Check for updates

ACM DIGITAL LIBRARY | Association for Computing Machinery | acm open

Latest updates: https://dl.acm.org/doi/10.1145/3746252.3761512

RESEARCH-ARTICLE

GeoIndia V2: A Unified Graph and Language Model for Context-Aware Geocoding

ARPIT TIWARI

BHAVUK SINGHAL

ANSHU ADITYA

SHUBHAM JAIN

DEBASHIS MUKHERJEE

DEBDOOT MUKHERJEE

PDF Download
3746252.3761512.pdf
11 March 2026
Total Citations: 0
Total Downloads: 184

Published: 10 November 2025

Citation in BibTeX format

CIKM '25: The 34th ACM International Conference on Information and Knowledge Management
November 10 - 14, 2025
Seoul, Republic of Korea

Conference Sponsors:
SIGWEB
SIGIR

https://doi.org/10.1145/3746252.3761512
ISBN: 9798400720406

GeoIndia V2: A Unified Graph and Language Model for Context-Aware Geocoding

Arpit Tiwari
arpit.tiwari1@meesho.com
Meesho
Bengaluru, India

Bhavuk Singhal
bhavuk.singhal@meesho.com
Meesho
Bengaluru, India

Anshu Aditya
anshu.aditya@meesho.com
Meesho
Bengaluru, India

Shubham Jain
shubham.jain1@meesho.com
Meesho
Bengaluru, India

Debashis Mukherjee
debashis.mukherjee@meesho.com
Meesho
Bengaluru, India

Debdoot Mukherjee
debdoot.mukherjee@meesho.com
Meesho
Bengaluru, India

Abstract

Geocoding in India presents unique challenges due to the unstructured, multilingual and diverse nature of its address systems. While recent advances in geospatial AI have explored the combination of spatial and semantic cues, existing methods often fall short in effectively integrating both dimensions for robust address resolution. In this work, we propose GeoIndia-V2, an enhanced version of GeoIndia [21], that unifies geospatial and semantic modeling through a novel fusion framework. Our unified model combines the Graphormer architecture [27] and a Pre-trained Transformer based Language Model (PTLM) that is trained from scratch on proprietary Indian address data, using our proposed Key Modulated Cross-Attention (KMCA) mechanism. KMCA enables deep cross-modal interaction between geospatial topology and linguistic structure and allows the model to reason contextually across both geographic and textual dimention, effectively handling the semantic intricacies of Indian addresses—including colloquial usage, inconsistent formatting, and multilinguality. We leverage last-mile e-commerce delivery data to construct a fine-grained graph of neighbourhood connectivity, enabling Graphormer to capture rich spatial relationships. Unlike prior methods that rely on self-loops, we generate graphs dynamically at inference time to exploit Graphormer's topological strength. Additionally, we introduce a generative decoding strategy for predicting hierarchical H3 cells[1], moving beyond conventional bit-wise classification approaches. To the best of our knowledge, this is the first method to explicitly fuse graph-based geospatial learning with language-driven semantic modeling via cross-attention in the Indian geocoding context. Our approach significantly outperforms existing solutions and marks a substantial advancement toward building scalable real-world geocoding systems for complex address ecosystems like India.

[1]https://www.uber.com/en-IN/blog/h3/

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
CIKM '25, Seoul, Republic of Korea
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-2040-6/2025/11
https://doi.org/10.1145/3746252.3761512

CCS Concepts

• **Information systems → Data mining**.

Keywords

geocoding, graph, H3-cell

ACM Reference Format:
Arpit Tiwari, Bhavuk Singhal, Anshu Aditya, Shubham Jain, Debashis Mukherjee, and Debdoot Mukherjee. 2025. GeoIndia V2: A Unified Graph and Language Model for Context-Aware Geocoding. In *Proceedings of the 34th ACM International Conference on Information and Knowledge Management (CIKM '25), November 10–14, 2025, Seoul, Republic of Korea.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3746252.3761512

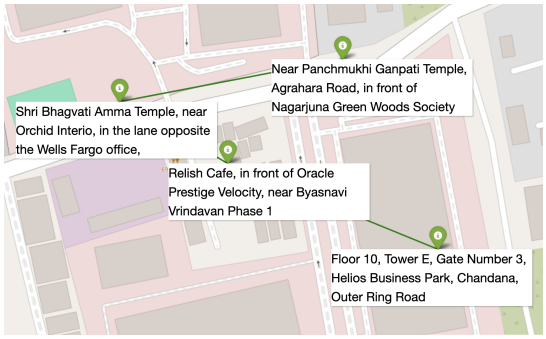
1 Introduction

The rapid growth of e-commerce has intensified the need for accurate last-mile delivery, especially in countries like India where addressing systems are often informal and unstructured. In most Tier-2 and Tier-3 cities, addresses rely heavily on local landmarks or personal references—such as *"next to Golu Mishra's house"* or *"in front of the temple"*—rather than standardized formats [18]. While easily understood by locals, such descriptions are difficult for automated systems to interpret, complicating geocoding and increasing delivery inefficiencies. These challenges contribute to an estimated annual economic loss of $10–14 billion in India, roughly 0.5% of GDP [2].

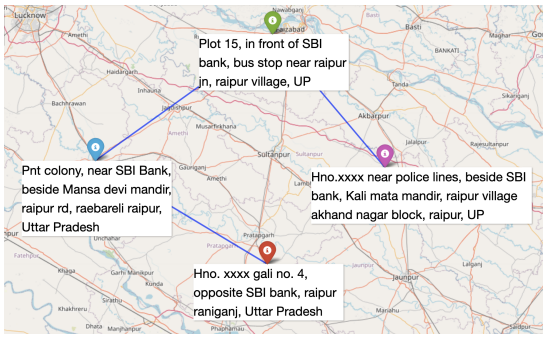
Geocoding in India demands more than just parsing the tokens in an address—it requires spatial awareness and contextual understanding. As shown in Figure 1a, multiple addresses from the same locality describe nearby locations using completely different cues—some refer temples (e.g., *"Shri Bhagvati Amma Temple"*, *"Panchmukhi Ganpati Temple"*), others refer to offices, cafes, or business parks (e.g., *"Relish Cafe"*, *"Helios Business Park"*). These variations involve different types of landmarks and phrasing, with little lexical overlap, making it difficult for models to recognize their geographic closeness. In contrast, Figure 1b shows addresses from Uttar Pradesh, a northern Indian state, which share similar terms like *"SBI Bank"* and *"Raipur"* but refer to different towns across the region. This misleading lexical similarity often confuses models lacking geographic context, highlighting the importance of spatially grounded address representations.

Several recent approaches have used pretrained language models or Seq2Seq architectures to resolve unstructured addresses [8, 19].

Our prior work, GeoIndia [21], framed address resolution as a structured generation task, using a transformer-based model to map raw text to hierarchical H3 cells. However, it relied solely on surface text and lacked spatial understanding, leading to errors in cases of misleading lexical similarity or disjoint phrasing for nearby locations—common in the Indian context. To address these limitations, we propose GeoIndia-V2, a unified model that augments language-based address understanding with spatial context derived from real-world delivery data. Field executives (FEs), who deliver packages, have practical knowledge of localities and can resolve vague or inconsistent addresses based on past experience and neighborhood layout. GeoIndia-V2 captures this intuition by building a large address graph from historical FE delivery traces, where edges represent local connectivity inferred from delivery sequences. This graph is then encoded using a Graphormer [27] to learn topological relationships between addresses.



(a)



(b)

Figure 1: Examples illustrating lexical variability in Indian addresses. (a): Diverse expressions for nearby locations within the same locality. (b): Lexically similar addresses referring to distinct locations across Uttar Pradesh.

Graphormer was originally proposed for molecular property prediction [20]. Since then, it has demonstrated strong performance in various domains, including time-series forecasting [1], decision-making [6], and hierarchy-aware text classification [9]. In geocoding, models like G2PTL [24] combine spatial and textual features but exhibit limited modality interaction and lack explicit modeling of hierarchical dependencies across the bits of geospatial grid system cells. Our framework addresses these gaps through several enhancements: (1) a multilingual pretrained language model for regional and vernacular inputs, (2) KMCA for fusing token-level semantics with Graphormer-derived spatial embeddings, (3) an autoregressive decoder for structured spatial outputs, and (4) preservation of the natural topological structure of the input address during inference, avoiding reliance on synthetic self-loops. These design choices enable robust, context-aware geolocation from noisy, unstructured addresses. Our major contributions can be summarized as follows:

- We introduced GeoIndia-V2, an extension of our previous work GeoIndia, designed to handle noisy, unstructured addresses by leveraging graph structures built from real-world delivery paths of FEs.
- Our architecture integrates Graphormer for spatial encoding and a multilingual pretrained language model for token-level semantics, fused using our proposed KMCA mechanism to combine geographic and linguistic context. A decoder then generates hierarchical spatial representations using the H3 indexing system.
- We conducted ablation studies to assess the impact of each architectural component and demonstrated the effectiveness of our approach against recent geocoding benchmarks.
- We devised a graph construction strategy during inference that avoids synthetic self-loops, thereby preserving spatial fidelity and maintaining the original topological structure. GeoIndia-V2 is now deployed on Meesho's[2] platform, supporting multiple real-world use cases.

2 Related Work

Recent works have used large language models (LLMs) and retrieval-augmented generation (RAG) to handle noisy address inputs. AddrLLM [25] normalized irregular addresses for better geocoding, while GeoAgent [5] and RACCOON [11] combined LLMs with spatial tools for reasoning and coordinate extraction. [26] showed strong generalization in GPT-3 for address parsing with retrieval. However, these models often lack spatial grounding and struggle with reasoning over address hierarchies.

Seq2Seq models have been widely used for geocoding by framing address resolution as a structured generation task, offering more control and better parsing than general-purpose LLMs. In India, GeoIndia [21] used transformer-based models with state-specific fine-tuning to predict H3 cells, while others [8, 19] tackled GPS mismatches and data sparsity. Globally, T5-based models [22], hierarchy-aware frameworks [10], and Chinese efforts like [13, 17] targeted coordinate prediction and standardization. Despite their effectiveness, Seq2Seq models treat addresses as flat sequences and struggle with spatial hierarchies, which graph-based methods better capture.

Graph-based models improved upon sequence methods by capturing spatial and hierarchical address structures. G2PTL [24] pretrained on a heterogeneous address graph but suffered from weak modality interaction and limited global consistency. Weakly supervised models [16] relied on inferred neighborhood structures but were sensitive to noise. Other works enhanced matching using

[2]https://www.meesho.com/

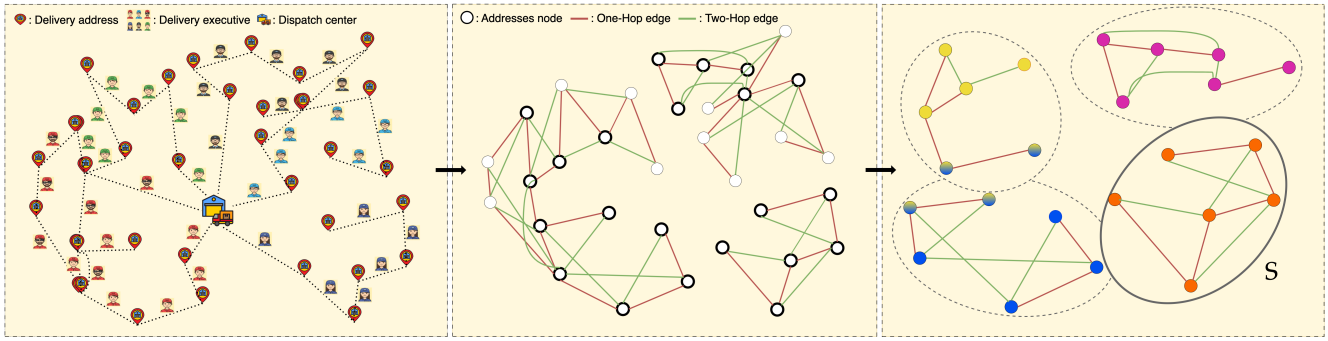


Figure 2: Schematic of graph construction from delivery scan logs.

graph attention [29], relational modeling [15], or subgraph learning [28], though many were limited to specific tasks like pairwise matching or trajectory data. IP-focused models [3, 14] demonstrated spatial reasoning via GNNs but did not support informal addresses. Hybrid models [7, 12] combined transformers and graphs, yet often featured shallow fusion and limited multimodal capability.

3 Proposed Method

Our approach draws inspiration from recent efforts such as *GeoIndia* [21], which addressed the challenges of noisy and diverse Indian addresses, and *G2PTL* [24], which highlighted the effectiveness of graph-based pretraining on real-world delivery networks. Building on these ideas, we design a system that combines token-level semantic embeddings with address-level spatial representations through graph-based modeling. In this section, we first present the construction of a graph-based input representation, describe our model architecture that integrates semantic and spatial features, and outline the training strategy used for model learning.

3.1 Graph construction

We represent the delivery address space as a undirected graph $G = (V, E)$, where each node $v \in V$ corresponds to a unique delivery address, and each undirected edge $e_{ij} \in E$ connects a source node $v_i \in V$ to a target node $v_j \in V$. These edges model spatial proximity derived from real-world delivery traces. Each node is associated with both the raw address text and its geographic coordinates.

To construct the graph topology, we utilize two years of Field Executive (FE) delivery scan logs. For each FE, deliveries are ordered chronologically by scan timestamps. A **1-hop edge** is added between two consecutive deliveries, and a **2-hop edge** connects deliveries with exactly one intermediate stop. We enforce a maximum time gap of 15 minutes between scans to ensure spatial locality. Each edge e_{ij} is annotated with a one-hot vector $F_{ij} \in \{0, 1\}^2$ indicating whether it corresponds to a 1-hop or 2-hop transition. The resulting large and sparse graph is implemented using the Deep Graph Library (DGL) [23].

Each node $v \in V$ is assigned a tokenized sequence $x_v \in \mathbb{Z}^l$, where $l = 90$, using a tokenizer trained on 67 million Indian addresses[3]. These token sequences are used both as node features and as input to the PTLM encoder during model training. To enable scalable training, we constructed subgraphs $S = (V_S, E_S) \subseteq G$ using random walks. Each walk began from a randomly chosen node and expanded by traversing connected neighbors, resulting in a subgraph composed of $n_S = |V_S|$ address nodes and $m_S = |E_S|$ edges between them. This approach preserves local graph structure while balancing computational efficiency.

[3]Tokenizer training details are provided in GeoIndia [21].

3.2 Model Architecture

Our proposed architecture consisted of five main components: a Pretrained Transformer Language Model (PTLM)-based encoder, a Graphormer for spatial encoding, a Key-Modulated Cross Attention (KMCA) fusion block, a Multi-Layer Perceptron (MLP) based projection block, and an autoregressive decoder for H3-cell prediction. An overview of the model architecture is illustrated in Figure 3.

As described earlier in section 3.1, we modeled the full address space as a undirected graph $G = (V, E)$, where each node $v \in V$ represented a unique delivery address, and each edge $e \in E$ captured spatial proximity between address pairs. Specifically, an edge $e_{ij} \in E$ connected two nodes $v_i, v_j \in V$ if they were observed within 1-hop or 2-hop delivery transitions, based on field executive traces.

We sampled individual subgraphs $S = (V_S, E_S) \subseteq G$, where V_S was a set of n_S address nodes and E_S represented edges between them.

Each address for node $v \in V_S$ was associated with a raw textual address. We tokenized it using a PTLM tokenizer, producing a fixed-length token sequence of $l = 90$ tokens:

$$\left.\begin{aligned} x_v &= \text{Tokenizer}(Address\ v) \in \mathbb{Z}^l \\ x_S &= \{x_v \mid v \in V_S\} \in \mathbb{Z}^{n_S \times l} \end{aligned}\right\} \quad \text{Tokenization} \tag{1}$$

The token sequences were passed through a shared PTLM encoder, yielding contextual token embeddings with hidden dimension $d = 768$:

$$\left.\begin{aligned} X_v &= \text{PTLMEncoder}(x_v) \in \mathbb{R}^{l \times d} \\ X_S &= \{X_v \mid v \in V_S\} \in \mathbb{R}^{n_S \times l \times d} \end{aligned}\right\} \quad \text{Semantic encoding} \tag{2}$$

We then constructed structural information for each subgraph. The shortest-path distance matrix $A_S \in \mathbb{Z}^{n_S \times n_S}$ recorded distances between node pairs. The path-type tensor $Q_S \in \mathbb{Z}^{n_S \times n_S \times (n_S - 1) \times 2}$ encoded sequences of 1-hop or 2-hop edge types. The degree vector $D_S \in \mathbb{Z}^{n_S \times 2}$ stored in-degrees and out-degrees. The subgraph edge-type matrix can be defined as:

$$F_S = \{F_{ij} \mid e_{ij} \in E_S,\ v_i \in V_S,\ v_j \in V_S\} \in \mathbb{Z}^{m_S \times 2} \tag{3}$$

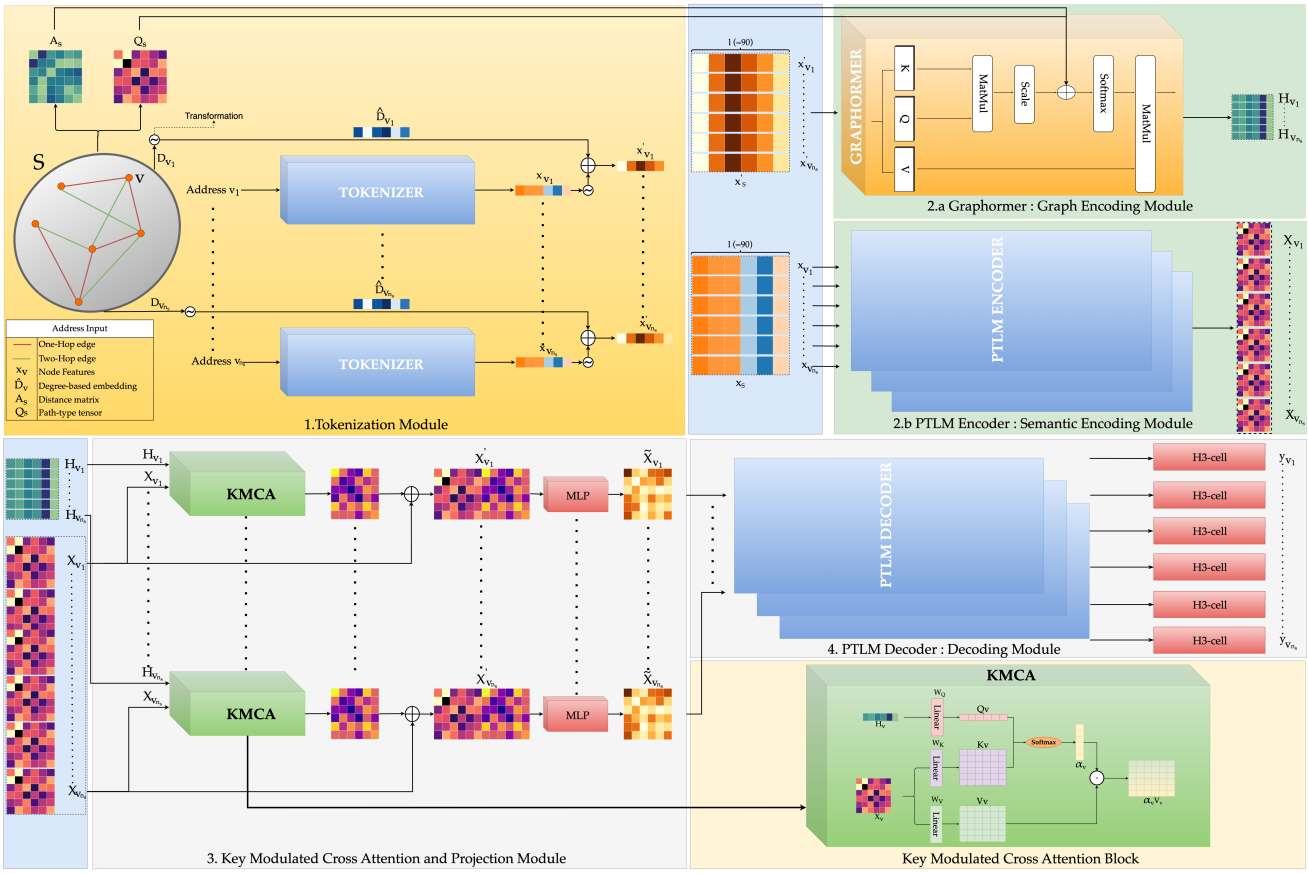


Figure 3: Overview of the GeoIndia-V2 architecture. The model first tokenizes address, which are then encoded separately by Graphormer and a domain-specific PTLM. The outputs are fused using our Key Modulated Cross-Attention (KMCA) module and passed to a PTLM decoder for hierarchical H3-cell generation.

To prepare spatial inputs, we learned semantic bias embeddings $\hat{x}_S \in \mathbb{R}^{n_S \times l}$ and degree-based embeddings $\hat{D}_S \in \mathbb{R}^{n_S \times l}$. These were combined and passed to the Graphormer:

$$\left.\begin{aligned} x'_S &= \hat{x}_S + \hat{D}_S \in \mathbb{R}^{n_S \times l} \\ H_v &= \text{Graphormer}(x'_S, F_S, A_S, Q_S) \in \mathbb{R}^d \\ H_S &= \{H_v \mid v \in V_S\} \in \mathbb{R}^{n_S \times d} \end{aligned}\right\} \quad \text{Spatial encoding} \qquad (4)$$

To integrate spatial and semantic representations, we employed **KMCA** mechanism. In this formulation, token-level embeddings act as keys and values, while the graph embedding serves as the query. The attention weights α_v are computed for each token based on its interaction with the global graph context. Unlike standard cross-attention, where the query is enriched, KMCA **modulates and enriches the keys themselves**. This key-centric modulation enables the model to adaptively emphasize tokens relevant to graph context.

$$\left.\begin{aligned} &K_v = X_v W_K, \quad V_v = X_v W_V, \quad Q_v = H_v W_Q \\ &\alpha_v = \text{softmax}\left(\frac{Q_v K_v^\top}{\sqrt{l}}\right) \\ &X'_v = [X_v \,\|\, \alpha_v V_v] \in \mathbb{R}^{l \times 2d} \end{aligned}\right\} \quad \text{KMCA fusion} \qquad (5)$$

The fused representation was projected and decoded to produce H3-cell predictions. The output sequence length was set to $h = 15$, corresponding to the resolution of the hierarchical H3 representation:

$$\left.\begin{aligned} \tilde{X}_v &= \text{MLP}(X'_v) \in \mathbb{R}^{l \times d} \\ y_v &= \text{PTLMDecoder}(\tilde{X}_v) \in \mathbb{Z}^h \\ y_S &= \{y_v \mid v \in V_S\} \in \mathbb{Z}^{n_S \times h} \end{aligned}\right\} \quad \text{Projection and decoding} \qquad (6)$$

This architecture enabled joint spatial-semantic reasoning by fusing structural topology with language-aware address features.

3.3 Training Strategy

The model was trained to generate an H3-cell sequence of length $h = 15$ for each address node within a sampled subgraph. Each subgraph contained n_S nodes, and the architecture was designed to process all nodes in parallel using shared encoder and decoder components. At each training step, a batch of B sampled subgraphs was provided as input, resulting in $B \times n_S$ nodes being processed concurrently. Each node independently underwent tokenization, semantic encoding, KMCA fusion, and autoregressive decoding, while graph-based spatial encoding is calculated by passing the entire sub-graph. For each node v_i, the model predicted a sequence

of H3-cell tokens, and a cross-entropy loss $\mathcal{L}_i$ was computed over the sequence. The total batch loss was computed as the average of individual node-level losses:

$$\mathcal{L} = \frac{1}{B \times n_S} \sum_{i=1}^{B \times n_S} \mathcal{L}_i \tag{7}$$

This formulation ensured that each node contributed equally to the parameter updates. For example, with $B = 8$ and $n_S = 6$, a total of 48 nodes were processed in parallel, and the loss was averaged across them. The model was optimized using the *AdamW* optimizer with a learning rate of 5×10^{-5}. Weight decay was applied to reduce overfitting. To stabilize training, gradient clipping with a maximum norm of 1.0 was employed. Dropout was used after the MLP projection block and within the decoder layers as a regularization technique. During training, teacher forcing was used, where the ground truth H3 token from the previous timestep was supplied to the decoder to predict the next token in the sequence.

4 Experiments

In this section, we verify the effectiveness of GeoIndia-V2 by conducting extensive experiments.

4.1 Experimental Setup

State	#Addresses		Train		#Test
	WSD	HCD	#sub-graphs	#edges	
UP	6774636	15795510	70M	276864988	1763104
HR	1344594	2374582	15M	29634783	174155
KR	3225645	7020804	25M	95342518	515088

Table 1: Dataset statistics across states. WSD: Weakly Supervised Dataset, HCD: High-Confidence Dataset.

4.1.1 Dataset. To support robust learning under variable label quality, we partition the training data into two sets based on annotation reliability. The Weakly Supervised Dataset (WSD) comprises large-scale address GPS pairs from historical delivery logs with minimal human oversight, offering broad linguistic coverage despite noisy labels. It is used for initial training to expose the model to real world address variability. The High-Confidence Dataset (HCD) is curated using stricter criteria, such as consistent multi-day deliveries, spatial proximity between initial and final scans, along with manual spot checks and subsequently used for fine-tuning. Both datasets follow the same normalization process as GeoIndia [21], with GPS coordinates converted to H3-cell sequences at resolution 9[4]. Data statistics are summarized in Table 1.

4.1.2 Baselines. We evaluate our proposed approach against the following state-of-the-art (SOTA) baselines:

- **Google Maps Geocoding API(GMaps)** [4]: A commercial geocoding system with global coverage and widespread usage.
- **GeoIndia** [21]: A sequence-to-sequence transformer model trained to predict hierarchical H3-cell sequences from address text.
- **G2PTL** [24]: A G2PTL like architecture trained on our dataset without any additional pretraining.

[4]https://h3geo.org/docs/core-library/restable

4.1.3 Implementation Details and Evaluation Protocol. For fair comparison, all models—including ours and the baselines—were trained using a two-stage process. First, the PTLM was pretrained on the WSD dataset to learn from noisy, informal address structures. Then, the full model was fine-tuned on the cleaner HCD dataset for better generalization. At inference time, we compared two subgraph construction strategies: (1) a self-loop setup where the same address is repeated across six isolated nodes, and (2) our proposed method, that generates dynamic graph during inference using historical successful deliveries (refer Section 5). All models were implemented in PyTorch and trained on NVIDIA A100 GPUs. We evaluated performance using standard geocoding metrics: Accuracy within 1 km (Acc@1km), Mean distance error (Mean), and the 90th (P90) and 99th (P99) percentile distance errors. These metrics were computed using the Haversine formula[5], with results reported separately for each state.

4.2 Overall Performance

We evaluate the geocoding performance of all models across three linguistically and structurally diverse Indian states: Uttar Pradesh (UP), Haryana (HR), and Karnataka (KR). These regions present a broad spectrum of address patterns, from highly informal rural areas to densely structured urban layouts. Table 2 summarizes the results for all baseline models and our proposed approach across these states. Our model consistently outperforms all baselines on Acc@1km, Mean, P90 and P99, demonstrating strong generalization across varied geographic and linguistic contexts. In addition to standard baselines, we also report ablated variants of each model trained without the initial weak supervision phase using WSD (denoted as *-w/o WSD*). This variant shows a noticeable drop in performance, highlighting the value of pretraining on noisy but diverse address data to build stronger initial representations.

4.3 Ablation Study

We conducted a series of experiments to analyze the contribution of key architectural components, using address data from two high-density Indian states—Uttar Pradesh (UP) and Haryana (HR)—evaluated with standard geocoding metrics described in Section 4.1.3.

4.3.1 Bridging Local Semantics with Global Structure. Graphormer captured global spatial context, while the PTLM encoder extracted token-level semantics. To combine these views, we introduced KMCA, a fusion module that aligns the global node embedding with each token in the address (Figure 3). The PTLM outputs token embeddings $X_v \in \mathbb{R}^{l \times d}$, and Graphormer produces a global vector $H_v \in \mathbb{R}^d$. Each token embedding is cross-attended with H_v, then concatenated with the original sequence to form a fused $l \times 2d$ representation, which is projected back to $l \times d$ using MLP for decoding (Section 3.2).

We repeated $\mathbf{H}_v$ across the sequence length to form a matrix $\mathbf{M}^{(v)} \in \mathbb{R}^{l \times d}$ defined as:

$$\mathbf{M}^{(v)}_{i,:} = \mathbf{H}_v, \quad \forall i \in \{1, \ldots, l\} \tag{8}$$

[5]https://en.wikipedia.org/wiki/Haversine_formula

Model	Acc@1km ↑			Mean (km) ↓			P90 (km) ↓			P99 (km) ↓		
	HR	UP	KR	HR	UP	KR	HR	UP	KR	HR	UP	KR
GMaps	57.54%	43.73%	49.18%	5.71	9.84	7.88	6.30	16.23	12.56	38.29	61.48	60.33
G2PTL	66.10%	51.70%	56.80%	4.10	7.60	6.10	5.60	12.80	10.20	30.00	50.20	51.00
-w/o WSD	61.20%	47.60%	52.80%	4.90	8.60	7.00	5.90	14.80	11.30	34.10	55.90	54.20
GeoIndia	74.31%	57.30%	62.42%	3.31	6.71	5.29	5.46	9.49	9.70	24.51	44.69	48.92
-w/o WSD	69.80%	54.60%	59.70%	3.80	7.20	5.90	5.70	10.20	10.00	27.30	47.60	50.60
GeoIndia-V2	**89.62%**	**86.34%**	**87.73%**	**0.91**	**1.98**	**1.67**	**1.08**	**2.07**	**1.85**	**11.2**	**23.55**	**23.13**
-w/o WSD	76.12%	60.02%	63.23%	2.37	5.62	4.68	3.51	8.26	8.41	29.23	42.44	47.36
-w self-loop	85.30%	81.40%	84.90%	1.20	2.40	1.90	1.50	2.90	2.10	13.60	26.40	25.30

Table 2: Overall Performance Comparison.

Strategy	Acc@1km ↑		Mean (km) ↓	
	HR	UP	HR	UP
Concat	80.21%	76.30%	1.65	3.45
Mean Pool	83.90%	79.74%	1.41	2.88
Gated-Sigmoid	85.57%	81.90%	1.22	2.56
Gated-ReLU	87.13%	84.01%	1.09	2.31
KMCA	**89.62%**	**86.43%**	**0.91**	**1.98**

Table 3: Performance comparison of different fusion strategies.

#Nodes	Acc@1km ↑		Mean (km) ↓	
	HR	UP	HR	UP
2	81.40%	76.22%	1.71	3.29
4	86.34%	82.57%	1.18	2.48
6	**89.62%**	**86.43%**	**0.91**	**1.98**
8	89.03%	85.81%	0.94	2.11
10	88.25%	84.95%	1.02	2.24

Table 4: Performance comparison of different Graph Neighborhood Size.

The fused representation X'_v was used to generate the final H3-cell sequence. As shown in Table 3, we evaluated several fusion strategies, formally defined as follows:

$$\textbf{Concatenation:} \quad X'_v = [X_v \,\|\, \mathbf{M}^{(v)}] \tag{9}$$

$$\textbf{Mean Pooling:} \quad X'_v = \frac{1}{2}(X_v + \mathbf{M}^{(v)}) \tag{10}$$

$$\textbf{Gated Fusion (Sigmoid):} \quad X'_v = X_v \odot \sigma(W\mathbf{M}^{(v)} + b) \tag{11}$$

$$\textbf{Gated Fusion (ReLU):} \quad X'_v = X_v \odot \mathrm{ReLU}(W\mathbf{M}^{(v)} + b) \tag{12}$$

Here, $W \in \mathbb{R}^{d\times d}$ and $b \in \mathbb{R}^d$ are learnable parameters, σ denotes the sigmoid activation function, and $\odot$ represents element-wise multiplication. The resulting fused representation $X'_v \in \mathbb{R}^{l\times d}$ was passed to the decoder for H3-cell sequence prediction. And as shown in Table 3, KMCA consistently outperformed other strategies.

4.3.2 Effect of Graph Neighborhood Size. To assess the impact of spatial context, we varied subgraph size n_s from 2 to 10 nodes, each representing a delivery address with edges from real delivery sequences. As shown in Table 4, performance improved up to 6 nodes—Acc@1km increased and Mean distance error dropped, indicating that moderate context aids disambiguation. Beyond 6 nodes, performance gains diminished, with slight drops in accuracy and increased variance in error metrics, likely due to noise from overly dense or less relevant connections. These results highlight the need to balance subgraph size for effective, context-aware geocoding.

4.3.3 Effect of Weak Supervision. To assess the impact of weak supervision, we evaluated all baselines and GeoIndia-V2 with and without pretraining on WSD. As shown in Table 2, omitting WSD led to a consistent drop in Acc@1km and an increase in Mean error, indicating reduced generalization and robustness across all models. This effect was especially pronounced for informal or underspecified addresses, where the linguistic and spatial diversity in WSD played a crucial role. These results highlight that weak supervision provided a valuable initialization for address representation learning, enabling both our model and baselines to better handle structural variability before fine-tuning on high-quality data.

4.3.4 Effect of Inference Strategy. GeoIndia-V2 is trained on subgraphs from real delivery trajectories, but inference often involves a single address without context. We compare two inference strategies: (1) a self-loop-based setup where the same query address is replicated across six nodes, each connected only to itself (as in G2PTL), and (2) our proposed method, that generates dynamic graph during inference using historical successful deliveries to retain spatial structure. As shown in Table 2, the self-loop strategy (*-w self-loop*) leads to a notable drop in performance—for example, Acc@1km drops from 89.62% to 85.30% in HR, and from 86.34% to 81.40% in UP—with similar trends across other metrics. These results highlight the importance of spatial context at inference, enabling better disambiguation of lexically similar but geographically distant addresses, a common challenge in India.

4.3.5 Embedding-Based Analysis of Geographical Knowledge. To evaluate how well different models capture geographical structure, we visualized address embeddings using t-SNE. We sampled addresses from seven districts in Haryana and extracted embeddings

from G2PTL, GeoIndia, and GeoIndia-V2. As shown in Figure 4, GeoIndia-V2 produced the most distinct and compact district-level clusters, in contrast to the significant overlap observed in G2PTL and GeoIndia. This indicates that GeoIndia-V2 encodes stronger geographical inductive biases, allowing it to distinguish addresses not just by local semantics but also by their broader regional context.

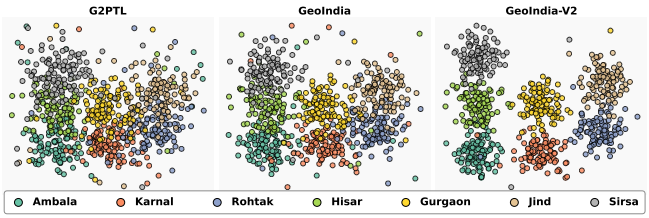


Figure 4: t-SNE visualization of address embeddings across seven districts in Haryana using models.

5 Real World Deployment

Our method leverages structured topological input during training, but inference poses challenges when relational context is missing for unseen addresses. A common fallback is using self-loop-only graphs, which preserve architectural compatibility but fail to exploit the model's structural inductive bias. To assess the value of graph topology at inference, we ran batch experiments using real delivery sequences. By reconstructing graphs with one-hop and two-hop edges between delivered addresses, we observed significant performance gains over the self-loop baseline (refer Table 2)—highlighting the importance of relational context during inference. In production, fine-grained delivery sequences are not available. To approximate relational context at inference, we construct graphs using nearest neighbor retrieval over a curated address database, based on embeddings from the GeoIndia-V2 encoder. Given a query address q, the inference-time graph is defined as $G_q = (V_q, E_q)$, where the node set is given by:

$$V_q = \{v_q, v_1, v_2, \ldots, v_5\} \tag{13}$$

where $\{v_1, \ldots, v_5\}$ are the top-5 nearest neighbors retrieved from a pre-indexed address database using Approximate Nearest Neighbor (ANN) search over GeoIndia-V2 embeddings.

To construct the edge set E_q, we define two types of connections:

One-hop edges: For each node $v_i \in V_q$, we connect it to its most similar neighbor within the subgraph based on embedding similarity:

$$E_q^{(1\text{-}hop)} = \left\{ E_{ij} \,\middle|\, v_i \in V_q,\ j = \underset{k \mid v_k \in V_q}{\overset{k \neq i}{\arg\max}} \operatorname{Sim}(\operatorname{Addr}(v_i), \operatorname{Addr}(v_k)) \right\} \tag{14}$$

Two-hop edges: To enrich the graph with indirect structure while maintaining sparsity, we introduced transitive connections. For any pair of nodes v_i and v_j, if there exists a common neighbor $v_k \in V_q$ such that:

$$E_q^{(2\text{-}hop)} = \left\{ E_{ij} \,\middle|\, E_{ik}, E_{kj} \in E_q^{(1\text{-}hop)},\ k \neq i, j;\ v_i, v_j, v_k \in V_q \right\} \tag{15}$$

$$E_q = E_q^{(1\text{-}hop)} \cup E_q^{(2\text{-}hop)} \tag{16}$$

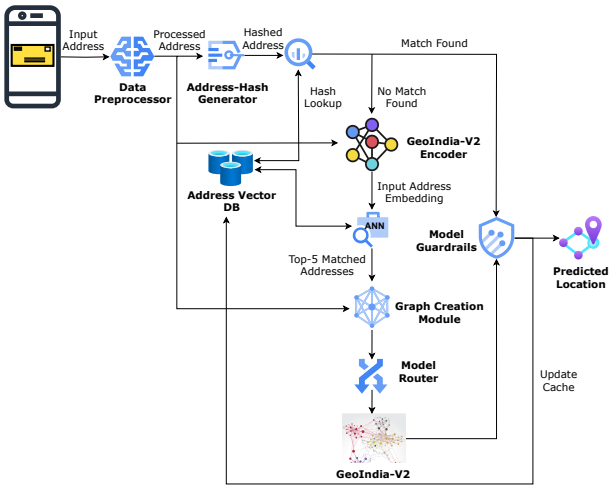


Figure 5: GeoIndia-V2 Real-Time Inference Workflow

Self-loops were intentionally omitted to encourage the model to rely on relational structure rather than identity shortcuts. Our ANN-based graph construction provided a practical approximation of real delivery graphs, enabling contextual inference without explicit relational data. Currently, we are observing a median latency of ~495 ms for GeoIndia-v2. Figure 5 illustrated the pipeline, highlighting the method's scalability and production readiness.

6 Conclusion and Future work

In this work, we introduced GeoIndia-V2, a novel geocoding framework that explicitly fuses geospatial and semantic representations through our proposed Key Modulated Cross-Attention (KMCA) mechanism. By integrating a Graphormer-based geospatial encoder with a domain-specific PTLM, our system explicitly models spatial topology and linguistic variability, to the best of our knowledge, has not been operationalised in prior geocoding systems. In contrast to previous approaches we also proposed dynamic graph creation at inference time, enabling fine-grained neighbourhood modelling, and adopted a decoder-based generative strategy for hierarchical H3-cell prediction, which captures dependencies across bits. Our method consistently outperforms industry baselines such as GeoIndia and Google Maps on multiple evaluation metrics, demonstrating both accuracy and robustness in handling India's complex address ecosystem. Extensive ablation studies further validate the effectiveness of our design choices, including dynamic graph generation and KMCA-based fusion. We also aim to extend our proposed framework to other real-world applications such as high-precision address matching, address quality assessment, and address enrichment, further demonstrating its versatility across downstream geospatial tasks.

7 GenAI Usage Disclosure

During the preparation of this work, the authors used generative AI (GenAI) tools in limited, supportive roles under direct human supervision. Specifically:

- **Manuscript Writing and Revision:** GenAI tools (e.g., ChatGPT) were used to assist in refining the structure and phrasing of certain sections of the manuscript, including the abstract, related work, and ablation study descriptions. All AI-generated suggestions were manually reviewed, edited, and verified by the authors to ensure technical accuracy, originality, and alignment with the research contributions.
- **No Use in Code or Data:** No generative AI tools were used in the development of the model architecture, training code, data preprocessing pipeline, or during experimental evaluation. All implementation and data processing were performed manually by the authors.

The authors affirm that they take full responsibility for the content of this paper, and that the use of GenAI tools complies with ACM's authorship and publication policies.

References

[1] Hao Yuan Bai and Xue Liu. 2025. T-Graphormer: Using Transformers for Spatiotemporal Forecasting. *arXiv preprint arXiv:2501.13274* (2025).

[2] Santanu Bhattacharya. 2018. Poor Addresses Cost India $10–14 Billion a Year. Available at: Medium Article.

[3] Shichang Ding, Xiangyang Luo, Jinwei Wang, and Xiaoming Fu. 2023. GNN-Geo: a graph neural network-based fine-grained IP geolocation framework. *IEEE Transactions on Network Science and Engineering* 10, 6 (2023), 3543–3560.

[4] GoogleMaps. [n. d.]. Google Maps Geocoding API. https://developers.google.com/maps/documentation/geocoding. https://developers.google.com/maps/documentation/geocoding.

[5] Chenghua Huang, Shisong Chen, Zhixu Li, Jianfeng Qu, Yanghua Xiao, Jiaxin Liu, and Zhigang Chen. 2024. Geoagent: To empower llms using geospatial tools for address standardization. In *Findings of the Association for Computational Linguistics ACL 2024*. 6048–6063.

[6] Wanjing Huang, Tongjie Pan, and Yalan Ye. 2025. Graphormer-Guided Task Planning: Beyond Static Rules with LLM Safety Perception. *arXiv preprint arXiv:2503.06866* (2025).

[7] Konstantin Klemmer, Nathan S Safir, and Daniel B Neill. 2023. Positional encoder graph neural networks for geographic data. In *International conference on artificial intelligence and statistics*. PMLR, 1379–1389.

[8] Govind Kothari and Saurabh Sohoney. 2022. Learning Geolocations for Cold-Start and Hard-to-Resolve Addresses via Deep Metric Learning. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing: Industry Track*. 322–331.

[9] Song Li, Jingjing Huo, Wenqiang Xu, Maozong Zheng, Kezun Zhang, Hong Chen, and Xuan Lin. 2023. Label-Guided Graphormer for Hierarchy Text Classification. In *International Conference on Artificial Neural Networks*. Springer, 536–547.

[10] Linlin Liang, Yuanfei Chang, Yizhuo Quan, and Chengbo Wang. 2024. A Hierarchy-Aware Geocoding Model Based on Cross-Attention within the Seq2Seq Framework. *ISPRS International Journal of Geo-Information* 13, 4 (2024), 135.

[11] Jonathan Lin, Aditya Joshi, Hye-young Paik, Tri Dung Doung, and Deepti Gurdasani. 2025. RACCOON: A Retrieval-Augmented Generation Approach for Location Coordinate Capture from News Articles. *arXiv preprint arXiv:2501.11440* (2025).

[12] Cheng Liu, Xinyang He, Shiliang Su, and Mengjun Kang. 2023. A graph-based method for Chinese address matching. *Transactions in GIS* 27, 3 (2023), 859–876.

[13] Yan Lu, Hongli Liu, and Yanquan Zhou. 2019. Chinese address standardization based on seq2seq model. In *Proceedings of the 2019 2nd International Conference on Computational Intelligence and Intelligent Systems*. 1–5.

[14] Zhaorui Ma, Shicheng Zhang, Xinhao Hu, Na Li, Qinglei Zhou, Fenlin Liu, Hongjian Wang, Guangwu Hu, and Qilin Dong. 2023. GWS-Geo: A graph neural network based model for street-level IPv6 geolocation. *Journal of Information Security and Applications* 75 (2023), 103511.

[15] Saket Maheshwary and Saurabh Sohoney. 2023. Learning geolocation by accurately matching customer addresses via graph based active learning. In *Companion Proceedings of the ACM Web Conference 2023*. 457–463.

[16] Arpan Paul, Saket Maheshwary, and Saurabh Sohoney. 2024. Accurate Customer Address Matching via Weak Supervision for Geocode Learning. In *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems*. 454–464.

[17] Chunyao Qian, Chao Yi, Chengqi Cheng, Guoliang Pu, and Jiashu Liu. 2020. A coarse-to-fine model for geolocating Chinese addresses. *ISPRS International Journal of Geo-Information* 9, 12 (2020), 698.

[18] Razorpay. 2020. Introducing Address Correction For E-Commerce Businesses. https://razorpay.com/blog/shipping-address-ecommerce.

[19] Yaswanth Reddy, Sumanth Sadu, Abhinav Ganesan, and Jose Mathew. 2022. Address Location Correction System for Q-commerce. In *Proceedings of the Second International Conference on AI-ML Systems*. 1–7.

[20] Yu Shi, Shuxin Zheng, Guolin Ke, Yifei Shen, Jiacheng You, Jiyan He, Shengjie Luo, Chang Liu, Di He, and Tie-Yan Liu. 2022. Benchmarking graphormer on large-scale molecular modeling datasets. *arXiv preprint arXiv:2203.04810* (2022).

[21] Bhavuk Singhal, Anshu Aditya, Lokesh Todwal, Shubham Jain, and Debashis Mukherjee. 2024. GeoIndia: A Seq2Seq Geocoding Approach for Indian Addresses. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*. 395–407.

[22] Yuval Solaz and Vitaly Shalumov. 2023. Transformer based geocoding. *arXiv preprint arXiv:2301.01170* (2023).

[23] Minjie Wang, Da Yu, Da Zheng, Quan Gan, Yizhou Gai, Zihao Ye, Mufei Li, Jinjing Zhou, Chengfan Qian, Hao Wang, et al. 2020. Deep Graph Library: Towards Efficient and Scalable Deep Learning on Graphs. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*.

[24] Lixia Wu, Jianlin Liu, Junhong Lou, Minhui Deng, Jianbin Zheng, Haomin Wen, Chao Song, and Shu He. 2024. G2PTL: A Geography-Graph Pre-trained Model. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management*. 4991–4999.

[25] Qinchen Yang, Zhiqing Hong, Dongjiang Cao, Haotian Wang, Zejun Xie, Tian He, Yunhuai Liu, Yu Yang, and Desheng Zhang. 2024. AddrLLM: Address Rewriting via Large Language Model on Nationwide Logistics Data. *arXiv preprint arXiv:2411.13584* (2024).

[26] Zhengcong Yin, Diya Li, and Daniel W Goldberg. 2023. Is ChatGPT a game changer for geocoding-a benchmark for geocoding address parsing techniques. In *Proceedings of the 2nd ACM SIGSPATIAL International Workshop on Searching and Mining Large Collections of Geospatial Data*. 1–8.

[27] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. 2021. Do transformers really perform badly for graph representation? *Advances in neural information processing systems* 34 (2021), 28877–28888.

[28] Bing Zha and Alper Yilmaz. 2023. Subgraph Learning for Topological Geolocalization with Graph Neural Networks. *Sensors* 23, 11 (2023), 5098.

[29] Chen Zhang, Biao He, Renzhong Guo, and Ding Ma. 2023. A graph-based approach for representing addresses in geocoding. *Computers, Environment and Urban Systems* 100 (2023), 101937.