing, translation, formatting, data processing, and experimental analysis. They were also used to provide feedback on the presentation and interpretation of the experiments. All AI-assisted outputs were reviewed, verified, and, where necessary, corrected by the authors. The authors take full responsibility for the final content, analyses, and conclusions of this work.

REFERENCES

Marianne Abémgnigni Njifon and Dominic Schuhmacher. Graph convolutional networks for spatial interpolation of correlated data. *Spatial Statistics*, 60:100822, 2024. doi: 10.1016/j.spasta.2024.100822.

Gabriel Appleby, Linfeng Liu, and Li-Ping Liu. Kriging convolutional networks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 3187–3194, 2020. doi: 10.1609/aaai.v34i04.5716.

Chen Chu and Cyrus Shahabi. Geo2Vec: Shape- and distance-aware neural representation of geospatial entities. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 18985–18993, 2026. doi: 10.1609/aaai.v40i23.38970.

Matthias Demuzere, Jonas Kittner, Alberto Martilli, Gerald Mills, Christian Moede, Iain D. Stewart, Jasper van Vliet, and Benjamin Bechtel. A global map of local climate zones to support earth system modelling and urban-scale environmental science. *Earth System Science Data*, 14(8): 3835–3873, 2022. doi: 10.5194/essd-14-3835-2022.

Max J. Egenhofer and Robert D. Franzosa. Point-set topological spatial relations. *International Journal of Geographical Information Systems*, 5(2):161–174, 1991. doi: 10.1080/02693799108927841.

Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. VectorNet: Encoding HD maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11525–11533, 2020.

Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1263–1272, 2017. URL https://proceedings.mlr.press/v70/gilmer17a.html.

Chiyu Max Jiang, Dana Lansigan, Philip Marcus, and Matthias Nießner. DDSL: Deep differentiable simplex layer for learning geometric signals. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8769–8778, 2019.

Jonas Jongejan, Henry Rowley, Takashi Kawashima, Jongmin Kim, and Nick Fox-Gieg. The quick, draw! a.i. experiment. https://quickdraw.withgoogle.com/, 2017.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017. URL https://openreview.net/forum?id=SJU4ayYgl.

Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015. doi: 10.1126/science.aab3050.

Bernhard Lehner and Günther Grill. Global river hydrography and network routing: Baseline data and new approaches to study the world's large river systems. *Hydrological Processes*, 27(15): 2171–2186, 2013. doi: 10.1002/hyp.9740.

Lihuan Li, Hao Xue, Yang Song, and Flora Salim. T-JEPA: A joint-embedding predictive architecture for trajectory similarity computation. In *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems*, pp. 569–572. ACM, 2024. doi: 10.1145/3678717.3691271.

Xiucheng Li, Kaiqi Zhao, Gao Cong, Christian S. Jensen, and Wei Wei. Deep representation learning for trajectory similarity computation. In *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, pp. 617–628. IEEE, 2018. doi: 10.1109/ICDE.2018.00062.

Yi Li, Weiming Huang, Gao Cong, Hao Wang, and Zheng Wang. Urban region representation learning with OpenStreetMap building footprints. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 1363–1373, 2023. doi: 10.1145/3580305.3599538.

Gengchen Mai, Krzysztof Janowicz, Bo Yan, Rui Zhu, Ling Cai, and Ni Lao. Multi-scale representation learning for spatial feature distributions using grid cells. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=rJljdh4KDH.

Gengchen Mai, Krzysztof Janowicz, Yingjie Hu, Song Gao, Bo Yan, Rui Zhu, Ling Cai, and Ni Lao. A review of location encoding for GeoAI: Methods and applications. *International Journal of Geographical Information Science*, 36(4):639–673, 2022. doi: 10.1080/13658816.2021.2004602.

Gengchen Mai, Chiyu Jiang, Weiwei Sun, Rui Zhu, Yao Xuan, Ling Cai, Krzysztof Janowicz, Stefano Ermon, and Ni Lao. Towards general-purpose representation learning of polygonal geometries. *GeoInformatica*, 27(2):289–340, 2023. doi: 10.1007/s10707-022-00481-2.

Yansong Ning, Hao Liu, Hao Wang, Zhenyu Zeng, and Hui Xiong. UUKG: Unified urban knowledge graph dataset for urban spatiotemporal prediction. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Ethan Perez, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron Courville. FiLM: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11671.

Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006. URL https://gaussianprocess.org/gpml/.

Leo Sampaio Ferraz Ribeiro, Tu Bui, John Collomosse, and Moacir Ponti. Sketchformer: Transformer-based representation for sketched structure. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 14153–14162, 2020.

Maria Despoina Siampou, Jialiang Li, John Krumm, Cyrus Shahabi, and Hua Lu. Poly2Vec: Polymorphic Fourier-based encoding of geospatial objects for GeoAI applications. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 55511–55532, 2025. URL https://proceedings.mlr.press/v267/siampou25a.html.

Martin Simonovsky and Nikos Komodakis. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3693–3702, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ.

Alexander Wang, Mengye Ren, and Richard Zemel. SketchEmbedNet: Learning novel concepts by imitating drawings. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10870–10881. PMLR, 2021.

Andrew Wilson and Ryan Adams. Gaussian process kernels for pattern discovery and extrapolation. In *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pp. 1067–1075, 2013. URL https://proceedings.mlr.press/v28/wilson13.html.

Xiongfeng Yan, Tinghua Ai, Min Yang, and Xiaohua Tong. Graph convolutional autoencoder model for the shape coding and cognition of buildings in maps. *International Journal of Geographical Information Science*, 35(3):490–512, 2021. doi: 10.1080/13658816.2020.1768260.

Dazhou Yu, Yuntong Hu, Yun Li, and Liang Zhao. PolygonGNN: Representation learning for polygonal geometries with heterogeneous visibility graph. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 4012–4022, 2024. doi: 10.1145/3637528.3671738.

Wentao Zhang, Jingyuan Wang, Yifan Yang, and Leong Hou U. VecCity: A taxonomy-guided library for map entity representation learning. *Proceedings of the VLDB Endowment*, 18(8): 2575–2588, 2025. doi: 10.14778/3742728.3742749.

Zhongyue Zhang, Guangyin Jin, Yuxuan Liang, Suwan Yin, and Yuankai Wu. AdaKernel: Learning adaptive kernel parameters for spatiotemporal graph neural networks. arXiv preprint arXiv:2606.01283, 2026.

Appendix

A ArcSet Computation Process

ArcSet is an entity-level VSRL encoder that uses arcs as the encoding units of a spatial entity. The resulting token set preserves arc-level resolution and is used by ArcX for cross-entity interaction.

A.1 Arc Primitive Construction

Given a spatial entity E represented by ordered vertices $\{\mathbf{v}_i\}$, each pair of adjacent vertices defines an arc primitive. Multi-part geometries are handled by taking the union of the arc sets from all components.

For an arc a_i with endpoints $\mathbf{v}_i = (x_i, y_i)$ and $\mathbf{v}_{i+1} = (x_{i+1}, y_{i+1})$, we compute its midpoint, length, and orientation as

$$\begin{aligned} \mathbf{p}_i &= \frac{1}{2}\left(\mathbf{v}_i + \mathbf{v}_{i+1}\right), \\ \ell_i &= \left\|\mathbf{v}_{i+1} - \mathbf{v}_i\right\|_2, \\ \theta_i &= \operatorname{atan2}\left(y_{i+1} - y_i,\, x_{i+1} - x_i\right). \end{aligned} \tag{15}$$

To describe the local geometric transition around each arc, we further compute the signed turning angles with respect to its previous and next adjacent arcs:

$$\Delta_i^- = \operatorname{wrap}\left(\theta_i - \theta_{i-1}\right), \qquad \Delta_i^+ = \operatorname{wrap}\left(\theta_{i+1} - \theta_i\right), \tag{16}$$

where $\operatorname{wrap}(\cdot)$ maps an angle to $[-\pi, \pi)$. For open endpoints, the turning angle is marked as missing.

Each arc is therefore characterized by

$$X(a_i) = \{\mathbf{p}_i,\, \ell_i,\, \theta_i,\, \Delta_i^-,\, \Delta_i^+\}. \tag{17}$$

We treat a point as a degenerate arc and zero-fill its unavailable length and angular features.

A.2 Arc Feature Encoding

ArcSet encodes the position, length, orientation, and turning information of each arc using Fourier and angular encodings.

For a scalar s, the Fourier encoding is defined as

$$\gamma(s) = \left[s,\, \sin(\omega_0 s),\, \cos(\omega_0 s),\, \ldots,\, \sin(\omega_{K-1} s),\, \cos(\omega_{K-1} s)\right], \tag{18}$$

where $\omega_k = 2^k \pi$.

For a two-dimensional coordinate $\mathbf{p} = (x, y)$, we additionally compute $r = \sqrt{x^2 + y^2}$ and encode its spatial position as

$$\gamma_{xyr}(\mathbf{p}) = [x,\, y,\, r,\, \Gamma(x),\, \Gamma(y),\, \Gamma(r)], \tag{19}$$

where

$$\Gamma(q) = [\sin(\omega_0 q),\, \cos(\omega_0 q),\, \ldots,\, \sin(\omega_{K-1} q),\, \cos(\omega_{K-1} q)]. \tag{20}$$

Angular variables are encoded using harmonic functions:

$$\alpha(\phi) = [\cos\phi,\, \sin\phi,\, \cos 2\phi,\, \sin 2\phi,\, \ldots,\, \cos H\phi,\, \sin H\phi]. \tag{21}$$

We use $H = 2$ in our implementation.

The position, length, and directional features of arc a_i are then

$$\mathbf{p}_i^{\mathrm{emb}} = \gamma_{xyr}(\mathbf{p}_i), \tag{22}$$

$$\boldsymbol{\ell}_i^{\mathrm{emb}} = \gamma(\ell_i), \tag{23}$$

$$\mathbf{d}_i^{\mathrm{emb}} = [\alpha(\theta_i) \| \alpha(\Delta_i^-) \| \alpha(\Delta_i^+)]. \tag{24}$$

These features are concatenated to form the geometric representation of the arc:

$$\mathbf{e}_i = [\mathbf{p}_i^{\text{emb}} \| \boldsymbol{\ell}_i^{\text{emb}} \| \mathbf{d}_i^{\text{emb}}]. \tag{25}$$

A shared projection network maps each arc representation into an arc token:

$$\mathbf{h}_i = \phi_\theta(\mathbf{e}_i) \in \mathbb{R}^{d_a}. \tag{26}$$

A.3 Arc-Token Interaction

For an entity containing n arcs, the projected arc tokens are stacked as

$$\mathbf{H}_E = [\mathbf{h}_1; \ldots; \mathbf{h}_n] \in \mathbb{R}^{n \times d_a}. \tag{27}$$

ArcSet applies self-attention over the token set so that each arc can exchange information with other arcs belonging to the same entity:

$$\mathbf{A}_E = \mathrm{ST}_\theta\left(\mathbf{H}_E\right) \in \mathbb{R}^{n \times d_a}, \tag{28}$$

where ST_θ denotes the self-attention-based set encoder. Padding introduced for variable-length batching is masked during attention.

The resulting matrix $\mathbf{A}_E$ contains one contextualized embedding for each arc and corresponds to the ArcSet token representation used in the main paper:

$$\mathbf{A}_E = \mathrm{ArcSet}_{\text{token}}(E). \tag{29}$$

When an entity-level representation is required, the valid arc tokens can further be aggregated through a permutation-invariant readout:

$$\mathbf{e}_E = \rho_\theta\left(\frac{1}{n}\sum_{i=1}^{n} \mathbf{A}_{E,i}\right), \tag{30}$$

where ρ_θ is an MLP. For ArcX, however, we retain the arc-token representations before this entity-level readout so that cross-entity interaction can be performed at the arc level.

B Representative Entity-Level Results of ArcSet

To demonstrate ArcSet's capability as an entity-level VSRL encoder, we report representative results on four datasets. We compare ArcSet against baselines from different modeling paradigms. For polygon-based tasks, the baselines include Geo2Vec, Poly2Vec, PolygonGNN, NUFTspec, ResNet1D, and Vertex-Edge GNN. For polyline-based tasks, we compare against Poly2Vec, T2Vec (Li et al., 2018), T-JEPA (Li et al., 2024), SketchEmbedNet (Wang et al., 2021), and Sketchformer (Ribeiro et al., 2020). These baselines cover both geometry-specific and unified vector representation approaches, providing comparisons across different input structures and encoding mechanisms.

Buildings. Buildings (Yan et al., 2021) contains 5,010 real-world building footprints extracted from OpenStreetMap and labeled into ten letter-shape classes. We use a 70/15/15 train/validation/test split.

MNIST. MNIST contains 60,000 MNIST images converted to polygon entities. We use a 70/15/15 train/validation/test split.

Omniglot. Omniglot (Lake et al., 2015) contains Omniglot samples, where handwritten character strokes are treated as multi-polylines. We use the published LAKE background/evaluation split, with 964 classes for training and 659 classes for testing, for episodic evaluation.

QuickDraw100. QuickDraw100 is a truncated subset of the full Quick, Draw! dataset (Jongejan et al., 2017). We choose 100 classes for a total of 892,373 entities, with per-class counts ranging from 6,995 to 9,810; we do not use the full 345-class release. We use a 70/15/15 train/validation/test split.

Table 7: Classification results on Buildings and MNIST. The best result in each column is bold.

Method	Buildings	MNIST
ArcSet	$\mathbf{97.50 \pm 0.76}$	$\mathbf{98.46 \pm 0.13}$
Geo2Vec	97.34 ± 0.31	97.58 ± 0.10
ResNet1D	96.72 ± 0.27	97.47 ± 0.10
Vertex-Edge GNN	94.77 ± 0.47	97.80 ± 0.10
Poly2Vec	81.99 ± 1.41	96.11 ± 0.11
PolygonGNN	94.01 ± 0.52	91.88 ± 0.85
NUFTspec	85.51 ± 0.88	96.46 ± 0.15

Table 8: Low-shot classification results on QuickDraw100. The best result in each column is bold.

Method	QD-50	QD-100	QD-200	QD-500	QD-1000
ArcSet	$\mathbf{52.45 \pm 0.73}$	$\mathbf{62.86 \pm 0.68}$	$\mathbf{68.34 \pm 0.44}$	$\mathbf{74.45 \pm 0.31}$	$\mathbf{78.08 \pm 0.24}$
Poly2Vec	24.99 ± 0.30	31.22 ± 0.36	36.79 ± 0.17	42.28 ± 0.19	45.69 ± 0.18
T2Vec	46.88 ± 0.22	56.19 ± 0.55	64.07 ± 0.25	71.51 ± 0.15	75.81 ± 0.08
T-JEPA	37.65 ± 0.40	46.51 ± 0.33	54.40 ± 0.45	64.70 ± 0.75	71.55 ± 1.49
SketchEmbedNet	33.75 ± 0.44	42.58 ± 0.49	51.56 ± 0.38	60.04 ± 1.04	65.78 ± 0.51
Sketchformer	31.01 ± 0.52	40.49 ± 0.64	48.97 ± 0.25	58.92 ± 0.16	66.67 ± 0.76

Table 9: Few-shot classification results on Omniglot. The best result in each column is bold.

Method	5w-1s	5w-5s	20w-1s	20w-5s
ArcSet	$\mathbf{95.52 \pm 0.20}$	$\mathbf{98.44 \pm 0.13}$	$\mathbf{90.08 \pm 0.41}$	$\mathbf{96.71 \pm 0.13}$
Poly2Vec	63.77 ± 0.58	79.79 ± 0.31	41.76 ± 0.19	61.46 ± 0.24
T2Vec	89.75 ± 0.16	96.24 ± 0.21	79.71 ± 0.70	92.38 ± 0.26
T-JEPA	91.89 ± 0.70	97.39 ± 0.23	83.16 ± 1.16	93.97 ± 0.52
SketchEmbedNet	95.05 ± 0.19	98.42 ± 0.11	87.31 ± 0.36	95.74 ± 0.20
Sketchformer	71.69 ± 3.41	88.45 ± 2.80	63.25 ± 1.63	77.71 ± 5.19

C Dataset and Task Details

This appendix describes the datasets used in the experiments and, for each task, how the samples are divided into training, validation and test subsets. All splits are fixed files that are shared by every method and are never regenerated by a training seed.

C.1 VecCity

VecCity is a library and benchmark for map-entity representation learning that provides processed city-scale geometry for point, polyline and polygon map entities. We use its Beijing and Xi'an geometry files: Beijing contains 132,694 geometries (81,180 points, 40,306 polylines and 11,208 polygons) and Xi'an contains 25,432 (19,107 points, 5,269 polylines and 1,056 polygons).

Pairwise spatial relation prediction. Each sample is a pair of geometries, and the model predicts three relations between them from the geometry alone: (i) *topology*, a four-way classification

(disjoint, crosses, containment, touches); (ii) *direction*, the orientation of the pair discretized into eight bins, defined for all six geometry-type pairs and evaluated using 8-bin classification accuracy and angular MAE from a continuous-angle prediction head; and (iii) *centroid distance*, a regression target defined for all six geometry-type pairs. Pairs are drawn from Beijing and split into training, validation and test subsets such that no geometry appears in more than one subset (the Beijing manifests reference 55,846 distinct geometries). Xi'an is used only as a zero-shot external test set: it contributes no training or validation pairs, and no checkpoint selection is performed on it (its manifests reference 8,716 distinct geometries). Table 10 lists the number of pairs in each subset.

Table 10: Number of geometry pairs per subset for the VecCity pairwise tasks.

Task	Beijing train	Beijing val	Beijing test	Xi'an test (zero-shot)
Topology (4-way classification)	12,726	450	513	2,196
Direction	12,000	3,000	3,000	3,000
Centroid distance (regression)	12,000	3,000	3,000	3,000

C.2 UUKG

UUKG is a unified urban knowledge graph dataset for urban spatio-temporal prediction (NeurIPS 2023 Datasets and Benchmarks Track) that links roads, points of interest and areas of several cities. We use only its Chicago (CHI) data with two regression tasks, each evaluated using RMSE:

- **Bike** (entity level). The sample is a road entity, and the target is the logarithm of its total bike inflow, $\log(1 + \text{inflow})$.
- **HUMOB** (entity level). The sample is a point-of-interest entity, and the target is the logarithm of its total human-mobility inflow, $\log(1 + \text{inflow})$.

The two entity-level tasks use different sets of labeled entities and independent splits (70%/15%/15% by entity), so their training sets are not shared. Table 11 lists the number of samples in each subset.

Table 11: Number of samples per subset for the UUKG-CHI Bike and HUMOB tasks.

City	Task (sample unit)	Train	Val	Test	Total
CHI	Bike (entity)	1,050	225	225	1,500
	HUMOB (entity)	700	150	150	1,000

C.3 HydroSHEDS

HydroSHEDS provides hydrographic data of river networks and drainage basins. Our benchmark is built from its basin polygons and river polylines: every basin is a region, and the river polylines that intersect it are the entities of that region. The benchmark contains 1,783 basins and 98,480 river polylines (99,757 basin–river memberships; between 6 and 926 polylines per basin, 37 at the median).

Basin classification. Each sample is a basin, and the task is a three-class classification whose label is the basin's overall flow direction; the three classes contain 774, 718 and 291 basins. The evaluation metric is accuracy. The split is fixed and contains 1,248 training, 178 validation and 357 test basins (Table 12).

C.4 Local Climate Zones

Local Climate Zones (LCZ) is a standardized classification of urban form and land cover that groups neighborhoods by their building morphology. Our LCZ benchmark consists of 867 LCZ regions in New York City and 163,441 building footprints, which are the entities (180,036 building–region memberships determined by geometric intersection; 5 to 608 buildings per region, 201 at the median).

Region classification. Each sample is a region, and the task is a six-class classification of its LCZ class; the classes contain 105, 394, 242, 66, 16 and 44 regions. The evaluation metric is accuracy. The split is fixed and contains 606 training, 87 validation and 174 test regions (Table 12).

C.5 Building Fragments

Building Fragments is derived from a single-building shape-classification dataset whose footprints are letter-like shapes from ten classes (E, F, H, I, L, O, T, U, Y and Z). Each original building footprint is treated as one region. Sixteen interior seed points define a clipped Voronoi partition of the footprint, and every cell is independently shrunk by a scale factor drawn from $[0.3, 0.7]$; the resulting separated polygons, the fragments, are the entities. The benchmark contains 5,010 regions and 80,160 fragments (exactly 16 per region). Coordinates are normalized, unit-less values, and every region is an independent local scene.

Building shape classification. Each sample is a region, and the task is a ten-class classification of the shape of the original building; every class contains exactly 501 regions. The evaluation metric is accuracy. The split is stratified by class and contains 3,507 training, 501 validation and 1,002 test regions (Table 12).

Table 12: Number of regions per subset for the three region-classification benchmarks.

Dataset	Classes	Entities	Train	Val	Test	Total
HydroSHEDS	3	98,480	1,248	178	357	1,783
Local Climate Zones	6	163,441	606	87	174	867
Building Fragments	10	80,160	3,507	501	1,002	5,010

Pretraining performance. We further report the performance of the distance and direction estimation tasks used to pretrain ArcX. Across the four datasets, ArcX achieves consistently high direction classification accuracy and low direction-angle error, while also providing accurate centroid-distance estimation under the corresponding coordinate systems. We do not include a topology prediction objective during pretraining because entity pairs with strong topological relations, such as intersection or containment, are relatively sparse in these datasets. Such labels are therefore highly imbalanced and provide substantially weaker supervision than distance and direction relations. We thus pretrain ArcX using only distance and direction estimation tasks.

Table 13: ArcX pretraining performance on the datasets used for downstream entity- and scene-level tasks. Direction is evaluated using 8-bin classification accuracy and angular MAE, while distance is evaluated using centroid-distance MAE.

Dataset	Unit	Direction Acc ↑	Direction MAE (°) ↓	Centroid Distance MAE ↓
UUKG-CHI	m	0.8745	1.369	4.473
HydroSHEDS	m	0.9148	2.023	271.5
Local Climate Zones (LCZ)	m	0.9282	1.521	0.975
Building Fragments	Normalized coordinates	0.9124	1.934	0.000105

These results indicate that the pretraining tasks provide ArcX with effective supervision for learning both directional and distance-related spatial relations before downstream entity- and scene-level tasks.

C.6 Construction of the Building Fragments Dataset

We construct the Building Fragments dataset from 5,010 real-world building footprints covering 10 balanced shape categories, with 501 buildings in each category. Each complete building footprint is treated as one region, and its shape category is used as the scene-level label.

For each building, we uniformly sample 16 points from its interior and use them as seeds to construct a Voronoi tessellation. The Voronoi cells are clipped by the original building boundary, producing 16

non-overlapping cells that collectively cover the complete footprint. Each cell is then independently contracted around its centroid using a randomly sampled scale factor from 0.3 to 0.7, retaining 30%–70% of its original linear dimensions. The contracted geometries are clipped again by the original footprint to prevent boundary leakage. The complete construction process is illustrated in Figure 3.

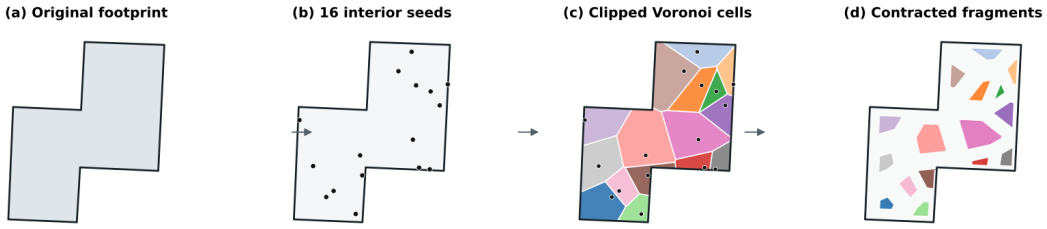

Figure 3: Construction of the Building Fragments dataset. From left to right: an original building footprint, 16 points uniformly sampled from its interior, the Voronoi tessellation clipped to the building boundary, and the final spatially separated fragments obtained by independently contracting each cell with a scale factor between 0.3 and 0.7. Colors indicate the correspondence between Voronoi cells and contracted fragments.

This procedure produces 16 spatially separated entities for each building and 80,160 entities in total. The complete building footprint is not provided as model input. The resulting scene-level task requires predicting the shape category of the original building solely from the geometry and spatial arrangement of its fragments.

For location encoding, all fragments belonging to the same building are transformed using a shared coordinate system. The center of the original building's bounding box is translated to the origin, and its longer bounding-box dimension is scaled to span $[-1, 1]$. This shared transformation preserves the relative positions, orientations, and spatial arrangement of the 16 fragments. We encode fragment shape and normalized location separately as 64-dimensional Geo2Vec embeddings and concatenate them into a 128-dimensional entity representation.

D Experimental Settings

All results reported with $\pm$ denote the mean $\pm$ population standard deviation over three random seeds, rather than the standard error. The training, validation, and test splits are fixed and are not affected by the random seed. Model selection is performed using only the validation set. No learning-rate scheduling, gradient clipping, or mixed-precision training is used in any experiment.

D.1 Model Architecture

Table 14: ArcX architecture. The architecture is identical for Part I, dataset-specific pretraining, and frozen downstream use.

Component	Setting
Arc token	Number of Fourier frequencies $K = 6$, with $\omega_k = 2^k\pi$, $k = 0, \ldots, 5$.
Arc embedding ϕ	2-layer MLP with width $d_a = 128$.
Intra-entity attention	One self-attention block (SAB), 4 attention heads, FFN expansion ratio 2, pre-LayerNorm, and residual connections.
Entity pooling	Mean pooling $\rightarrow$ 2-layer MLP $\rho \rightarrow$ 128-dimensional entity embedding.
Cross-entity branch	Linear projection to $d_c = d_a/4 = 32$; arc-wise scalar gate implemented as a 2-layer MLP with 64-dimensional input $[q_i; c_j]$, hidden width 32, and sigmoid output. Here, c_j is the mean of the projected arc tokens of the other entity. Cross-entity responses are aggregated using max pooling.
Relation fusion	3-layer MLP taking two entity embeddings and two cross-branch vectors as input, with total input dimension $2 \times 128 + 2 \times 32 = 320$, followed by $128 \rightarrow 128 \rightarrow d_r = 128$.
Prediction heads	2-layer MLP ($128 \rightarrow 128 \rightarrow$ output); the direction head outputs a 2-D vector followed by L2 normalization, and the distance head outputs a scalar.
Dropout	0.1 during training; disabled during inference and frozen downstream use.
Parameter count	0.29M encoder parameters.

ArcSet uses the same architecture except that it does not include the cross-entity branch. Its relation-fusion input therefore consists only of the two 128-dimensional entity embeddings, resulting in a 256-dimensional input.

D.2 Pairwise Relation Prediction

Topology, direction, and centroid-distance prediction are jointly trained for both ArcSet and ArcX.

Table 15: Training settings for Part I pairwise relation prediction.

Item	Setting
Optimizer	AdamW with weight decay 10^{-5}.
Learning rate	10^{-3}.
Epochs	20 epochs, with 200 steps per epoch, for a total of 4,000 updates.
Batch	128 pairs per task per update.
Prediction heads	Topology: 2-layer MLP ($128 \rightarrow 128 \rightarrow$ class logits); direction: 2-layer MLP ($128 \rightarrow 128 \rightarrow 2$) followed by L2 normalization; centroid distance: 2-layer MLP ($128 \rightarrow 128 \rightarrow 1$).
Loss	Topology: cross-entropy; direction: $1 - \cos$; centroid distance: Huber loss ($\delta = 1$) on standardized $\log(1+d)$. The three losses are equally weighted. Predictions from the two input orders (A, B) and (B, A) are averaged.

D.3 ArcX Pretraining

ArcX is pretrained separately on each downstream dataset.

Table 16: ArcX relation-pretraining settings on downstream datasets.

Item	Setting
Optimizer	AdamW with weight decay 10^{-5}.
Learning rate	10^{-3}.
Epochs	Up to 20 epochs, with 200 steps per epoch.
Batch	One direction batch and one distance batch per update, each containing 128 pairs.
Training pairs	All directed edges in the k-nearest-neighbor graph constructed within the training regions.
Prediction heads	Direction: 2-layer MLP ($128 \to 128 \to 2$) followed by L2 normalization; centroid distance: 2-layer MLP ($128 \to 128 \to 1$).
Loss	$L = 2L_{\mathrm{dir}} + L_{\mathrm{dist}} + 0.1L_{\mathrm{cons}}$. $L_{\mathrm{dir}} = 1 - \cos$, where AB is supervised by u and BA by $-u$, and the two orders are averaged. L_{dist} is the Huber loss ($\delta = 1$) on standardized $\log(1 + d_{\mathrm{cent}})$, averaged over the two input orders. $L_{\mathrm{cons}} = \frac{1}{2}\left[(1 - \cos(\hat{u}_{AB}, -\hat{u}_{BA})) + \mathrm{Huber}(\hat{c}_{AB}, \hat{c}_{BA})\right]$.

D.4 Downstream Training

Table 17: Training settings for region-classification tasks.

Item	Setting
Optimizer	AdamW with weight decay 10^{-4}.
Learning rate	10^{-3}.
Epochs	100.
Batch	32 regions per batch.
Prediction head	Mean pooling over node states within each region $\to$ Linear$(d, 64)$ $\to$ ReLU $\to$ Linear$(64,$ number of classes$)$.
Loss	Class-weighted cross-entropy, where class weights are inversely proportional to class frequencies and normalized to have mean 1.

Table 18: Training settings for UUKG-CHI bike and human-mobility regression.

Item	Setting
Optimizer	AdamW with weight decay 10^{-4}.
Learning rate	10^{-3}.
Epochs	100; each optimization step processes one area graph.
Prediction head	Target-entity node state $\to$ Linear$(d, 64)$ $\to$ ReLU $\to$ Linear$(64, 1)$.
Loss	Mean squared error on $\log(1 + \mathrm{inflow})$.

D.5 Neighborhood Construction

Table 19: Neighborhood construction settings for downstream datasets.

Dataset	k	Distance unit
HydroSHEDS	8	meters
LCZ	8	meters, converted from U.S. survey feet
Building Pattern	4	normalized coordinates (unitless)
UUKG-CHI (bike, human)	8	meters

For all datasets, neighborhoods are constructed as

$$\mathcal{N}(i) = \{j \neq i : j \in \mathrm{kNN}(i) \text{ or } i \in \mathrm{kNN}(j)\}, \tag{31}$$

where $\mathrm{kNN}(i)$ denotes the k entities with the smallest centroid Euclidean distances to entity i. Ties are broken by entity index.

The resulting graph is symmetric and contains no self-loops. A separate graph is constructed for each region (area), with no edges between different regions. The graph is constructed once before training and remains fixed throughout training. For regions containing at most $k + 1$ entities, the graph is fully connected. If an entity belongs to multiple regions, as in HydroSHEDS and LCZ, it appears independently in the graph of each corresponding region.