GeoAggregator: An Efficient Transformer Model for Geo-Spatial Tabular Data

Rui Deng[1], **Ziqi Li**[2], **Mingshu Wang**[1]*

[1]School of Geographical and Earth Science, University of Glasgow
[2]Department of Geography, Florida State University
{rui.deng, mingshu.wang}@glasgow.ac.uk, ziqi.li@fsu.edu

Abstract

Modeling geospatial tabular data with deep learning has become a promising alternative to traditional statistical and machine learning approaches. However, existing deep learning models often face challenges related to scalability and flexibility as datasets grow. To this end, this paper introduces GeoAggregator, an efficient and lightweight algorithm based on transformer architecture designed specifically for geospatial tabular data modeling. GeoAggregators explicitly account for spatial autocorrelation and spatial heterogeneity through Gaussian-biased local attention and global positional awareness. Additionally, we introduce a new attention mechanism that uses the Cartesian product to manage the size of the model while maintaining strong expressive power. We benchmark GeoAggregator against spatial statistical models, XGBoost, and several state-of-the-art geospatial deep learning methods using both synthetic and empirical geospatial datasets. The results demonstrate that GeoAggregators achieve the best or second-best performance compared to their competitors on nearly all datasets. GeoAggregator's efficiency is underscored by its reduced model size, making it both scalable and lightweight. Moreover, ablation experiments offer insights into the effectiveness of the Gaussian bias and Cartesian attention mechanism, providing recommendations for further optimizing the GeoAggregator's performance.

Code and Data —
https://github.com/ruid7181/GeoAggregator

1 Introduction

Geospatial data are increasingly available with the widespread deployment of GPS-enabled sensors and the growing demand for location-based services (Luo, Liu, and Liu 2021; Stewart et al. 2022). Geospatial data modeling is crucial in natural and social sciences to understand intricate spatial relationships, predict future spatial scenarios, and inform decision-making. Its applications span various fields, including public health, environmental science, urban and regional planning, among others (Karimi and Karimi 2017). To date, geospatial studies have primarily relied on spatial and geo-statistical models. Although these models explicitly account for spatial effects, such as spatial autocorrelation (SA) and spatial heterogeneity (SH), they often depend on strong data and model assumptions. Besides, their effectiveness diminishes when handling large volumes of data or modeling complex non-linearity due to their high computational complexity and linear nature (Li 2022).

In addition to classic methods, efforts have been broadened to cover a wide range of deep learning paradigms (Jiang 2019). By explicitly defining a graph to represent the data points and the underlying spatial structure, the problems can be approximated to node-level learning tasks. Graph-based networks can be trained for node regression and classification (Zhu et al. 2020; Wu et al. 2021; Zhu et al. 2022). Alternatively, with a spatial proxy grid proposed by (Dai et al. 2022), convolutional networks can work on irregular grids and effectively model SH in spatial regression. On top of these approaches, ensemble learning methods have also been applied to further improve predictive performance (Cheng et al. 2024). However, the above models are typically associated with large input sizes and a considerable number of parameters, which makes them scale poorly on large datasets.

Lately, transformer models have demonstrated their effectiveness in handling irregular grids (Lee and Oh 2024), point clouds (Zhao et al. 2021) as well as geospatial datasets (Li et al. 2023; He et al. 2023; Jia et al. 2024; Unlu 2024). The attention mechanism selectively focuses on elements in a sequence and aggregates the information according to relation-based attention scores. This approach brings new horizons for geospatial data modeling in that it accounts for contextual information based on proximity, and is sensitive to positional information (Vaswani 2017; Su et al. 2024; Vivanco Cepeda, Nayak, and Shah 2024). That is, it allows information aggregation from neighboring points to learn SA patterns while remaining aware of SH patterns across the global space. Besides, transformers represent a group of modern architectures that can serve as a general-purpose feature extractor for multiple tasks and multi-modal fusion (Jaegle et al. 2021; Xu, Zhu, and Clifton 2023).

However, the well-known problem of computational and time complexity increasing quadratically with input sequence length limits the application of transformers, especially in geospatial contexts, where the size of real-world

*Corresponding author

datasets varies dramatically in different scenarios. To overcome this issue, efforts have focused on optimizing the self-attention mechanism and transformer architecture (e.g., (Jaegle et al. 2021; Dao et al. 2022)). Nevertheless, efficiency improvements in the context of geospatial tabular data remain limited. Moreover, most models do not explicitly incorporate geographical priors to better account for geospatial effect,s including SA and SH. In this work, we propose GeoAggregator, an efficient and lightweight transformer model enhanced by geographical priors for geospatial tabular data modeling. We demonstrate the effectiveness of GeoAggregator in various geospatial regression tasks.

2 Background

2.1 Modeling Spatial Effects

SA and SH are two main spatial effects that govern the distribution and interaction of spatial data (Anselin 2010). SA refers to the phenomenon where spatial data exhibit spatial clustering, meaning that similar values are observed in close geographic proximity. On the other hand, SH refers to the differences in spatial patterns and processes at different locations. This means that relationships or patterns observed in one place may not be the same in another. These differences are often due to local contexts, such as varying environmental and socio-economic conditions, and cultural and policy influences that are hard to measure or quantify (Fotheringham and Li 2023).

One common approach to express SA is to use spatially lagged variables to capture the dependency with nearby values when modeling a target location. One can formally assign a spatial weight between pairs of N locations to build a spatial weight matrix $\mathbf{W} \in \mathbb{R}^{N \times N}$ to characterize the degree of spatial interactions. Then, a spatial lag term can be expressed as:

$$\tilde{\mathbf{y}} = \rho \mathbf{W} \mathbf{y} \tag{1}$$

where $\mathbf{y} \in \mathbb{R}^N$ is a vector containing N data points, ρ is a parameter to control the strength of SA. It can be seen that determining optimized $\mathbf{W}$ and ρ is crucial for properly expressing SA.

SH is often addressed using *local* modeling approaches. The most well-known example is Geographically Weighted Regression (GWR), which allows regression parameters to be location-specific (Fotheringham, Brunsdon, and Charlton 2003):

$$y_i = \mathbf{x}_i \boldsymbol{\beta}_i + \epsilon_i \tag{2}$$

where $i = 0, 1, \ldots, N-1$; $\boldsymbol{\beta}_i \in \mathbb{R}^p$ is the corresponding p regression coefficients, which can be estimated by:

$$\hat{\boldsymbol{\beta}}_i = [\mathbf{X}^T \mathbf{W}_i \mathbf{X}]^{-1} \mathbf{X}^T \mathbf{W}_i \mathbf{Y} \tag{3}$$

where $\mathbf{X} = [\mathbf{x}_1^T, \mathbf{x}_2^T, \ldots, \mathbf{x}_N^T]^T$, $\mathbf{W}_i = diag(w_{i,1}, w_{i,2}, \ldots, w_{i,N})$ is a spatial weight matrix. Note that $w_{i,j}$ can be given by a stationary kernel function such as:

$$w_{i,j} = K(d_{i,j}) = e^{-\frac{1}{2}(\frac{d_{i,j}}{\gamma})^2} \tag{4}$$

based on the distance $d_{i,j}$ between two points, scaled by a bandwidth γ.

2.2 Problem Statement

We consider the following geospatial regression problem. Given a set $\mathcal{P}_c$ of data points in a 2D continuous geographical space. Each point (referred to as contextual point) is located by a 2D spatial location vector $\mathbf{l}^c \in \mathbb{R}^2$ while associated with m covariates organized as $\mathbf{x}^c \in \mathbb{R}^m$ and a target variable $y^c \in \mathbb{R}$. We denote the i-th contextual points as $\boldsymbol{p}_i^c = (\mathbf{x}_i^c, \mathbf{l}_i^c, y_i^c)$. For another set $\mathcal{P}_t$ of data points in the same 2D space, m covariates $\mathbf{x}^t \in \mathbb{R}^m$ are observed, but the target variable y^t is missing, i.e., $\mathcal{P}_t = \{\boldsymbol{p}_j^t\} = \{(\mathbf{x}_j^t, \mathbf{l}_j^t)\}$. Note that a point set can be organized in a tabular format with $|\mathcal{P}|$ rows and $m+3$ attributes.

We define a mapping *ContextQuery* : $\mathcal{P}_t \rightarrow \mathcal{N}_r(\mathcal{P}_c)$, where $\mathcal{N}_r \subseteq \mathcal{S}(\mathcal{P}_c)$ denotes a collection of subsets of $\mathcal{P}_c$ determined by spatial proximity. For each $\boldsymbol{p}^t \in \mathcal{P}_t$, the mapping is a subset of $\mathcal{P}_c$ consisting of neighboring points of $\boldsymbol{p}^t$ within a query radius r.

We assume target variable y is generated by unknown processes that potentially exhibit a mixture of SA and SH effects. The aim is to learn to predict the unobserved target variable y_j^t of $\boldsymbol{p}_j^t$ as a function of 1) covariates and spatial location of $\boldsymbol{p}_j^t$; and 2) covariates, spatial location and target variable of all $\boldsymbol{p}_i^c \in$ *ContextQuery*$(\boldsymbol{p}_j^t)$.

3 Approaches

This section introduces the proposed GeoAggregator model for geospatial regression tasks. Figure 2 provides a high-level overview of the architecture.

3.1 Input Sequence

GeoAggregator takes in a sequence of points with a maximum length of ℓ_{max}. For each target point $\boldsymbol{p}_j^t \in \mathcal{P}_t$, we use the set $\boldsymbol{h}_j^t =$ *ContextQuery*$(\boldsymbol{p}^t)$ as the corresponding input sequence and adjust the length of the sequence by clipping and padding to match ℓ_{max}:

$$\boldsymbol{h}_j^{in} = \begin{cases} (\boldsymbol{h}_j^t)_{0:\ell_{max}} & \text{if } \left|\boldsymbol{h}_j^t\right| > \ell_{max} \\ \text{ZeroPad}_{\ell_{max}}(\boldsymbol{h}_j^t) & \text{if } \left|\boldsymbol{h}_j^t\right| \leq \ell_{max} \end{cases} \tag{5}$$

where $\text{ZeroPad}_{\ell_{max}}(\cdot)$ is the operation to pad a sequence with 0 to the length of ℓ_{max}. Note that the clipping operation randomly removes contextual points since the order of the points in $\boldsymbol{h}_j^t$ is random.

We then generate a mask sequence $\boldsymbol{m}^{in}$ for each target point $\boldsymbol{p}_j^t$ to ensure that the encoder attends only to the non-zero points:

$$m_k^{in} = \begin{cases} 0 & \text{if } 0 \leq k < \min(\left|\boldsymbol{h}_j^t\right|, \ell_{max} - 1) \\ 1 & \text{if } \min(\left|\boldsymbol{h}_j^t\right|, \ell_{max} - 1) \leq k < \ell_{max} \end{cases} \tag{6}$$

We denote the actual number of contextual points in the input sequence as $\ell_{in} = \ell_{max} - \sum_k m_k^{in}$.

3.2 Feature Projection

We first use a slightly modified batch normalization to normalize each covariate within a mini-batch (Bjorck et al. 2018). Note that we only use unmasked points to calculate the mean/standard deviation values.

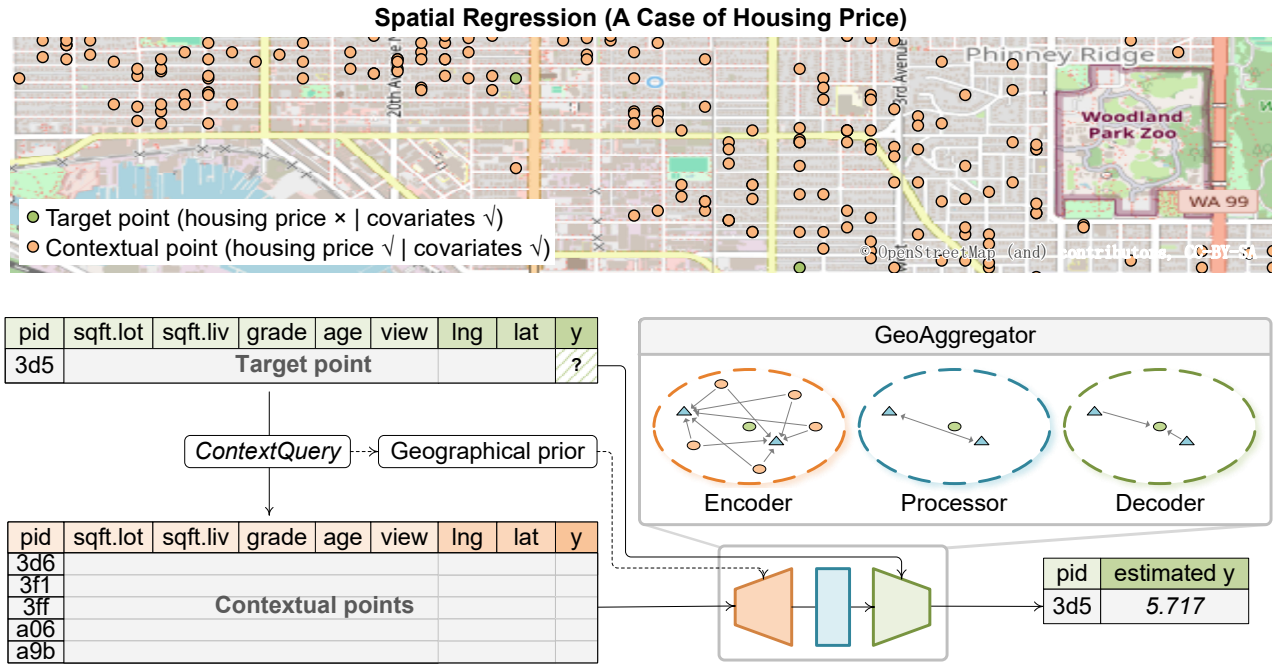

Figure 1: An illustration of the geospatial regression workflow. Each data point has several covariates and a spatial location. The target variable (the housing price in this case) is observed only for *part* of the points. Using aggregated neighborhood information, we propose an encoder-processor-decoder architecture to predict unobserved target variables.

We use two parallel dense networks with Tanhshrink activation for feature projection. Covariates ($\mathbf{x}^c$ and $\mathbf{x}^t$) and the target variable (y^c) are projected into higher dimensional feature space separately. This can be written as:

$$\mathbf{e}_x = \text{Dense}_x(\mathbf{x}^{\{t,c\}}) \in \mathbb{R}^{\frac{d_{model}}{2}} \tag{7}$$

$$\mathbf{e}_y = \text{Dense}_y(y^c) \in \mathbb{R}^{\frac{d_{model}}{2}} \tag{8}$$

where d_{model} is the dimension of the latent embedding in the GeoAggregator model. Since y^t is unknown, we use a learnable feature vector instead.

3.3 2D Rotary Positional Embedding

Due to the inherent permutation invariance, positional information must be explicitly injected into the attention mechanism. Inspired by (Su et al. 2024), (Mai et al. 2020), and (Unlu 2024), we use an augmented rotation matrix to incorporate 2D spatial locations into the embedding features.

We first propose the sinusoidal representation features for this purpose. For a data point $\boldsymbol{p}$ at the spatial location $\mathbf{l} = (l_1, l_2)$, the representation feature at scale s is $\varphi_s = \{\cos(l_1\theta_s), \sin(l_1\theta_s), cos(l_2\theta_s), \sin(l_2\theta_s)\}$, where $\theta_s = 10000^{\frac{2-2s}{d}}$; $s = 0, 1, \ldots, S-1$; $S = \frac{d}{4}$ and d is some embedding dimension. We construct 4-by-4 2D rotation matrices based on $\mathbf{\Phi}_s$ that rotate the embedding feature vectors alternatively in the two dimensions of $\mathbf{l}$:

$$\mathbf{\Phi}_s = \begin{pmatrix} \cos(l_{i,1}\theta_s) & -\sin(l_{i,1}\theta_s) & 0 & 0 \\ \sin(l_{i,1}\theta_s) & \cos(l_{i,1}\theta_s) & 0 & 0 \\ 0 & 0 & \cos(l_{i,2}\theta_s) & -\sin(l_{i,2}\theta_s) \\ 0 & 0 & \sin(l_{i,2}\theta_s) & \cos(l_{i,2}\theta_s) \end{pmatrix} \tag{9}$$

The d-by-d rotation matrix can then be constructed by placing the rotation matrices at each scale along the diagonal:

$$\mathbf{\Phi} = diag(\mathbf{\Phi}_0, \mathbf{\Phi}_1, \cdots, \mathbf{\Phi}_S) \in \mathbb{R}^{d \times d} \tag{10}$$

We inject the spatial location information into embedding features produced in section 3.3 through a matrix multiplication (Su et al. 2024):

$$\tilde{\mathbf{e}} = \mathbf{\Phi}\mathbf{e} \tag{11}$$

3.4 Cartesian Product Attention

The learnable parameters in attention are mainly contributed by the linear projections, which produce Query (Q), Key (K), and Value (V) vectors with rich information. Consequently, as the embedding dimension and the number of layers increase, the model size scales rapidly. This poses challenges for even moderately sized geospatial datasets. To address this, we propose Multi-head Cartesian Product Attention (MCPA) to help manage the learnable parameters and computational complexity while maintaining the expressive power as much as possible.

As shown in Figure 2, our MCPA takes two input embeddings from the feature projection: $\tilde{\mathbf{e}}_x$ from covariates

and $\tilde{\mathbf{e}}_y$ from the target variable. Like in the feature projection, we employ parallel linear projections $\mathbf{W}_{hx}^{\{Q_x,K_x,V_x\}} \in \mathbb{R}^{\frac{d_{model}}{2}\times d_c}$ and $\mathbf{W}_{hy}^{\{Q_y,K_y,V_y\}} \in \mathbb{R}^{\frac{d_{model}}{2}\times d_c}$ to map $\tilde{\mathbf{e}}_x$ and $\tilde{\mathbf{e}}_y$ to a $\mathbb{R}^{d_c}$ space. Note that $hx, hy = 0, 1, \ldots, H$ and the following equation holds: $2d_c \cdot H^2 = d_{model}$.

Here, for each point, we define two sets of the mapped embeddings in the $\mathbb{R}^{d_c}$ space: $\mathcal{X}$ and $\mathcal{Y}$, with $|\mathcal{X}| = |\mathcal{Y}| = H$. Cartesian product is conducted between them as a concatenation strategy to generate new embedding sets:

$$\mathcal{X} \times \mathcal{Y} = \{\mathbf{e}_c = [\mathbf{x}; \mathbf{y}] \mid \mathbf{x} \in \mathcal{X} \text{ and } \mathbf{y} \in \mathcal{Y}\} \tag{12}$$

where $[;]$ is the concatenation operation to enrich the feature interaction while ensuring the attention outputs are still in the same dimensions as the inputs. $\mathbf{e}_c \in \mathbb{R}^{2d_c}$ and $|\mathcal{X} \times \mathcal{Y}| = H^2$ is the total number of heads.

After the attention operation, we rearrange the orders to reform the embeddings of covariates and the target variable (indicated in the right of Figure 2). Finally, two linear projections $\mathbf{W}^{\{O_x,O_y\}} \in \mathbb{R}^{\frac{d_{model}}{2}\times\frac{d_{model}}{2}}$ are used to map the rearranged features into a final output.

3.5 Gaussian Attention Bias

Attention mechanisms intrinsically model internal interactions of a sequence, i.e., they selectively aggregate information that is beneficial to the task. However, the vanilla attention does not explicitly consider the effect of spatial proximity.

To tackle this problem, we propose to bias the interactions directly with a geographical prior. The Gaussian kernel, widely used in GWR models, is a common choice in various applications (as given in Equation 4) (Jia et al. 2024). We, therefore, adjust the attention coefficients with a Gaussian bias, derived from the spatial proximity of point pairs (Guo, Zhang, and Liu 2019; Kim et al. 2023). Given the embeddings of these two points, $\mathbf{e}_i$ and $\mathbf{e}_j$:

$$\alpha_{i,j} = \frac{\exp(\mathbf{e}_i \cdot \mathbf{e}_j)}{\sum_l \exp(\mathbf{e}_i \cdot \mathbf{e}_l)} \tag{13}$$

$$\begin{aligned}\tilde{\mathbf{e}}_i &= \sum_j \frac{\exp(-d_{i,j}^2)\cdot\alpha_{i,j}}{\sum_k \exp(-d_{i,k}^2)\cdot\alpha_{i,k}} \cdot \mathbf{e}_j \\ &= \sum_j \frac{\exp(\mathbf{e}_i \cdot \mathbf{e}_j - d_{i,j}^2)}{\sum_k \exp(\mathbf{e}_i \cdot \mathbf{e}_k - d_{i,k}^2)} \cdot \mathbf{e}_j \\ &= \sum_j \text{softmax}(\mathbf{e}_i \cdot \mathbf{e}_j - d_{i,j}^2) \cdot \mathbf{e}_j\end{aligned} \tag{14}$$

where $\alpha_{i,j}$ denotes the attention coefficient between $\mathbf{e}_i$ and $\mathbf{e}_j$. We add a learnable attention bias factor $\lambda \in \mathbb{R}$ to control the magnitude of the bias so that:

$$\tilde{\mathbf{e}}_i = \sum_j \text{softmax}(\mathbf{e}_i \cdot \mathbf{e}_j - \lambda d_{i,j}^2) \cdot \mathbf{e}_j \tag{15}$$

The effectiveness of the proposed Gaussian bias is examined in the ablation study.

3.6 GeoAggregator Model

Overall, GeoAggregator has an *encoder-processor-decoder* architecture that achieves end-to-end prediction.

Encoder. The encoder contains one MCPA operation, compressing the information from the contextual points into ℓ_{hidden} learnable states (denoted as inducing points). We inject the spatial location of the target point into the inducing points. This enables each inducing point to learn different interaction patterns between contextual and target locations.

Processor. Similar to (Lee and Oh 2024), we introduce L processor modules so that the computational complexity of all-point attention ($O(\ell_{in}^2)$) is eased to $O(\ell_{in} \cdot \ell_{hidden})$, where $\ell_{hidden} \ll \ell_{in}$. Therefore, the computational complexity grows linearly with the input sequence length, as will be shown in later analysis. The processor module learns the interactions between ℓ_{hidden} inducing points.

Decoder and Output Head. The decoder is a cross-attention module that queries inducing points using the target point. We append a dense network with Tanhshrink activation as the output head. We concatenate the decoded embeddings and the original features $\mathbf{x}_i^t$ and $\mathbf{l}_i^t$ to ensure that a linear solution is always a subclass of our model.

4 Experiments

Datasets. Experiments are first conducted on eight synthetic datasets to test the performance of models in handling different spatial processes. We generate the datasets using four different spatial processes: **Linear Model** (Lin) (Li 2022), **Spatial Lagged Model** (SL), **Spatial Lagged X Model** (SLX) (Anselin 2009), and **Spatial Durbin Model** (Durbin) (Mur and Angulo 2005). Additionally, two types of covariates ($\mathbf{x}$) are generated: 1) sampled from a uniform distribution ($x \sim U(-1, 1)$) whose values are independent of spatial location (denoted as {Lin, SL, SLX, Durbin}-r), and 2) sampled from a real-world Digital Elevation Map (DEM) which exhibits spatial autocorrelation (denoted as {Lin, SL, SLX, Durbin}-d).

To further illustrate the performances, we use three real-world datasets of different sizes: (1) **PM25**: from (Dai et al. 2022), which includes 1,457 PM2.5 concentration measurements across mainland China, coupled with related environmental factors. It presents a long-range spatial regression problem due to its sparse and uneven data distribution; (2) **Housing**: as per (Li 2024), this dataset contains housing prices in King County, WA, USA. It represents a small-scale, densely distributed spatial dataset with notable spatial effects; (3) **Poverty**: Sourced from (Kolak et al. 2020), this dataset includes 14 socioeconomic variables that estimate poverty levels across the continental US. It features a mid-range spatial regression challenge with densely distributed neighborhood-level data.

We set the splitting ratio of training-validation-testing to be 7:1:2 for all datasets except for the PM25 dataset, whose splitting ratio is 56:14:30 (Dai et al. 2022).

Baseline Models. We select five models as baselines. One classic spatial statistical model, Geographically Weighted

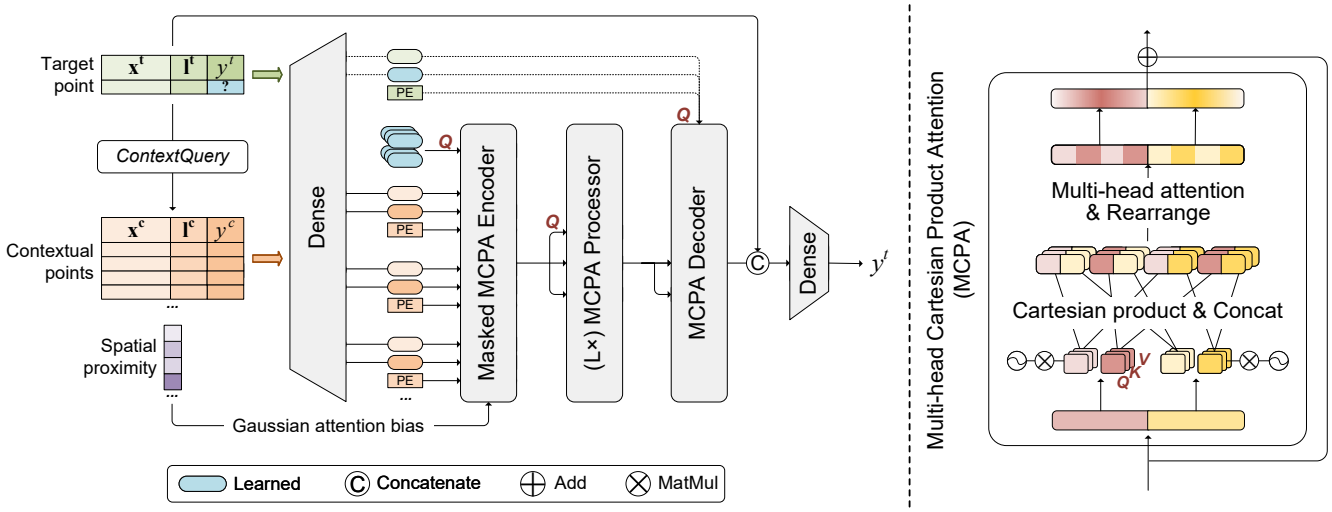

Figure 2: Illustration of the GeoAggregator model and the Multi-head Cartesian Product Attention (MCPA) mechanism.

Dataset	#data point	#covariate	range of spatial effect
Synthetic datasets	2500	2	-
PM25	1457	7	mid-range
Housing	16580	8	short-range
Poverty	71900	14	long-range

Table 1: Basic profiles of the datasets used in this paper.

Regression (**GWR**) (Brunsdon, Fotheringham, and Charlton 1998). One tree-based ensemble model, **XGBoost** (Chen and Guestrin 2016). We also select three representative deep networks, Spatial Regressive Graph Convolutional Network (**SRGCNN**) (Zhu et al. 2022), Geographically CNN Weighted Regression (**GCNNWR**) (Dai et al. 2022), and Geographical Spatial Heterogeneous Ensemble Learning model (**GSH-EL**) (Cheng et al. 2024). Additionally, we include 3 GeoAggregator models with the MCPA operation replaced by vanilla attention (denoted as **Vanilla** in the following experiments) to demonstrate the effectiveness of the MCPA mechanism.

Evaluation Metrics. We use mean absolute error (MAE) as an objective function and evaluation metric and report the R-square (R^2) as another evaluation metric. We report the mean value from 3 repeated training and testing runs.

Implementation and Training Details. We set the latent dimension d_{model} as 32, total number of heads $H^2 = 4$. We implement three versions of the GeoAggregator, with the number of processor modules $L = 0, 1, 2$ named GeoAggregator-mini (GA-mini), GA-small, and GA-large, respectively. We set l_{hidden} to be 0, 4, 8; the parameter l_{max} to be 81, 144, and 256, respectively. Corresponding searching radius in the $ContextQuery$ operation is estimated on training datasets.

We use the Adam optimizer with a cyclical learning rate scheduler (max learning rate is 5×10^{-3}) (Kingma 2014; Popel and Bojar 2018). We conduct synthetic data experiments on a laptop with 32GB of RAM and real-world data experiments on a Google Colab virtual machine equipped with an NVIDIA P100 GPU with 16GB of GPU memory.

4.1 Performance on Synthetic Datasets

The regression performances on eight synthetic datasets are listed in Table 2. 3 GeoAggregator models exhibit the strongest or most competitive performance. GA-mini performs the best in all six variants of transformer models, achieving four best results. This showcases its strong ability to capture stationary/non-stationary spatial patterns and to learn global-scale SH. For the convolutional solutions, the GCNNWR model achieves four best results, the same as our GA-mini. The competitive performances indicate that GCNNWR learns to map the spatial proximity grid directly to regression coefficients. SRGCNN and GSH-EL perform less satisfactorily on datasets with randomly sampled covariates. This means they overemphasize the global point interactions, which introduce noise instead of useful information. Two non-deep learning models, GWR and XGBoost, exhibit strong robustness across all datasets, although they are not among the top solutions.

Generally, more detailed hyperparameter tuning of the GeoAggregator, including varying l_{hidden}, L, and l_{max}, could potentially lead to better results. Besides, larger GeoAggregator models tend to perform worse than smaller ones. This could be due to increased overfitting and difficulty in model training.

4.2 Performance on Real-World Datasets

We further compare the models on 3 representative real-world datasets, the results are summarized in Table 3. On the smallest dataset (PM25), our GeoAggregator models show less satisfactory results than the best-performing model. This is because long-range dependence contributes less to the target point, considering the well-known complex spa-

Model	Lin-r		SL-r		SLX-r		Durbin-r		Lin-d		SL-d		SLX-d		Durbin-d		#best
	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	MAE	R^2	
GWR	0.873	0.906	1.445	0.860	0.864	0.697	1.338	0.705	0.830	0.802	**0.860**	**0.991**	0.807	0.607	0.869	0.980	2
XGBoost	0.957	0.895	1.719	0.820	0.891	0.746	1.642	0.649	0.840	0.827	0.880	0.991	0.822	0.720	0.912	0.979	0
SRGCNN	2.805	-0.271	2.904	0.374	1.78	-0.184	1.754	0.611	1.013	0.691	0.982	0.984	1.066	0.499	0.900	0.977	0
GCNNWR	0.897	0.907	**1.133**	**0.929**	0.898	0.768	1.013	0.858	0.818	0.827	0.868	0.990	0.864	0.713	**0.857**	**0.983**	4
GSH-EL	1.382	0.720	2.884	0.157	0.961	0.647	2.178	-0.178	1.115	0.547	2.938	0.828	0.917	0.558	1.791	0.921	0
Vanilla-mini	0.887	0.909	1.213	0.919	**0.839**	0.753	0.929	0.884	0.913	0.800	1.124	0.984	0.836	0.709	0.995	0.978	1
Vanilla-small	0.892	0.908	2.351	0.694	0.844	0.751	**0.897**	**0.892**	0.878	0.817	0.980	0.988	0.833	0.710	0.924	0.981	2
Vanilla-large	0.880	0.912	2.555	0.642	0.848	0.749	1.554	0.666	0.883	0.816	1.223	0.982	0.830	0.713	1.143	0.970	0
GA-mini	**0.870**	**0.920**	1.269	0.911	0.881	0.751	0.946	0.876	**0.818**	0.810	1.039	0.985	**0.804**	0.710	1.054	0.971	4
GA-small	0.905	0.910	1.530	0.840	0.872	**0.771**	0.920	0.880	0.851	**0.827**	1.000	0.988	0.819	0.736	1.280	0.964	2
GA-large	0.905	0.911	2.383	0.646	0.870	0.771	1.847	0.547	0.867	0.823	1.169	0.984	0.820	**0.737**	1.165	0.970	1

Table 2: Performance on our eight synthetic datasets. GeoAggregators (GA) achieved seven best results in total.

Model	PM25		Housing		Poverty		# best
	MAE	R^2	MAE	R^2	MAE	R^2	
GWR	3.933	0.801	0.733	0.803	4.214	0.739	0
XGBoost	4.017	0.813	0.645	0.888	3.622	**0.845**	1
SRGCNN	4.181	0.771	1.370	0.429	-	-	0
GCNNWR	**3.797**	0.832	0.704	0.895	-	-	1
GSH-EL	3.863	**0.842**	0.718	0.860	-	-	1
Vanilla-mini	4.075	0.832	**0.623**	0.906	3.563	0.839	1
Vanilla-small	4.649	0.776	0.647	0.904	3.698	0.832	0
Vanilla-large	4.494	0.787	0.649	0.900	3.637	0.835	0
GA-mini	4.480	0.821	0.624	**0.911**	3.547	0.842	1
GA-small	4.928	0.772	0.650	0.904	**3.537**	0.844	1
GA-large	4.721	0.778	0.641	0.896	3.565	0.842	0

Table 3: Performance on three real-world datasets. GeoAggregators (GA) achieve 2 best results in total.

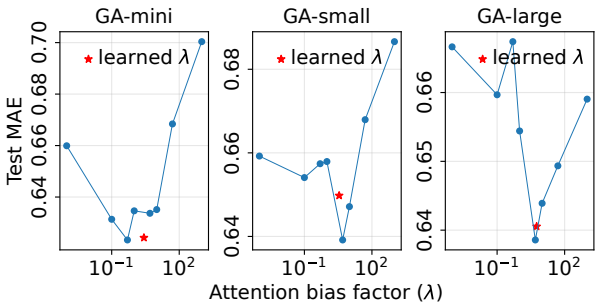

Figure 3: Effect of the attention bias factor λ of 3 variants of GeoAggregator, on the Housing dataset.

tial heterogeneity pattern in mainland China (Wang, Zhang, and Fu 2016). We also observe that different data splits lead to significant variations in the results.

On the Housing price dataset, two GA-mini models achieve SOTA, indicating their efficiency in modeling densely distributed data points with governing SA and the ability to capture SH and global spatial structure through local modeling. XGBoost demonstrates its flexibility as a strong baseline in handling tabular data.

On the Poverty dataset, GA-small achieves the lowest MAE and a second-best R^2 because it learns complex neighboring interactions governed by SA through inducing points. SRGCNN, GCNNWR and GSH-EL fail to complete training due to high memory requirements or prohibitively long runtime. XGBoost still performs well due to its ability to learn complex patterns and high flexibility.

4.3 Computational Efficiency

We compare the computational efficiency through the number of learnable parameters (#Param) and number of floating operations (#FLOPs) in one inference of each model (Table 4). As the dataset size increases, #Param and #FLOPs for GeoAggregators remain relatively stable, whereas other deep learning models exhibit a significant increase in both #Param and #FLOPs. The GeoAggregator models using the MCPA mechanism also demonstrate notably lower #Param and #FLOPs compared to those with vanilla attention. Finally, with an increasing number of attention layers, the #Param and #FLOPs values of GeoAggregators are effectively managed due to the introduction of inducing points in processor modules. The above characteristics of the GeoAggregator model make it well suited for efficient modeling across datasets of varying sizes.

4.4 Ablation Study

Effect of the Attention Bias Factor. We conduct a series of experiments using three variants of the GeoAggregator on the Housing dataset with varying attention bias factor λ, as shown in Figure 3. Specifically, we fix λ as $\{10^{-3}, 0.1, 0.5, 1, 5, 10, 50, 10^3\}$ during the training and testing stage. All 3 GeoAggregators perform not well when $\lambda = 10^{-3}$ or $\lambda = 10^3$, i.e., when little or too much geographical prior is added to guide the local attention. In contrast, 3 GeoAggregators perform best when using a balanced λ, demonstrating the validity of introducing the Gaussian bias. We also marked the results of 3 variants with learnable λ with red stars in Figure 3. The learned λs are consistent with our ablation experiments, further indicating the effectiveness of our design.

Effect of the Sequence Length. GeoAggregator models selectively capture neighboring interactions through a local

Model	Synthetic		PM25		Housing		Poverty	
	#Params	#FLOPs	#Params	#FLOPs	#Params	#FLOPs	#Params	#FLOPs
GWR	-	-	-	-	-	-	-	-
XGBoost	-	-	-	-	-	-	-	-
SRGCNN			224.4K	670.9M	2820K	8.5M	-	-
GCNNWR	1930K	474.6M	1210K	231.2M	8910K	3281.8M	-	-
GSH-EL	460K	29.3M	220K	14.0M	2980K	190.1M	-	-
Vanilla-mini	7.6K	1.7M	7.9K	3.5M	7.9K	3.6M	8.2K	3.7M
Vanilla-small	12.9K	3.3M	13.2K	6.7M	13.2K	6.7M	13.5K	7.0M
Vanilla-large	18.2K	10.5M	18.4K	21.3M	18.5K	21.4M	18.8K	21.8M
GA-mini	**4.3K**	**0.7M**	**4.6K**	**1.6M**	**4.6K**	**1.6M**	**4.9K**	**1.7M**
GA-small	6.3K	1.4M	6.5K	2.9M	6.6K	3.0M	6.8K	3.2M
GA-large	8.2K	2.7M	8.5K	5.7M	8.5K	5.8M	8.8K	6.2M

Table 4: The computational complexity of models in comparison, measured by #Params and #FLOPs.

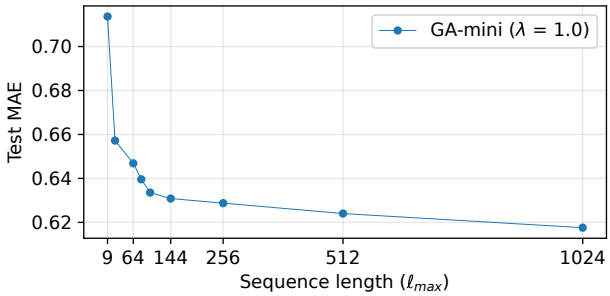

Figure 4: Effect of the input sequence length ℓ_{max}. We compare results of the GA-mini model with $\lambda = 1.0$, on the Housing dataset.

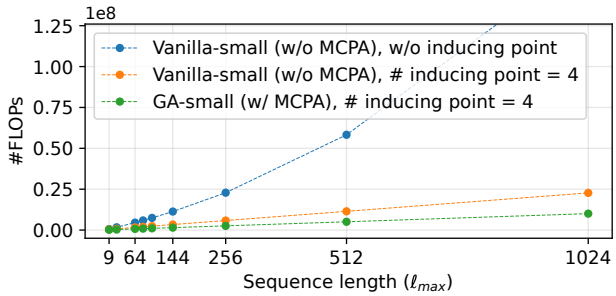

Figure 5: Computational cost of one inference of different attention mechanisms, on the Housing dataset.

attention operation. We train a GeoAggregator-mini model (without the processor module) with varying input sequence lengths ℓ_{max} to assess the impact of sequence length on the performance. As shown in Figure 4, the performance (in terms of test MAE) continues to improve with increasing sequence length until it reaches 0.618 when $\ell_{max} = 1024$. This indicates that the GeoAggregator learns to efficiently aggregate useful information from local input sequences. Unlike GWR or graph-based models, GeoAggregator eases the over-smoothing problem when the bandwidth or neighborhood size is overly large. That is, it's robust to less relevant data points, even within a large neighborhood.

Based on the previous discussion, we argue that for datasets with unclear SA and SH patterns, one can gradually increase ℓ_{max} until one reaches a satisfactory result that balances computational efficiency and prediction accuracy.

Computational Complexity of Different Attention Mechanisms. To demonstrate the computational efficiency of the proposed Multi-head Cartesian Product Attention (MCPA), we compare the number of floating-point operations (#FLOPs) in one inference of the vanilla full attention, vanilla inducing point attention, and our MCPA on the Housing dataset. All 3 models contain 3 attention layers, with the inducing point models incorporating 4 inducing points in both the first and second attention layers.

We report #FLOPs with increasing ℓ_{max} in Figure 5. It is depicted that by introducing inducing points, the computational complexity of the transformer architecture scales linearly with growing ℓ_{max}. Moreover, replacing the vanilla attention mechanism with our MCPA further reduces the computational complexity, making it suitable for handling large datasets that involve dense point distributions and complex geospatial effects.

5 Conclusion and Future Work

In this work, we address the issues of scalability and flexibility in current deep networks for geospatial tabular data modeling, by introducing a novel lightweight transformer-based model, GeoAggregator. GeoAggregator explicitly accounts for spatial autocorrelation and spatial heterogeneity effects through a novel Gaussian-biased Cartesian product attention mechanism and a global positional awareness. Our GeoAggregator model shows superior performance and computational efficiency on synthetic and real-world datasets compared to several baseline models, offering a promising solution for geospatial tabular data tasks.

For future work, tailoring the bias for each spatial feature individually could potentially improve performance. In addition, incorporating new input heads to handle categorical variables would also be a beneficial improvement.

References

Anselin, L. 2009. Spatial regression. *The SAGE handbook of spatial analysis*, 1: 255–276.

Anselin, L. 2010. Thirty years of spatial econometrics. *Papers in Regional Science*, 89(1): 3–25.

Bjorck, N.; Gomes, C. P.; Selman, B.; and Weinberger, K. Q. 2018. Understanding batch normalization. *Advances in neural information processing systems*, 31.

Brunsdon, C.; Fotheringham, S.; and Charlton, M. 1998. Geographically weighted regression. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 47(3): 431–443.

Chen, T.; and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, 785–794.

Cheng, S.; Wang, L.; Wang, P.; and Lu, F. 2024. An ensemble spatial prediction method considering geospatial heterogeneity. *International Journal of Geographical Information Science*, 1–25.

Dai, Z.; Sensen, W.; Wang, Y.; Zhou, H.; Zhang, F.; Huang, H.; and Du, Z. 2022. Geographically convolutional neural network weighted regression: a method for modeling spatially non-stationary relationships based on a global spatial proximity grid. *International Journal of Geographical Information Science*, 36(11): 2248–2269.

Dao, T.; Fu, D.; Ermon, S.; Rudra, A.; and Ré, C. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Advances in Neural Information Processing Systems*, 35: 16344–16359.

Fotheringham, A. S.; Brunsdon, C.; and Charlton, M. 2003. *Geographically Weighted Regression: The Analysis of Spatially Varying Relationships*. John Wiley & Sons.

Fotheringham, A. S.; and Li, Z. 2023. Measuring the unmeasurable: models of geographical context. *Annals of the American Association of Geographers*, 113(10): 2269–2286.

Guo, M.; Zhang, Y.; and Liu, T. 2019. Gaussian Transformer: A Lightweight Approach for Natural Language Inference. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01): 6489–6496.

He, W.; Jiang, Z.; Xiao, T.; Xu, Z.; Chen, S.; Fick, R.; Medina, M.; and Angelini, C. 2023. A hierarchical spatial transformer for massive point samples in continuous space. *Advances in neural information processing systems*, 36: 33365–33378.

Jaegle, A.; Borgeaud, S.; Alayrac, J.-B.; Doersch, C.; Ionescu, C.; Ding, D.; Koppula, S.; Zoran, D.; Brock, A.; Shelhamer, E.; et al. 2021. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*.

Jia, Y.; Wu, Z.; Yi, S.; and Sun, Y. 2024. GeoTransformer: Enhancing Urban Forecasting with Geospatial Attention Mechanisms. *arXiv preprint arXiv:2408.08852*.

Jiang, Z. 2019. A Survey on Spatial Prediction Methods. *IEEE Transactions on Knowledge and Data Engineering*, 31(9): 1645–1664.

Karimi, H. A.; and Karimi, B. 2017. *Geospatial data science techniques and applications*. CRC Press.

Kim, B. J.; Choi, H.; Jang, H.; and Kim, S. W. 2023. Understanding gaussian attention bias of vision transformers using effective receptive fields. *arXiv preprint arXiv:2305.04722*.

Kingma, D. P. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Kolak, M.; Bhatt, J.; Park, Y. H.; Padrón, N. A.; and Molefe, A. 2020. Quantification of Neighborhood-Level Social Determinants of Health in the Continental United States. *JAMA Network Open*, 3(1): e1919928–e1919928.

Lee, S.; and Oh, T. 2024. Inducing Point Operator Transformer: A Flexible and Scalable Architecture for Solving PDEs. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, 153–161.

Li, J.; Shen, Y.; Chen, L.; and Ng, C. W. W. 2023. Ssin: Self-supervised learning for rainfall spatial interpolation. *Proceedings of the ACM on Management of Data*, 1(2): 1–21.

Li, Z. 2022. Extracting spatial effects from machine learning model using local interpretation method: An example of SHAP and XGBoost. *Computers, Environment and Urban Systems*, 96: 101845.

Li, Z. 2024. GeoShapley: A Game Theory Approach to Measuring Spatial Effects in Machine Learning Models. *Annals of the American Association of Geographers*, 1–21.

Luo, Y.; Liu, Q.; and Liu, Z. 2021. Stan: Spatio-temporal attention network for next location recommendation. In *Proceedings of the web conference 2021*, 2177–2185.

Mai, G.; Janowicz, K.; Yan, B.; Zhu, R.; Cai, L.; and Lao, N. 2020. Multi-scale representation learning for spatial feature distributions using grid cells. *arXiv preprint arXiv:2003.00824*.

Mur, J.; and Angulo, A. 2005. A closer look at the Spatial Durbin Model. *In: Presented at the European Regional Science Association 45th Congress, European Regional Science Association, Amsterdam*.

Popel, M.; and Bojar, O. 2018. Training tips for the transformer model. *arXiv preprint arXiv:1804.00247*.

Stewart, A. J.; Robinson, C.; Corley, I. A.; Ortiz, A.; Ferres, J. M. L.; and Banerjee, A. 2022. Torchgeo: deep learning with geospatial data. In *Proceedings of the 30th international conference on advances in geographic information systems*, 1–12.

Su, J.; Ahmed, M.; Lu, Y.; Pan, S.; Bo, W.; and Liu, Y. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568: 127063.

Unlu, E. 2024. Geotokens and Geotransformers. *arXiv preprint arXiv:2403.15940*.

Vaswani, A. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762*.

Vivanco Cepeda, V.; Nayak, G. K.; and Shah, M. 2024. Geoclip: Clip-inspired alignment between locations and images for effective worldwide geo-localization. *Advances in Neural Information Processing Systems*, 36.

Wang, J.-F.; Zhang, T.-L.; and Fu, B.-J. 2016. A measure of spatial stratified heterogeneity. *Ecological indicators*, 67: 250–256.

Wu, Y.; Tang, Y.; Yang, X.; Zhang, W.; and Zhang, G. 2021. Graph Convolutional Regression Networks for Quantitative Precipitation Estimation. *IEEE Geoscience and Remote Sensing Letters*, 18(7): 1124–1128.

Xu, P.; Zhu, X.; and Clifton, D. A. 2023. Multimodal learning with transformers: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 45(10): 12113–12132.

Zhao, H.; Jiang, L.; Jia, J.; Torr, P. H.; and Koltun, V. 2021. Point transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, 16259–16268.

Zhu, D.; Liu, Y.; Yao, X.; and Fischer, M. M. 2022. Spatial regression graph convolutional neural networks: A deep learning paradigm for spatial multivariate distributions. *GeoInformatica*, 26(4): 645–676.

Zhu, D.; Zhang, F.; Wang, S.; Wang, Y.; Cheng, X.; Huang, Z.; and Liu, Y. 2020. Understanding place characteristics in geographic contexts through graph convolutional neural networks. *Annals of the American Association of Geographers*, 110(2): 408–420.