Transactions in GIS

WILEY

Transactions in GIS

RESEARCH ARTICLE OPEN ACCESS

Omni Geometry Representation Learning Versus Large Language Models for Geospatial Entity Resolution

Kalana Wijegunarathna[1] | Kristin Stock[1] | Christopher B. Jones[2]

[1]School of Mathematical and Computational Sciences, Massey University, Auckland, New Zealand | [2]School of Computer Science and Informatics, Cardiff University, Cardiff, UK

Correspondence: Kalana Wijegunarathna (k.wijegunarathna@massey.ac.nz)

Received: 19 April 2025 | **Revised:** 23 December 2025 | **Accepted:** 14 January 2026

Keywords: database merging | deduplication | entity resolution | gazetteer matching | gazetteer merging | geospatial entity resolution | large language models

ABSTRACT

The development, integration, and maintenance of geospatial databases rely heavily on efficient and accurate matching procedures of Geospatial Entity Resolution (ER). While resolution of points-of-interest (POIs) has been widely addressed, resolution of entities with diverse geometries has been largely overlooked. This is partly due to the lack of a uniform technique for embedding heterogeneous geometries seamlessly into a neural network framework. Existing neural approaches simplify complex geometries to a single point, resulting in significant loss of spatial information. To address this limitation, we propose Omni, a geospatial ER model featuring an omni-geometry encoder. This encoder is capable of embedding point, line, polyline, polygon, and multi-polygon geometries, enabling the model to capture the complex geospatial intricacies of the places being compared. Furthermore, Omni leverages transformer-based pre-trained language models over individual textual attributes of place records in an Attribute Affinity mechanism. The model is rigorously tested on existing point-only datasets and a new diverse-geometry geospatial ER dataset. Omni produces up to 12% (F1) improvement over existing methods. Furthermore, we test the potential of Large Language Models (LLMs) to conduct geospatial ER, experimenting with prompting strategies and learning scenarios, comparing the results of pre-trained language model-based methods with LLMs. Results indicate that LLMs show competitive results.

1 | Introduction

Location-based services (LBS) are computer applications that cater to the user or device based on their current location (Raper et al. 2007). Vital to the robust functioning of all geographic information systems is a geospatial database providing adequate coverage and quality. These databases often store the place names (i.e., toponyms), place type, their geospatial footprint (as a point location or a complex geometry), and sometimes addresses, relations between places, quality attributes, temporal attributes, etymologies, officiality of toponyms, etc. However, researchers and application developers frequently encounter the need to merge geospatial databases (or search results from them) due to factors such as incomplete coverage in individual databases, disparate attribute focuses, or variations in the quality of certain attributes (Sun et al. 2023).

Entity resolution (ER) (Christophides et al. 2020; Köpcke et al. 2010) is the task of identifying different descriptions that represent the same real-world entities. The challenge in geospatial ER is rooted in the inherently multi-modal nature of geospatial data. A place is characterized not only by its textual attributes but also its geospatial footprint. Although POIs are commonly stored as point objects (simple pair of coordinates), more detailed spatial footprints in the form of polygons and lines are available in most comprehensive databases. For example, OpenStreetMap, one of the most widely used global geospatial resources, represents roughly 68% of features as polygons, 18% as lines, and only about 13% as points in New Zealand. Similarly, for the New Zealand Geographic Board data (the official gazetteer for New Zealand), when combined with their roads dataset, point geometries constitute only 33.8% data. Although these percentages vary across sources, the storage of complex geometries is the norm rather than the exception. Classical machine learning approaches that tackle this problem with engineered features exist (Hastings 2008; Martins 2011). Unfortunately, their data are not readily available. The current state-of-the-art methods for geospatial ER do not accommodate these complex geometries (Acheson et al. 2020; Balsebre et al. 2022, 2023; Zhou et al. 2021) primarily due to the absence of a unified embedding technique. Current neural approaches simplify these geometries to point objects, resulting in a loss of information or more commonly only consider point datasets for their evaluations (Balsebre et al. 2022, 2023; Zhou et al. 2021).

Advancing GeoAI necessitates encoding spatial data, including points, polygons, lines, and networks, into a representation compatible with neural network inputs (Mai, Huang, et al. 2023). These embeddings can subsequently be applied across a wide array of geospatial tasks, including but not limited to spatial relation prediction, geography-enhanced question answering, cartographic generalization, and building pattern classification. This area has received attention in recent research efforts, with a growing emphasis on representing non-Euclidean data (Bronstein et al. 2017). In downstream tasks where input sources have multiple types of geometries, an encoder capable of handling them in a single mini-batch becomes indispensable for a deep learning framework.

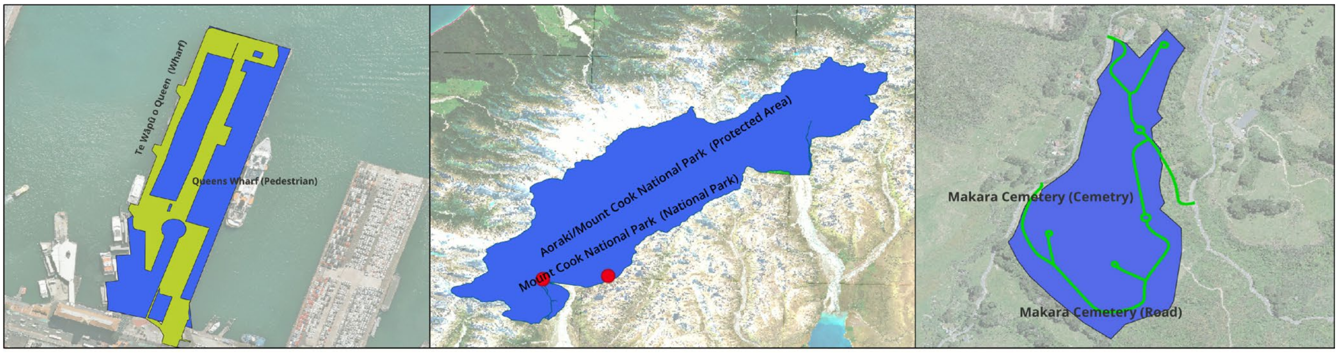
Figure 1 illustrates some of the acute challenges and nuances of ER in a geospatial context using real-world examples.

Limitation in textual similarity measures: Figure 1a shows two entries from two sources for the same wharf. A simple string similarity measure over the names of the two places, or their types (indicated within parentheses), shows minimal similarity. Semantic similarity between the place types also does not indicate strong similarity. However, upon considering all textual attributes along with the spatial footprints, it becomes evident that the two sources are referring to the same wharf. Both Figure 1a and b demonstrate the challenge of multilingualism, vernacular names, and different typing schemes when matching places (Laurini 2015).

Inadequacy of simple point-to-point geographic distance measures: Figure 1b provides an example of a true match in which the polygons overlap almost perfectly (a tiny sliver of the underlying green polygon is visible in the eastern region of the park). Both of these source databases store point locations in addition to the polygons. However, the point locations from the two sources, indicated by the red points, are over 7 km apart, not perfectly matching on the toponym or the place type. Imperfect matching on the name and type attributes combined with the possibility of such large spatial distances pose a significant challenge for existing distance-based methods that do not account for complex geometries. This is especially true for places with larger spatial extents (Ahlers 2013). An ER method capable of exploiting the details of the complex polygons will also enable a subsequent trivial resolution of the points as they are often internally linked to its complex geometry within a single source. However, point location sources can be highly error-prone even within a single database, making distances between places across databases highly unreliable (Ahlers 2013; Gao et al. 2017).

Need for individual attention to attributes: Figure 1c illustrates a non-match, where the place names are a perfect match and the footprints of the two entities are in close proximity and overlap. While these attributes may obfuscate the ER task, the key to its non-match lies in a single attribute: the place type. This underscores the necessity of individually attending to pairs of textual attributes when comparing entities from different databases. Current methods only consider summary representations of pairs of entities that fall short of understanding the structured nature and semantics of attribute value pairs in ER datasets (Paganelli et al. 2023).

To this end, we propose Omni, a model uniquely capable of addressing these challenges. Omni consists of three modules: a language module, a geographic distance module, and the Omni-GeoEncoder—the geospatial footprint encoder. We enable our model to learn from all available textual attributes of the places using the language module. Concurrently, we make the model


Te Wāpū o Queen (Wharf)
Queens Wharf (Pedestrian)
Aoraki/Mount Cook National Park (Protected Area)
Mount Cook National Park (National Park)
Makara Cemetery (Cemetry)
Makara Cemetery (Road)

(a) Queens Wharf - Te Wāpū o Queen : Match (b) Aoraki – Mount Cook national Park : Match (c) Makara cemetery - Makara cemetery road : Non-Match

FIGURE 1 | Illustration of the challenges of geospatial ER. (a) and (b) show examples of matches while (c) shows a non-match. The type of the place is indicated within parentheses. Zoom in for best view.

aware of geometric or topological relations between the footprints of the two places using our Omni-GeoEncoder module. To the best of our knowledge, this is the only encoder capable of uniformly embedding diverse geometry types. Finally, using our geo-distance module, we embed several pertinent distances to enhance the model's understanding of the spatial relations between the two places.

Recent LLMs such as GPT4 (Achiam et al. 2023), Llama (Touvron et al. 2023), and PaLM (Chowdhery et al. 2023) have established state-of-the-art performances in a variety of downstream tasks (Peeters et al. 2023; Wang et al. 2023; Wang, Yang, et al. 2024; Zhu et al. 2024). Although recent applications of LLMs in generic ER have seen them surpass pre-trained language model (PLM)-based approaches (Fan et al. 2024; Kasinikos and Papadakis 2024; Li et al. 2024; Narayan et al. 2022; Peeters et al. 2023; Wang, Chen, et al. 2024), LLMs have not yet been utilized in geospatial ER. We adapt LLMs for this task, exploring their spatial understanding and ability to match spatial entities, testing on numerous prompts and learning techniques. We compare and contrast their performance with existing PLM-based methods and our novel Omni model.

The key contributions of this paper are summarized as follows:

1. We propose Omni, an open source architecture providing a unified framework leveraging spatial and textual information from source databases for geospatial ER.
2. We develop the Omni-GeoEncoder, capable of encoding heterogeneous geometry types into a uniform embedding space, allowing neural models to comprehend spatial and topological relations of geospatial footprints. We demonstrate the effectiveness of this module in ER and geospatial relation mining.
3. NZER: The first publicly available dataset for the task of geospatial ER with diverse-geometry types from real-world databases.
4. We leverage LLMs for the task of geospatial ER. We explore their capabilities in zero-shot, few-shot, and fine-tuned settings.
5. Extensive experiments comparing (and demonstrating the benefit of) Omni and the LLM-based approach with existing methods on point-only and diverse-geometry datasets.

The remainder of the paper is organized as follows: We first introduce the technical background of ER within the broader context of database integration, followed by an overview of language models relevant to our setting. We then review related work on geospatial ER, representation learning for geospatial data, and the use of LLMs in ER. Section 4 presents the Omni framework and our methodology for applying LLMs to this task. We then report the experimental setup, analysis, and results in Section 5, before concluding the paper.

2 | Background

This section provides the foundational concepts underlying our work. We first outline the principles of ER and formalize the problem in the context of database integration, with a focus on the unique challenges posed by geospatial records. We then summarize key developments in language models, including both pre-trained language models (PLMs) and large language models (LLMs), and describe how these models are typically used in downstream tasks such as ER.

2.1 | Entity Resolution

Entity resolution, also known as entity matching, is a crucial task in database integration (Köpcke et al. 2010; Li et al. 2020; Wang et al. 2012). Given two source databases, $D1 = \left\{e_1^1, e_2^1, e_3^1, \ldots, e_m^1\right\}$ and $D2 = \left\{e_1^2, e_2^2, e_3^2, \ldots, e_n^2\right\}$, where e_i is a single record in the database, the goal of ER is to identify pairs of entities from both source databases that refer to the same real-world entity. In generic ER, $e_i = \left\{t_1, t_2, t_3, \ldots, t_k\right\}$, where t_k is a textual attribute. Although generic database records are not limited to textual attributes, geospatial database or gazetteer records are unique due to each record being characterized by a geospatial footprint, g_i, in addition to textual attributes.

Traditionally, string similarity measures have been widely used to capture textual attribute similarity (Köpcke et al. 2010; Sehgal et al. 2006; Smart et al. 2010; Wang et al. 2012). Methods relying on PLMs pass pairs of serialized entities $\left(Ser\left(e_i\right), Ser\left(e_j\right)\right)$ to the PLM and treat ER as a binary classification task (Balsebre et al. 2022, 2023; Brunner and Stockinger 2020; Li et al. 2020; Paganelli et al. 2023; Peeters and Bizer 2021; Zeakis et al. 2023), often using the [CLS] token as the representation of the pair of entities. ER methods using LLMs are based on a prompt consisting of a task description defining the ER task, together with a pair of serialized entities (Fan et al. 2024; Kasinikos and Papadakis 2024; Li et al. 2024; Narayan et al. 2022; Peeters et al. 2023; Wang, Chen, et al. 2024).

All geospatial ER methods rely on some distance measure to assess the level of match in the geospatial footprint. As all existing neural geospatial ER methods only use point locations, they are limited to using a distance measure to capture spatial similarity (Balsebre et al. 2022, 2023; Zhou et al. 2021).

2.2 | Language Models

Language models (LMs) are foundational tools in natural language processing (NLP), designed to understand, generate, and manipulate human language. Among the most impactful advancements in NLP are pre-trained language models and large language models which have set new benchmarks by harnessing massive datasets and sophisticated architectures. PLMs focus on leveraging pre-training with fine-tuning for specific tasks, while LLMs extend this approach by scaling up model size and data, achieving remarkable generalization across diverse language tasks. At the core of most modern LMs lies the transformer architecture (Vaswani 2017), a paradigm-shifting innovation in deep learning. Transformers eschew traditional recurrence mechanisms in favor of a self-attention mechanism, enabling efficient processing of text sequences while capturing long-range dependencies. Equation (1) expresses the attention mechanism.

$$Attention(Q, K, V) = \left(\frac{QK^T}{\sqrt{d_k}} \right) V, \quad (1)$$

where Q, K, and V are the query, key, and value vectors derived from the input and d_k is the dimension of the key vectors. This mechanism calculates a weighted sum of the value vectors, where the weights are determined by the similarity between the query and the key vectors. The resulting attention scores, normalized via the softmax function, allow the model to selectively focus on the relevant parts of the input sequence (Vaswani 2017).

The most significant difference between PLMs like BERT, RoBERTa (Liu et al. 2019), BART (Lewis 2019), and LLMs like GPT4, Llama, and PaLM is the scale of the models and the amount of training data. While both frameworks leverage unsupervised pre-training to learn generalizable features, LLMs are characterized by their scale and flexibility, making them suitable for emergent capabilities like instruction-following and creative generation. On the other hand, PLMs are optimized for task-specific fine-tuning with relatively smaller parameter sizes.

For many downstream tasks, including ER, PLMs are used to obtain contextual embeddings of text sequences. These models are often fine-tuned on annotated training datasets to allow the embeddings to capture the context. Conversely, decoder-only LLMs often use their text generation ability in downstream tasks. There are several approaches to utilizing LLMs effectively in downstream tasks:

1. Zero-shot prompting: A task description is provided along with the instance on which the model needs to make a prediction. With no access to training data, the model makes a prediction solely relying on the knowledge acquired during its pre-training and the provided prompt itself. This method does not require gradient-based fine-tuning or updates to the model's parameters.
2. Few-shot prompting: The model is provided with a few task-specific examples within the prompt. In an ER setting, the prompt could include examples of serialized entities along with their corresponding labels. No model weight updates are required.
3. Fine-tuning: A labeled dataset is used to update the model's weights through backpropagation, tailoring the model to a specific downstream task. Fine-tuning can be performed in multiple ways:
   - Full fine-tuning: All parameters of the original model are updated.
   - Parameter-efficient fine-tuning methods: Techniques like Low-Rank Adaptation (LoRA) introduce a small number of task-specific parameters while keeping most of the pre-trained model's parameters frozen. This approach is particularly advantageous in resource-constrained settings.

3 | Related Work

3.1 | Geospatial Entity Resolution

In addition to being studied as an information retrieval task, entity resolution also appears in the gazetteer conflation literature, where it is treated as a preliminary step in merging or integrating two or more gazetteers (Hastings 2008; Manguinhas et al. 2008; Smart et al. 2010; Wijegunarathna et al. 2025). In the early rule-based approach to geospatial ER (Hastings 2008; McKenzie et al. 2013), heuristics were used to filter places or place pairs until no duplicates remained. Later solutions involved machine learning, where textual similarities and geographical distances were converted into features for algorithms like Support Vector Machines (SVM) (Martins 2011; Sehgal et al. 2006; Zhou et al. 2021), logistic regression (Sehgal et al. 2006; Zhou et al. 2021), decision trees (Martins 2011; Zheng et al. 2010; Zhou et al. 2021), and random forests (Acheson et al. 2020). Graph-based approaches have also been popular, especially with POI conflation, and are often used in combination with spatial and linguistic similarity measures. Kim et al. (2017) build a labeled graph using information extracted from place descriptions and introduce a graph matching algorithm, essentially combining string matching with graph traversal. However, their matching is based on linguistic attributes such as the name and address of the place but does not take spatial attributes and similarity into account. Novack et al. (2018) leverage spatial similarity using bipartite graphs, representing POIs as nodes from two datasets and their edges representing possible matches with weights based on multi-criteria similarity, including Euclidean distance and WordNet (Meng et al. 2013) based semantic similarity.

Earliest use of deep learning can be seen in Santos et al. (2018), using Recurrent Neural Networks (RNN) for alternate place name classification. Subsequent work applies embedding techniques like FastText (Bojanowski et al. 2017) and Word2Vec (Mikolov et al. 2013) with Gated Recurrent Unit (GRU) models and Multilayer Perceptrons (MLPs) (Cousseau and Barbosa 2021; Yang, Hoang, et al. 2019) to capture toponym, category, and geographical similarities to identify duplicates. SkyEx (Isaj et al. 2019) and methods introduced by Deng et al. (2019) are notable exceptions to machine learning-based methods. While Deng et al. (2019) use improved Dempster-Shafer evidence theory, Isaj et al. (2019) use Pareto optimality to separate matching pairs from non-matching pairs of places from multiple sources. SkyEx shows superior performance to geographical distance and string similarity-based traditional methods (Berjawi et al. 2014; Morana et al. 2014).

Akin to generic ER (Brunner and Stockinger 2020; Li et al. 2020; Peeters and Bizer 2021; Zeakis et al. 2023), PLMs such as BERT (Devlin et al. 2018) have produced excellent results in geospatial ER. Although not specifically designed for ER, GTMiner (Balsebre et al. 2023), a graph-based geospatial relation prediction model, predicts *same_as* relations using a geo-textual interaction mechanism that combines geographic distance (Haversine distance) with BERT embeddings. GeoER (Balsebre et al. 2022) similarly uses Haversine distance but only uses the

[CLS] token from a BERT model, while also incorporating context from neighboring places through a neighborhood attention mechanism. GeoER produces excellent results, outperforming SkyEx and several other state-of-the-art ER methods adopted for the geospatial domain. He et al. (2024) present a similar approach for conflating spatial data in place knowledge graphs using the [CLS] representation of a serialized place entity embedded using a PLM. Although using the [CLS] token's representation is standard practice in classification tasks including ER (Balsebre et al. 2022; Brunner and Stockinger 2020; Li et al. 2020; Zeakis et al. 2023), recent research suggests that it does not fully capture the semantic similarities (or differences) in comparable attribute pairs. Furthermore, BERT and its variants struggle to fully grasp the structured nature of ER datasets (Paganelli et al. 2023).

3.2 | Representation Learning for Geospatial Data

Point encoders can be categorized into two types: encoders that represent a single point (or location) using only the location of the point, and encoders that incorporate the neighboring information of the point when encoding a single location (Mai et al. 2022). The first approach includes methods such as discretized grids with one-hot encoding (Tang et al. 2015), normalized latitude and longitude with MLPs (Chu et al. 2019; Xu et al. 2018), and encoding geographic coordinates using deterministic functions such as sinusoidal functions (Mac Aodha et al. 2019). Methods that aggregate the neighbor information often model the neighborhood as a point cloud. Kernel-based encoders (Mai et al. 2020; Yin et al. 2019), graph convolutional networks (Valsesia et al. 2018), CNN-based encoder-decoder architectures Li et al. (2018) and MLPs (Qi et al. 2017) are some of the many methods that have been experimented with for this approach. We refer to Mai et al. (2022) for a detailed review.

Apart from point encoders, attempts have also been made to encode and embed polygons and polylines. Polyline embeddings obtained using LSTMs have been used for the problem of trajectory prediction (Xu et al. 2018; Zhang et al. 2019). CNN-based architectures have been used to embed polygons (Mai, Jiang, et al. 2023; Veer et al. 2018). Additionally, Mai, Jiang, et al. (2023) propose a conversion of polygonal geometries into a spectral domain using a non-uniform Fourier transform, which is then embedded using an MLP. Yan et al. (2021) take a different approach employing a graph convolutional autoencoder's bottleneck layer representation as the latent space embedding of the polygon. To the best of our knowledge, no existing method attempts to encode and embed different types of geometry in a single encoder.

3.3 | Entity Resolution With LLMs

Narayan et al. (2022) first employed LLMs for ER, testing OpenAI's GPT-3 model, experimenting with various prompt designs for both in-context learning and zero-shot learning. Their findings demonstrated that GPT-3 achieved results comparable to those of PLM-based methods. Similarly, Wang, Chen, et al. (2024) introduce novel prompting strategies that diverged from the traditional pairwise matching approach commonly used in ER. They explore "comparison" and "selection" prompting strategies along with the traditional "matching" technique.

Peeters et al. (2023) further investigated the impact of prompt variations on performance across multiple LLMs. Their results underscored that fine-tuning LLMs could lead to substantial improvements in ER performance. Additionally, LLMs have been leveraged to enhance traditional and PLM-based ER approaches, as demonstrated by Li et al. (2024). In parallel, other studies have focused on the use of smaller models and cost-effective prompting strategies for ER to address the computational cost associated with LLM-based approaches (Fan et al. 2024; Kasinikos and Papadakis 2024). No research appears to have been conducted on utilizing LLMs for geospatial ER.

4 | Methodology

4.1 | Omni: Methods and System

In this section, we introduce Omni, our framework for performing entity matching. Omni performs matching on a set of entity pairs, $C = \left\{ \left(e_i, e_j\right) \mid e_i \in D_1, e_2 \in D_2 \right\}$ where D_1 and D_2 are the databases being merged and e is an individual place from the database. Omni consists of three modules that capture and compare different attributes of place pairs: (1) A pre-trained language module enhanced with Attribute Affinity generation, (2) Geographic distance embedding module, and (3) Omni-GeoEncoder. The model overview is shown in Figure 2.

4.1.1 | Language Module With Attribute Affinity

Each place record in a geospatial database, e_i, has a set of textual attributes such as the name, place type, address, and postal code. Though comparison of toponyms with a string similarity metric in the case of mono-lingual databases can be highly effective, it fails to capture changes of names (outdated names in one or more sources) and vernacular or unofficial names. It can also be inadequate when toponyms are multilingual. Other possible textual attributes like place type may have very little string similarity across sources (due to the use of different typing schemes) but often exhibit semantic relationships.

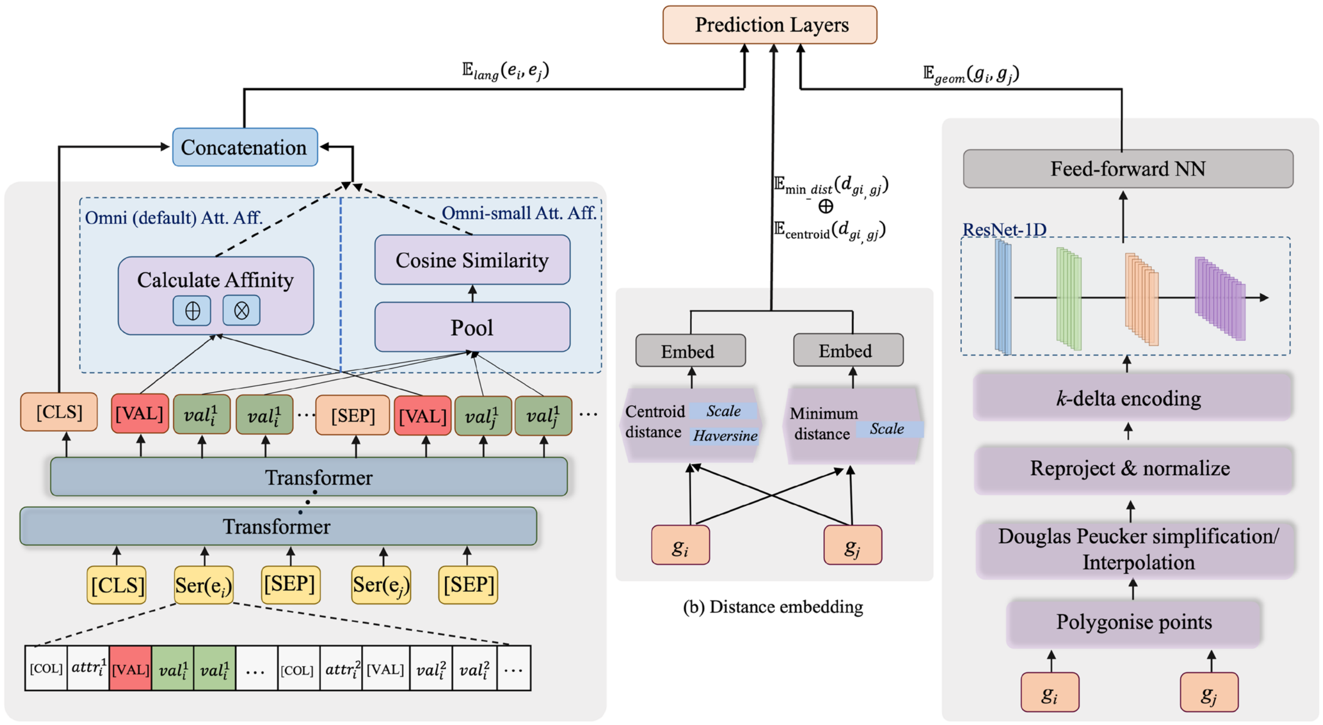
PLMs can be used to obtain highly contextualized semantic embeddings, making them especially useful in NLP tasks. Following previous ER approaches (Balsebre et al. 2022, 2023; Li et al. 2020), we serialize the textual attributes pertaining to a single entity from our sources in the following format:

$$Ser\left(e_i\right) = [COL]attr_i^1[VAL]val_i^1 \dots [COL]attr_i^H[VAL]val_i^H. \quad (2)$$

Subsequently, the serialized textual attributes of the pairs of places are combined.

$$Text\_input\left(e_i, e_j\right) = [CLS]Ser\left(e_i\right)[SEP]Ser\left(e_j\right)[SEP]. \quad (3)$$

Attribute affinity generation: While earlier PLM based ER approaches use the final embeddings of the [CLS] token from the

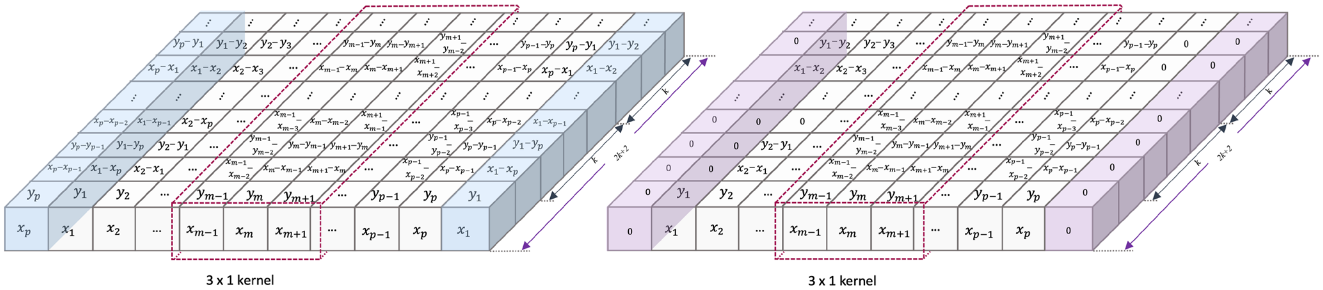


(a) Language module with Attribute Affinity

(b) Distance embedding

(c) Omni-GeoEncoder

FIGURE 2 | Illustration of the proposed Omni architecture. Both Attribute Affinity generation strategies are shown in (a), (b) and (c) show the distance embedding module and the GeoEncoder.



(a) Omni-Kdelta encoded polygon geometry with padding

(b) Omni-Kdelta encoded line geometry with padding

FIGURE 3 | Omni-Kdelta encoding. Edge columns with solid fill indicate padding and red dotted line shows a 3×1 kernel. (a) Vertex KDelta neighbors are cyclic. Circular padding is used on the complete KDelta encoding. (b) At edge vertices, KDelta neighbors are acyclic. Zero padding is used on the complete KDelta encoding.

language model (*lm*), $\mathbb{E}_{lm}(CLS)$, to represent the similarities between entities, recent research indicates that this method is inadequate to capture finer grained semantic differences in comparable textual attributes (Paganelli et al. 2023). The study also suggests that PLMs like BERT, pre-trained primarily on masked language modeling and next sentence prediction, do not fully comprehend the structure of ER datasets. To grasp the semantic similarities between corresponding attributes, we design an Attribute Affinity mechanism. We propose two variations:

(1) Default: A concatenation of the embeddings of the counterpart attributes with their Hadamard product [optimal operations were empirically determined, similar to Reimers (2019)]. We use the [VAL] token to represent the value for each attribute. The affinity of a single attribute between two entities is shown in Equation (4). Note that $\oplus$ indicates a tensor concatenation.

$$Affinity_{i,j}^{attr^h} = \left[\mathbb{E}_{lm}\left(VAL_i^h\right) \oplus \mathbb{E}_{lm}\left(VAL_j^h\right)\right] \oplus \left[\mathbb{E}_{lm}\left(VAL_i^h\right) \cdot \mathbb{E}_{lm}\left(VAL_j^h\right)\right]. \tag{4}$$

(2) Pooled cosine similarity: For a more concise representation of affinity, we pool the token embeddings associated with each attribute and calculate the cosine similarity with the corresponding representation from the other entity. Equation (5) shows affinity between two entities for a single attribute, $attr^h$.

$$Affinity_{i,j}^{attr^h} = \frac{\left(m_i^{attr^h}\right)^{\top} m_j^{attr^h}}{\parallel m_i^{attr^h} \parallel \parallel m_j^{attr^h} \parallel}, \tag{5}$$

where $m_i^{attr^h}$ and $m_j^{attr^h}$ are the pooled representations of the tokens for $attr^h$ for entities i and j, respectively. With this variation, $Affinity_{i,j}^{attr^h} \in [-1, 1]$ yields a single scalar value per attribute.

Finally, Equation (6) shows the final output of the language module. Note that the $\sum \oplus$ is used to represent a series of concatenation operations.

$$\mathbb{E}_{lang}(e_i, e_j) = \mathbb{E}_{lm}(CLS) \oplus \left(\sum_{h=1}^{H} \oplus Affinity_{i,j}^{attr^h} \right). \tag{6}$$

4.1.2 | Distance Embedding Module

Capturing geographic distance is a vital aspect of any geospatial ER framework. The choice of distance (or similarity) measure for geospatial ER is a widely discussed research topic. While a simple point-to-point distance is adequate for point-only datasets, the ideal choice for diverse geometries is more nuanced. Distance measures like centroid-to-centroid distance, minimum distance, maximum distance, Fréchet distance, and Hausdorff distance have been proposed in various ER and non-ER geometry comparing tasks (Ghosh et al. 2018; Lei and Lei 2022; Xavier et al. 2016).

Omni processes diverse geometries in a uniform manner, and as a neural framework, this requires handling heterogeneous geometries in a single mini-batch. Fréchet distance, for example, is defined on ordered sets of points and is particularly effective for comparing polylines in trajectory analysis. However, in our setting, the direction of traversal is not explicit in geometries across sources, and enforcing direction invariance is computationally expensive (Lei and Lei 2022). Moreover, the Fréchet distance cannot be directly applied to point–polygon or point–line comparisons, and for point–point comparisons, it reduces to simple Euclidean distance. While a highly engineered Fréchet distance is applicable in principle, this is not a practical solution within a neural framework that must process diverse geometries in the same mini-batch.

Methods like (Balsebre et al. 2022, 2023) use Haversine distance between point locations. We adapt this Haversine distance embedding as a centroid-to-centroid distance. The centroid-to-centroid distance is a widely used distance measure when comparing complex geometries in ER tasks (Hastings 2008; Martins 2011) and offers a simple yet generalizable distance measure across diverse geometries. This measure is particularly effective because many sources store the centroid of a complex polygon as the feature's point representation; consequently, centroid-based comparison provides a strong alignment signal when matching a complex geometry with its corresponding point representation or with a simplified point representation from a different source. Another viable and generalizable candidate distance measure is the Hausdorff distance. We conducted preliminary experiments to compare the performance of Hausdorff distance vs. centroid-to-centroid distance and found that the latter yields better performance in practice (see Section 5.6).

However, with complex geometries, centroid-to-centroid distance alone is not an adequate representation of the geospatial distance. Therefore, we supplement the distance module with a minimum distance measure (Acheson et al. 2020; Hastings 2008; Martins 2011). This minimum distance module uses the geometry normalization used in the Omni-GeoEncoder (Section 4.1.3). The minimum distance $d_{i,j}$ between the two geometries is scaled using the maximum normalized distance, max_norm_dist and embedded using a linear layer with two learnable parameters, α_{min_dist} and β_{min_dist}.

$$\mathbb{E}_{min\_dist}(d_{i,j}) = \alpha_{min\_dist}^{\top} \left(\frac{d_{i,j}}{max\_norm\_dist} - 1 \right) + \beta_{min\_dist}. \tag{7}$$

The centroid-to-centroid Haversine distance embedding and the minimum distance embedding are concatenated to obtain the final distance embedding.

4.1.3 | Omni-GeoEncoder

As discussed previously, a deep learning entity resolution model can benefit immensely by learning representations of complex geometries and their geometrical relationships. Representing places with complex geometries as points always results in a loss of information. Hence, we propose a novel geometry encoder: Omni-GeoEncoder that is capable of encoding complex geometries of varying types and also creating embeddings of the geometries that capture geometrical relations between them. We leverage CNNs adapting a ResNet architecture (He et al. 2016), inspired by Mai, Jiang, et al. (2023).

Firstly, given a pair of geometries (g_i, g_j), belonging to the two entities e_i and e_j, if any of the geometries is a point, we transform the point geometry to a simple circular disk with a nominal radius of 1 m and P vertices. Indeed, no physical place on Earth can be accurately represented as a zero-dimensional point. We test this approach with several exclusively point datasets (Section 5.4). Henceforth, this entity's point geometry is replaced by the circular disk geometry. Polygons with holes are simplified by removing the holes. For encoding purposes, all geometries need to be represented with a fixed number of vertices, P. Using a larger P value will result in a more detailed geometry (see Section 5.6 for the empirical determination of the ideal P value). If the number of vertices of geometry $|g|$ is greater than P, we use a modified Douglas-Peucker algorithm (Douglas and Peucker 1973) to decimate the geometry (polygon, multi-polygon, line or polyline) into a geometry of fixed number of P vertices. Instead of recursively removing all vertices that lie beyond a distance of ϵ, we order the vertices according to importance and retrieve the top P most important vertices, taking care to preserve first and last vertices in all cases. In the case of multi-polygons and polylines, the number of vertices allocated to each polygon or line segment is calculated proportional to the area or length respectively. Conversely, if the number of vertices in the original geometry is less than P, we do an equidistant interpolation to increase the number of vertices to P.

Subsequently, we carry out a projection of the geometries from their original datum to a planar projection. This projection enables easier distance calculation between vertices and normalization required for our subsequent steps. Next, the two geometries are normalized to a [−1, 1]×[−1, 1] 2D unit space using a common minimum bounding box. This resulting pair of geometries is then encoded using our Omni-Kdelta encoding, padded, and subsequently passed on to the ResNet1D encoder to obtain the embeddings of the geometries.

Omni-KDelta encoder: KDelta encoding is a preliminary encoding that is used to add the neighborhood structure information of each vertex to the encoding of each vertex, reducing the need for very deep encoders (Mai, Jiang, et al. 2023). We adapt the KDelta encoder, enabling it to encode both polygonal and linear geometries. This encoding treats a series of vertices (be it lines or polygons) as a 1D coordinate sequence. A geometry g is represented as:

$$\left[x_1, y_1, x_2, y_2, \ \dots, x_{m-1}, y_{m-1}, x_m, y_m, x_{m+1}, y_{m+1}, \ \dots, x_{P-1}, y_{P-1}, x_P, y_P\right].$$

KDelta encoding for the mth vertex, (x_m, y_m), can be shown as follows:

$$\begin{aligned} c_m = [&x_m, y_m, x_m - x_{m-k}, y_m - y_{m-k}, \ \dots, x_m - x_{m-1}, \\ &y_m - y_{m-1}, x_m - x_{m+1}, y_m - y_{m+1}, \ \dots, x_m - x_{m+k}, y_m - y_{m+k}]. \end{aligned} \tag{8}$$

To identify neighboring polygons in edge cases, that is, when $m - k < 0$ or $m - k \geqq P$, Omni-KDelta encoding uses a circular padding for polygons. Conversely, lines use zero padding as per Equation (9). This reflects the difference of the cyclic nature of a series of polygonal vertices and the acyclic nature of a line series.

$$m \begin{cases} < k; \forall l \ when \ k - m - l < 0, x_l \leftarrow x_m; y_l \leftarrow y_m; \\ \qquad \therefore x_m - x_l = 0; y_m - y_l = 0 \\ \quad k < m \leqq p - k; \text{follow Equation (8)} \\ \geqq p - k; \forall l \ when \ l < P - m, x_l \leftarrow x_m; y_l \leftarrow y_m; \\ \qquad \therefore x_m - x_l = 0; y_m - y_l = 0 \end{cases} \tag{9}$$

Thus, we obtain the final Omni-KDelta encoding for the whole geometry, $\mathbb{C} = [c_1^{\top}, c_2^{\top}, \ \dots, c_m^{\top}, \ \dots, c_P^{\top}]$ by stacking the point encodings. Additionally, we use custom padding according to the type of geometry. We pad linear geometries with zero padding and polygon geometries with circular padding (Figure 3). Note that this is different from the neighbor padding for each vertex described above. This padding reinforces the type of geometry and respects the clear difference between the two types of geometries.

Finally, the encoded geometry $\mathbb{C}$ is input to the ResNet1D encoder. We use a standard ResNet1D architecture to obtain the embeddings. Since we have applied a custom geometry-specific padding for our geometries, we omit any padding from the first convolutional layer. $\mathbb{C}$ is passed to the first 1D-CNN with stride of 1 and no padding with l 3×1 kernels. After a subsequent 1D batch normalization layer and ReLU activation, we carry out a 1D max pooling operation with a kernel of size 2, stride of 2 and zero padding. The output is then passed to a series of R standard ResNet1D layers with zero padding. The results from the ResNet1D layers are then passed through a global max pooling layer and a dropout layer to produce the embeddings of geometry, $\mathbb{E}_{Rsnt}(g_i)$. The embeddings for g_j are similarly obtained and the two embeddings $\mathbb{E}_{Rsnt}(g_i)$ and $\mathbb{E}_{Rsnt}(g_j)$ are concatenated and passed on to a fully connected neural network with ReLU activation and a dropout layer to learn spatial and topological relations between the geometries.

In conclusion, using the language module, we have captured the relations between the textual attributes, $\mathbb{E}_{lang}(e_i, e_j)$, not only by the summary representation of the serialized textual attributes, $\mathbb{E}_{lm}(CLS)$, but also by training the model to focus on pairs of attributes that should be compared for matching through attribute affinity generation, $Affinity_{i,j}^{attr}$. With our distance embedding model, we have focused on two distances: capturing minimum distance, $\mathbb{E}_{min\_dist}$, and centroid-to-centroid Haversine distance, $\mathbb{E}_{centroid}$. We leverage the Omni-GeoEncoder to embed the two geometries ($\mathbb{E}_{Rsnt}(g_i)$, $\mathbb{E}_{Rsnt}(g_j)$) and learn a combined representation, learning the spatial and topological relations between the geometries $\mathbb{E}_{geom}(g_i, g_j)$. Finally, we carry out a concatenation of these representations and pass it to an MLP for prediction.

$$\begin{aligned} &Prediction(match \mid e_i, e_j) = softmax(MLP([\mathbb{E}_{lang}(e_i, e_j) \\ &\oplus \mathbb{E}_{min\_dist}\left(d_{g_i, g_j}\right) \oplus \mathbb{E}_{centroid}\left(d_{g_i, g_j}\right) \oplus \mathbb{E}_{geom}(g_i, g_j)])). \end{aligned} \tag{10}$$

4.2 | LLMs for Geospatial ER

In this section, we detail the methods used to leverage large language models for geospatial ER: the learning strategies and the prompt variations used.

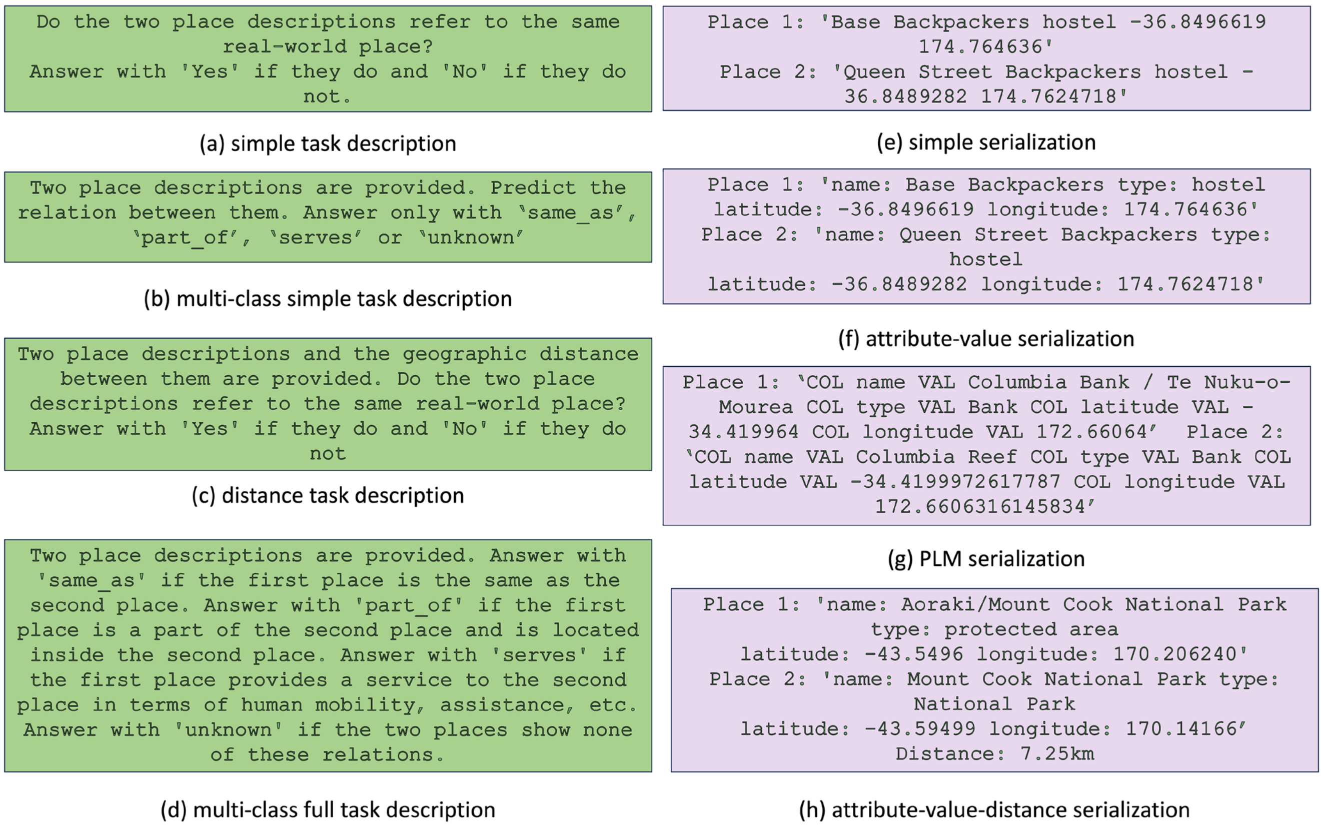
4.2.1 | Scenario 1: Zero-Shot Prompting

In contrast to PLMs, LLMs have demonstrated remarkable zero-shot capabilities (Kojima et al. 2022). In the zero-shot prompting scenario, we evaluate the performance of the LLM without using any training data. Various prompt variations are tested by adopting existing prompts from generic ER tasks and designing more domain-specific prompts tailored to the geospatial nature of the task. At its core, each prompt includes a task description and a serialized input of the two places to be matched. Following a building-block approach, we combine different task descriptions with various entity serialization formats. All task descriptions specify the format of the answer: either "Yes" or "No" in the entity resolution setting or one of the four predefined labels in the multi-class relation classification problem in GTMiner dataset (GTMD) (see Section 5.2).

Figure 4 details the task descriptions and the serialization formats used for each prompt. Examples for all prompt designs can be found in the project repository.[1]

Zero-shot prompts tested are listed below:

1. *simple*: This prompt is adapted from the *domain-complex-force* prompt for generic ER (Peeters et al. 2023). Diverging from their original prompt, the two entities to be matched are explicitly defined as "places" and "place descriptions." For ER, this prompt combines the task description in



```
Do the two place descriptions refer to the same
real-world place?
Answer with 'Yes' if they do and 'No' if they do
not.
```

(a) simple task description

```
Two place descriptions are provided. Predict the
relation between them. Answer only with ‘same_as’,
‘part_of’, ‘serves’ or ‘unknown’
```

(b) multi-class simple task description

```
Two place descriptions and the geographic distance
between them are provided. Do the two place
descriptions refer to the same real-world place?
Answer with 'Yes' if they do and 'No' if they do
not
```

(c) distance task description

```
Two place descriptions are provided. Answer with
'same_as' if the first place is the same as the
second place. Answer with 'part_of' if the first
place is a part of the second place and is located
inside the second place. Answer with 'serves' if
the first place provides a service to the second
place in terms of human mobility, assistance, etc.
Answer with 'unknown' if the two places show none
of these relations.
```

(d) multi-class full task description

```
Place 1: 'Base Backpackers hostel -36.8496619
174.764636'
Place 2: 'Queen Street Backpackers hostel -
36.8489282 174.7624718'
```

(e) simple serialization

```
Place 1: 'name: Base Backpackers type: hostel
latitude: -36.8496619 longitude: 174.764636'
Place 2: 'name: Queen Street Backpackers type:
hostel
latitude: -36.8489282 longitude: 174.7624718'
```

(f) attribute-value serialization

```
Place 1: ‘COL name VAL Columbia Bank / Te Nuku-o-
Mourea COL type VAL Bank COL latitude VAL -
34.419964 COL longitude VAL 172.66064’   Place 2:
‘COL name VAL Columbia Reef COL type VAL Bank COL
latitude VAL -34.4199972617787 COL longitude VAL
172.6606316145834’
```

(g) PLM serialization

```
Place 1: 'name: Aoraki/Mount Cook National Park
type: protected area
latitude: -43.5496 longitude: 170.206240'
Place 2: 'name: Mount Cook National Park type:
National Park
latitude: -43.59499 longitude: 170.14166’
Distance: 7.25km
```

(h) attribute-value-distance serialization

FIGURE 4 | Prompt building blocks: Task descriptions on the left and entity serializations on the right.

Figure 4a (and entity serialization Figure 4e). For multi-class relation prediction, it combines Figure 4b,e.

2. *attribute-value (a-v)*: Variation of *simple* where the place descriptions' serialization includes the attribute type and the value as opposed to only values, providing more context to the model. For ER, this strategy combines Figure 4a,f. Multi-class relation prediction uses Figure 4d combined with Figure 4f.

3. *plm-serialization (plm-ser)*: The place descriptions are serialized in the same format as the input to the PLMs in the PLM-based solutions (Balsebre et al. 2022, 2023; Li et al. 2020) as described in Section 4.1.1. This strategy combines Figure 4a,g for ER and Figure 4d,g for multi-class relation prediction.

4. *attribute-value-distance (a-v-d)*: The *attribute-value* prompt is enhanced with the distance between the two places explicitly included in the prompt. For ER, this prompt combines Figure 4c,h. For multi-class relations, the task description in Figure 4d is altered by changing the first sentence to “Two place descriptions and the geographic distance between them are provided.” This new task description is combined with the serialization shown in Figure 4h.

4.2.2 | Scenario 2: Few-Shot Learning

We use task-specific training examples in the prompt to test the model's in-context learning ability (Dong et al. 2024). We use *attribute-value* and *attribute-value-distance* prompts during our few-shot learning experiments. In the few-shot setting, the task description is followed by several demonstrations sampled from the training split and their ground truth labels before the serialized place descriptions of the place pairs for which the model should make a prediction for. The serialization of the entities in the demonstrations is kept consistent with the test prompt. We use two sampling strategies for selecting train samples:

1. Random: Demonstrations are randomly sampled from the training datasets, with four examples utilized in our experiments. This yields two experiments *random-attribute-value (rand-a-v)* and *random-attribute-value-distance (rand-a-v-d).*

2. Class-balanced: A fixed number of demonstrations are randomly sampled from each class to ensure that the model is exposed to examples from every class. Two demonstrations from each class were used in the experiments. This too yields two experiments *class-balanced-attribute-value (cbal-a-v)* and *class-balanced-attribute-value-distance (cbal-a-v-d).*

4.2.3 | Scenario 3: Fine-Tuning

In this scenario, the train and validation splits of each dataset are used to fine-tune the LLM locally using Low-Rank Adaptation for Quantized Models (QLoRA) (Dettmers et al. 2024). First, 4-bit quantization is applied to the base model, reducing the memory footprint. This step converts high-precision floating-point values into low-precision “4-bit NormalFloats.” Subsequently, low-ranked adapter matrices focused on specific modules are introduced. Instead of training the complete model, these low-rank matrices can be learned, significantly reducing the number of trainable parameters and VRAM requirements. The models were fine-tuned on three of the prompts used for zero-shot learning: *simple*, *attribute-value*, and *attribute-value-distance*. Upon fine-tuning a model with prompt using the respective dataset's training split, the model is set to generate (or evaluation) mode to make predictions on the relevant test split.

5 | Experiments and Analysis

This section presents our experimental findings in light of the following research questions:

- RQ1: How does Omni generalize to sources containing only point locations, and how does it compare with existing PLM-based methods and the novel LLM approaches? (Section 5.4)
- RQ2: How do Omni and the LLM based methods perform on diverse-geometry datasets? (Section 5.5)
- RQ3: How effective are the novelties of the Omni model and how do they contribute to the final output of the Omni model? (Section 5.6)
- RQ4: How do the models rank in terms of parameter efficiency and inference time? (Section 5.7)

In order to assess the performance of Omni and the LLMs on geospatial ER, we implement a comprehensive set of experiments on 4 datasets originating from 6 different real-world databases covering 12 different cities and regions.

5.1 | Implementation Details

Omni is implemented using PyTorch on a single A40 GPU. We employ an Adam optimizer and a linear scheduler with a warm up of 100 steps and a learning rate of 0.0003. We trained all models for 15 epochs. As our language model for Omni, *lm*, we used the Bert-base-uncased model from HuggingFace.[2] We only used at most two attributes for Attribute Affinity generation: toponym and place type, place type and address or toponym and address. For Omni-KDelta encoding, P is set to 300 and the number of neighbors for each vertex on a single side, k, is set to 6. Number of kernels, l set to 512. R, number of standard ResNet1D layers is set to 6 with a dropout rate of 0.3. Unless specified otherwise, we use this configuration for our model.

For generative LLM based experiments, we chose a 4-bit quantized Llama-3-8B-Instruct model by Meta.[3] The model was selected based on its superior performance compared to similar sized models, open availability, and hardware limitations. We used Quantized Low-rank Adapters to fine-tune the model on a single A40 GPU.

5.2 | Datasets

SwissGeoNames dataset (SGN) (Acheson et al. 2017, 2020): Dataset resolves 400 SwissNAMES3D (S3D)[4] and 400 GeoNames (GN)[5] places from Switzerland. The dataset only publishes the IDs of 400 positive matches. Unfortunately, due to the S3D's UUID updates, we were only able to retrieve 287 of the 400 positive pairs. Ninety-three of the retrieved S3D places were enhanced with their corresponding complex geometries from S3D.

GeoER dataset (GeoD) (Balsebre et al. 2022): Dataset covers four cities (Singapore, Edinburgh, Toronto, and Pittsburgh) from three different sources: Open Street Map (OSM),[6] FourSquare (FSQ),[7] and Yelp.[8] Eight different sub-datasets are presented with two datasets for each city matching OSM-FSQ and OSM-Yelp. Although this dataset is a POI dataset, we were able to find complex geometries for some places in the datasets in their original sources on manual inspection. Unfortunately, we were unable to enhance any of these places with complex geometries, as the dataset does not offer original OSM identifiers. Therefore, GeoD will serve as a point-only dataset. This dataset will help assess Omni's ability to generalize to point-only datasets when complex geometries are not available.

GTMiner dataset (GTMD) (Balsebre et al. 2023): Created for geospatial relation mining, the dataset covers four cities from OSM and Yelp and annotates three relations: *part_of*, *same_as*, and *serves*. This dataset too is originally a point-only dataset, but it publishes the source identifiers from both OSM and Yelp. However, we were unable to rely solely on the IDs, as OSM not only updates but also re-uses its IDs. This posed a challenge in verifying whether the features in OSM at the time of preprocessing were consistent with the features in the original dataset. Consequently, we only used complex geometries of features that we could programmatically confirm as corresponding to the original records. Alongside the ID matches, we enforced other constraints: perfect matches on the name, place type, and geospatial locations. Using these stringent filtering techniques, we retrieved 0, 19, 466, and 101 complex geometries from OSM for Singapore, Toronto, Seattle, and Melbourne sub-datasets, respectively.

New Zealand Entity Resolution dataset (NZER)[9]: This is a dataset we manually annotated covering five regions across New Zealand. New Zealand, a bilingual country with two official languages (English and Te Reo Māori), offers a complex problem in string matching for place names as places across sources can have English names, Te Reo Māori names or concatenations of English and Te Reo Māori names. This should require ER methods to shift the focus from place names (which often provide the strongest signal for a match) to other attributes like footprint similarity or feature type similarity. We chose five different regions to capture the nuances of population densities, proximity to large cities, percentages of English and Māori speakers, and the differences in ratios of natural and man-made features. The five regions selected were: Auckland, Hope Blue river range, Norsewood, Northland, and Palmerston North. We used three different sources: OSM, GN and the New Zealand Geographic Board's gazetteer (NZGB) from Land Information New Zealand (LINZ).[10] For both OSM and LINZ, we utilized not only the point shapefiles traditionally used in prior work, but also the polygon, line, and dedicated road shapefiles. We then overlaid all features from the shapefiles from the regions and employed four graduate GIS students to manually annotate the matches. Initial sandbox annotation revealed a Cohen's Kappa of 0.95 indicating a high inter-annotator agreement. We maintained a 30:1 ratio of negatives to positives which is suggested in the literature to reflect the real world situation (Acheson et al. 2020; Sehgal et al. 2006). NZER is the first manually annotated, publicly available dataset that allows complex geometries. Further details on the datasets are given in Table 1.

5.3 | Methods Compared

For a comprehensive analysis, we test Omni, the LLM-based methods, and the existing SOTA methods on all datasets. Here we list all methods compared:

- GTMiner (Balsebre et al. 2023) is a geospatial relation prediction model. For GeoD, NZER, and SGN, the classification layer is modified to carry out binary classification.
- GTMiner(ExRe) (Balsebre et al. 2023) is a knowledge graph refinement algorithm applied on top of the GTMiner relation predictor. This only applies to GTMD.
- GeoER (Balsebre et al. 2022) is a geospatial ER model. We extend its classification layer to predict multiple relations for GTMD. For NZER, we apply its blocking mechanism only on the train splits but not on the test and valid splits for a fair comparison. Furthermore, to support GeoER's neighborhood attention mechanism, we use their neighbor search algorithm to create neighboring entities for NZER.
- Zero-shot, Few-shot & Fine-tuned See Section 4.2.
- Omni-small is the variation of the Omni model using a mean pooled cosine similarity for the attribute affinity mechanism. We extend its classification layer to predict multiple relations for GTMD.

TABLE 1 | Table summarizes the attributes of the datasets used.

Dataset	Matching type	Diverse geometry?	Complex geometry enhanced?	Regions	# of pairs	#positive pairs	%complex geometries
GeoD	Dirty-Dirty	No	No	Pittsburgh (PIT)	5001	1459	0
					5116	1622	
				Toronto (TOR)	17,858	3826	0
					27,969	5426	
				Edinburgh (EDI)	17,386	3350	0
					18,733	2310	
				Singapore (SIN)	19,243	2116	0
					21,588	2914	
SGN	Clean-Clean	No	Yes	Switzerland	8387	287	2.1
GTMD	Dirty-Dirty	No	Yes	Singapore (SIN)	26,157	12,729*	0
				Toronto (TOR)	16,979	8194*	1
				Seattle (SEA)	15,815	6610*	4.5
				Melbourne (MEL)	6117	3717*	7.5
NZER	Clean-Clean	Yes	NA	Auckland (ACK)	4001	130	48.62
				Hope Blue (HOP)	19,374	624	30.14
				Norsewood (NRS)	11,885	388	48.62
				Northland (NTH)	23,027	752	32.21
				Palmerston (PLM)	7934	254	78.92

Note: The "Diverse Geometry" column indicates whether the datasets originally included complex geometries and the next column indicates if we were able to enhance the original datasets with complex geometries from their original sources. # of matching pairs for GeoD shows OSM-FSQ first followed by OSM-YELP subsets. *For the purposes of this summary representation, GTMD's number of positive pairs count all pairs that are not of the "unknown" type. For the exact distribution of relations in GTMD, refer to the original paper Balsebre et al. (2023).

- Omni is the default Omni model. We extend its classification layer to predict multiple relations for GTMD.

5.4 | Point-Only Datasets (RQ1)

Table 2 reports the performance of the models on a point-only dataset, GeoD. As an ER model, GeoER outperforms GTMiner on all of GeoD sub-datasets. Despite being limited to only point geometries, Omni outperforms GeoER in all but two sub-datasets. It should be noted that all fine-tuned ER models compete very closely in this point-only dataset. It is also important to note that Omni only uses attributes from the two entities being compared and does not use the additional neighborhood details that GeoER leverages. These results attest to Omni's ability to generalize to point-only data. These results also confirm the intuitive observation that augmenting a theoretical zero-dimensional point location as a two-dimensional disk does not have any detrimental effect on the results.

Omni-small uses a mean pooling strategy (see Section 4.1.1). However, the pooling produces some loss of information. Other pooling strategies tested produced similar or worse results. In a dirty data setting, with sparsely populated columns, the use of the [VAL] token's representation for Attribute Affinity, as in the default Omni model, consistently produces better results, as evident in the experiments with GeoD.

The effectiveness of prompts shows massive variations depending on the sub-dataset. This is consistent with the findings of Peeters et al. (2023) in generic ER. In general, the LLM favors a simple prompt, as is evident from the comparatively superior results produced by a *simple* prompt in a zero-shot setting. This is stressed in the poor performance of the complex and verbose *plm-serialization*. In the same setting, the *attribute-value-distance* prompt consistently outperforms the *attribute-value* prompt. This suggests the LLM's inability to calculate geographic distance on its own from the coordinates provided when the distance is not explicitly provided in the prompt.

In general, few-shot learning produces better results than zero-shot, although results show large variations. Randomly sampling the training dataset produced better results than ensuring class balance in the demonstrations. This can be attributed to the class imbalance present in all of these datasets. In the class-balanced sampling setting, larger numbers of false positives were recorded, resulting in a drastic drop in precision as the model appears to carry a bias created by the balanced demonstrations.

TABLE 2 | Comparison between SOTA PLM-based methods, our LLM, and Omni on point-only datasets (F1%).

	GeoD							
	PIT		**TOR**		**EDI**		**SIN**	
Methods	**OSM-YELP**	**OSM-FSQ**	**OSM-YELP**	**OSM-FSQ**	**OSM-YELP**	**OSM-FSQ**	**OSM-YELP**	**OSM-FSQ**
PLM baselines								
GeoER	**97.11**	92.65	95.87	93.35	96.64	94.90	**92.45**	88.90
GTMiner	95.83	92.23	95.52	87.79	95.40	94.15	80.98	87.51
Zero-shot LLM								
simple	68.39	63.58	67.23	56.81	85.95	68.71	52.64	43.80
a-v	42.08	38.42	50.35	37.24	53.6	46.00	35.99	31.39
plm-ser	21.16	21.41	19.14	9.26	22.08	15.51	8.27	11.29
a-v-d	67.17	70.25	65.54	66.35	56.83	61.92	44.81	39.86
Few-shot LLM								
rand-a-v	68.35	82.99	77.92	67.11	92.50	83.79	58.50	67.72
rand-a-v-d	80.28	81.71	84.89	86.19	93.35	91.70	80.71	63.57
cbal-a-v	70.15	82.51	27.26	40.83	87.45	45.71	64.28	70.10
cbal-av-d	78.48	87.37	53.66	50.18	91.49	63.74	73.10	71.72
Fine-tuned LLM								
simple	96.24	92.90	95.03	94.79	93.19	94.25	91.62	85.70
a-v	96.98	93.71	95.33	94.65	95.16	93.49	90.14	88.20
a-v-d	96.57	**93.90**	95.47	94.42	94.51	94.50	91.31	87.90
Omni-small	95.43	91.88	95.31	93.66	95.96	94.72	90.91	88.65
Omni	96.68	93.19	**96.77**	**94.92**	**97.58**	**95.46**	92.36	**89.40**

Note: Bold denotes best performance. Underlined numbers indicate the next best results. All PLM results are averages of three tests.

The fine-tuned LLMs demonstrate closely competitive performance, with none of the prompts used for fine-tuning emerging as a definitive one-size-fits-all solution for the task. An interesting observation made was the absence of a clear distinction between LLMs fine-tuned on *attribute-value* and *attribute-value-distance*. The lack of a clear difference in results as seen in the zero-shot setting indicates the LLM's ability to calculate or deduce distances from the provided coordinate pairs upon fine-tuning.

5.5 | Diverse-Geometry Datasets (RQ2)

Omni's improvements become more prominent with datasets that contain higher numbers of complex geometries (Table 3). Omni produces the best results among all the tested models except in two sub-datasets: NZER's Auckland sub-dataset and GTMD's Singapore sub-dataset. It should be noted that the Singapore sub-dataset is essentially a point-only dataset. Although with minimal modification to predict multiple relation classes, Omni shows outstanding improvements over GTMiner on GTMD, especially with sub-datasets that we were able to enhance with their recovered original complex geometries. The performance gain resulting from the addition of this small number of geometries is compelling. With the enhancement of just 4.5% complex geometries in SEA (GTMD) and 7.5% in MEL (GTMD), the F1 score improves by over 10% and nearly 5% respectively. This is a clear indication of the loss of information when simplifying complex features to points. These improvements can mainly be attributed to the Omni-GeoEncoder's spatial insights.

Omni demonstrates significant gains in the NZER dataset, outperforming existing PLM-based state-of-the-art (SOTA) methods by up to ~14% in F1 score in certain sub-datasets. The Auckland region proves to be distinctly challenging for all methods due to three reasons: (i) Large number of polyline geometries (streets); (ii) Very close proximity of all entities as a dense urban region; (iii) This region has the highest percentage of textually dissimilar names (Māori and English) in annotated matches. From the PLM-based methods, Omni produces the best results, as it is not limited by the simplification of linear geometries to simple points (like GeoER or GTMiner) resulting in a minimal information loss. It is also aided by the Attribute Affinity mechanism's ability to capture finer grained semantic similarities (from attributes like place type) where the [CLS] token's summary representation is inadequate to identify matching entities with completely dissimilar names.

Although not a direct comparison, we highlight the original results reported by Acheson et al. (2020) on the full SGN

TABLE 3 | Comparison between SOTA PLM-based methods, our LLM, and Omni on diverse-geometry datasets (F1%).

		GTMD				NZER				
Methods	**SGN**	**SIN**	**TOR**	**SEA**	**MEL**	**ACK**	**HOP**	**NRS**	**NTH**	**PLM**
PLM baselines										
GeoER	91.66	85.97	85.32	78.59	84.98	72.67	95.93	86.73	92.13	88.45
GTMiner	92.84	**90.71**	89.15	81.10	86.92	62.82	95.19	88.59	92.89	92.56
GTMiner(ExRe)	—	89.92	87.64	80.97	85.68	—	—	—	—	—
Zero-shot LLM										
simple	48.48	13.30	13.67	27.67	18.91	63.82	69.65	69.56	75.51	58.66
a-v	73.23	43.68	46.53	42.28	60.59	46.66	53.16	42.62	75.69	27.43
plm-ser	54.23	46.58	44.14	40.93	56.21	44.44	10.08	8.33	24.48	10.00
a-v-d	37.62	32.26	23.29	37.53	26.01	55.07	81.98	79.16	68.43	60.19
Few-shot LLM										
rand-a-v	74.07	28.07	36.93	24.81	22.99	78.26	64.74	69.49	76.63	79.51
rand-a-v-d	54.16	22.58	37.64	19.01	14.55	78.26	71.85	61.99	70.90	72.72
cbal-a-v	78.12	66.41	63.57	48.11	65.28	79.71	69.47	66.66	78.78	81.15
cbal-a-v-d	86.95	65.91	70.19	67.46	56.68	73.68	77.88	66.66	86.03	85.71
Fine-tuned LLM										
simple	94.20	73.52	54.26	44.89	35.94	81.00	89.27	93.47	92.75	94.71
a-v	93.33	60.10	70.23	54.01	53.36	81.36	81.88	92.64	93.61	91.80
a-v-d	91.89	75.30	79.93	72.31	75.07	**86.90**	89.65	91.30	92.18	82.75
Omni-small	94.37	89.27	89.00	90.51	90.66	82.10	98.22	93.85	95.24	95.11
Omni	**96.10**	89.58	**90.36**	**91.33**	**90.87**	84.64	**98.92**	**96.75**	**95.77**	**96.38**

Note: Bold denotes best performance. Underlined numbers indicate the next best results. All PLM results are averages of three tests.

dataset. Using their best-performing combination of engineered machine-learning features, they achieve an F1 score of 90.2% on the complete dataset. On the subset of this dataset that was recoverable, Omni achieves an F1 score exceeding 96%. It is also worth noting that Acheson et al. (2020) incorporate additional information—such as landcover and elevation—that is not used by any of the other methods evaluated in our study.

Omni-small produces more competitive results in a clean-clean ER setting as seen in NZER. GTMD is also a much cleaner and complete dataset than GeoD Balsebre et al. (2023). It can be concluded that the pooled cosine similarity strategy produces competitive results when the data are fully (or almost fully) populated. In both cases, selecting a limited number of well-populated attributes (as mentioned in Section 5.1) was preferable.

Contrary to what was observed in the point-only datasets, the performance of the *attribute-value-distance* prompt is poorer than the *attribute-value* prompt for GTMD. This is caused by distance being a misleading factor when it came to relations like *part_of* and *serves*. Especially if only point locations were considered, a *part of* relation can easily be misclassified as a *same_as* relation when the distance is zero. This is quite common, for example: when a stall in a mall and the mall (*part_of* relation) have the same or very close point locations. LLMs also respond better to *class-balanced* few-shot prompts than *randomly* sampled few-shot prompts with GTMD as the relations are more balanced in the dataset than in the strictly ER datasets.

The fine-tuned LLM-based approaches consistently outperform existing PLM-based methods on the strictly ER datasets like SGN and NZER. However, the performance of the LLM on GTMD is significantly lower than existing PLM-based approaches. This is mainly due to the complexity of the task as GTMD is a multi-class relation classification dataset. Although fine-tuning has led to a notable gain in performance for all prompts in the LLM, it has fallen short of learning all the nuanced relations in the dataset.

LLM fine-tuned on the *attribute-value-distance* produces excellent results on the NZER's Auckland sub-dataset, surpassing all PLM-based methods. Upon further investigation, this standout performance was discovered to be attributable to the prior knowledge acquired by the LLM during its pre-training. With exposure to massive amounts of textual knowledge, even few-shot prompts consistently outperform all PLM-based solutions except Omni. We analyze more on this improvement in Section 5.8.

In summary, Omni demonstrates strong generalizability. In addition to significantly outperforming existing baselines on our own complex geometry dataset (NZER dataset) and third party datasets that we were able to enhance with a few complex geometries (GTMD, SGN), Omni performs competitively and almost always surpasses existing baselines that can only process point data on their original point-only datasets (GeoD).

5.6 | Ablation Experiments and Analysis (RQ3)

Ablation experiments: We present the results of our ablation study conducted to verify the effectiveness of our novelties in Tables 4 and 5. We carry out four experiments: (1) No Lang.: removing the language component to rely only on distance and GeoEncoder modules; (2) No GeoEnc.: removing the GeoEncoder module; (3) No Att. Aff.: removing the Attribute Affinity generation; (4) No Dist.: removing the distance module. As expected, the language module proves to be the backbone of the framework. The No GeoEnc. experiments demonstrate the effectiveness of the GeoEncoder. The impact of the removal varies as expected, with greater reduction in performance in datasets with larger numbers of complex geometries. The effect of Attribute Affinity remains fairly constant in all datasets as this mechanism is shown to help in the challenging comparisons (see Section 5.8) where semantic similarities of specific attributes hold the key to correct predictions. The importance of the Distance module is seen to decrease as the significance of the GeoEncoder increases. While the GeoEncoder exhibits the ability to capture relative spatial distances between two features, it is not exposed to their actual geographic distance because all geometries are encoded using the K-Delta encoding (Section 3). Results indicate that the explicit distance embedding therefore plays a complementary role to the GeoEncoder. The Distance module demonstrates consistent contribution across datasets—particularly in point-only settings, where the distance embedding contributes more to performance than the GeoEncoder itself. A notable exception, as a diverse geometry dataset, is the SGN dataset where the Distance module's contribution remains more significant than the GeoEncoder's contribution owing to the very large geographic coverage of the dataset (covering places all across Switzerland as opposed to all other datasets covering

TABLE 4 | Ablation study results (F1%) on point-only datasets.

	GeoD							
	PIT		TOR		EDI		SIN	
Methods	**OSM-YELP**	**OSM-FSQ**	**OSM-YELP**	**OSM-FSQ**	**OSM-YELP**	**OSM-FSQ**	**OSM-YELP**	**OSM-FSQ**
Omni	**96.7**	**93.2**	**96.8**	**94.9**	**97.6**	**95.5**	**92.4**	**89.4**
No Lang.	88.7	68.6	90.3	87.5	86.5	84.7	74.8	62.6
No GeoEnc.	95.3	91.5	95.2	94.2	96.3	94.2	91.7	88.9
No Att. Aff.	94.9	91.6	95.0	92.8	95.5	94.3	90.1	87.8
No Dist.	95.0	89.7	94.5	92.6	95.8	92.6	87.6	85.2

Note: Bold denotes best performance.

several cities or smaller regions). In this setting, geographic distance serves as a very strong indicator of match likelihood.

Distance embedding experiments: We experimented with incorporating Hausdorff distance across several sub-datasets, comparing it against Omni's existing distance embeddings (centroid-to-centroid and minimum distance). We evaluated Hausdorff distance both as a stand-alone metric and in combination with minimum distance, replacing only the centroid-to-centroid component from the original method. The resulting F1 scores are presented in Table 6.

As the experiments suggest, performance with Hausdorff distance as a stand-alone distance measure falls significantly below the current method. Minimum distance is a highly informative signal for geospatial ER. A point in a polygon or a point lying on a line evaluates to zero with minimum distance—offering a strong indication that the two records may represent the same real-world feature. Conversely, Hausdorff distance may return a misleadingly large distance. For example, some linear features like rivers may have a single point representing them in some point-only databases (like GeoNames) or consist of multiple representations within the same database (like LINZ or OSM) where it is represented both as a linear feature and a point. In these cases, Hausdorff distance returns the distance furthest from the point, which is an inflated and misleading value.

Hausdorff distance performs much better when used in combination with the minimum distance. However, we note that it does not outperform our existing approach. This is due to the effect of the centroid representations discussed earlier (Section 4.1.2) that provides the framework a better signal about true positives. As expected, in point-only datasets, the performance of all the methods remains consistent since all methods yield effectively equivalent distance values.

Number of K-Delta vertices, an empirical analysis: The choice of P (see Section 4.1.3) is crucial in determining the quality and fidelity of the geometry representation. Figure 5 shows the results from an empirical study on the optimal value for P. For point-only datasets, the experiments reveal minimal information gain as P increases. This observation highlights a critical finding: for point-only datasets, Omni can achieve comparable performance with as few as 50 points, maintaining the same results. Conversely, for datasets containing diverse and complex geometries, the results demonstrate significant improvements as P increases. Notably, in all diverse geometry datasets except the AKL (NZER) dataset, very low P values can mislead the model, resulting in F1 scores lower than those observed in ablation tests where the GeoEncoder was completely removed. The improvements in performance generally plateau around $P = 300$ in most cases.

Generic spatial relation understanding of Omni-GeoEncoder: We designed a diagnostic task to probe the fidelity of the geometry representations learned by the Omni-GeoEncoder during the training on our geospatial ER task. This experiment evaluates how well the encoder exhibits an upstream capability

TABLE 5 | Ablation study results (F1%) on diverse-geometry datasets.

		GTMD				NZER				
Methods	**SGN**	**SIN**	**TOR**	**SEA**	**MEL**	**ACK**	**HOP**	**NRS**	**NTH**	**PLM**
Omni	**96.1**	**89.6**	**90.4**	**91.3**	**90.9**	**84.6**	**98.9**	**96.8**	**95.8**	**96.4**
No Lang.	84.6	56.3	57.2	68.2	70.5	58.8	56.3	54.0	65.0	78.2
No GeoEnc.	92.9	89.0	89.7	81.5	89.5	78.4	97.2	92.9	93.1	92.4
No Att. Aff.	94.5	87.7	86.3	89.2	89.9	80.1	98.1	95.5	94.2	95.1
No Dist.	91.9	86.8	87.7	87.5	89.2	81.8	98.0	95.9	94.6	94.5

Note: Bold denotes best performance.

TABLE 6 | F1 scores comparing Hasudorff distance with the current combination of minimum distance and centroid-to-centroid distance.

Distance embedding	**AKL (NZER)**	**NTH (NZER)**	**PLM (NZER)**	**SGN**	**PIT (OSM-YELP)**	**PIT (OSM -FSQ)**	**SIN (OSM-YELP)**	**SIN (OSM -FSQ)**
Omni	84.64	95.77	96.38	96.1	96.68	93.19	92.36	89.40
Hausdorff Distance	80.26	91.59	91.88	90.85	96.70	92.90	93.16	88.83
Minimum Distance + Hausdorff Distance	82.4	94.96	94.06	94.78	95.90	93.37	92.88	89.15

Note: Second row shows results from using Hausdorff distance by itself and the third row is with Hausdorff distance replacing only centroid-to-centroid distance in the Distance module.

(spatial relation understanding) based solely on the embedding representations acquired during ER training. For this purpose, we isolated a GeoEncoder module "pre-trained" on an ER dataset and evaluated its performance in predicting three spatial relations which are expected to give strong signals in the downstream ER task:

1. Contain: Object B (including its boundary) is fully contained inside object A.
2. Touch: The objects share a boundary but no interior points.
3. Overlap: The objects share some but not all interior points.

We created three separate datasets for each of these relations, modeling the spatial relation understanding as a binary classification problem similar to Fleuret et al. (2011), Mai, Jiang, et al. (2023), and Yang, Russakovsky, and Deng (2019). The geometries were sourced from OSM and LINZ and the relations were automatically annotated using QGIS.[11] To evaluate the isolated GeoEncoder, we implemented a classification head on top of the pre-trained GeoEncoder, training it for 10 epochs on the train split while keeping the GeoEncoder's weights frozen. The results, presented in Figure 6, highlight the spatial understanding of the GeoEncoder. GeoEncoders trained on point-only ER datasets exhibited limited spatial understanding, as expected, given their exposure only to point pairs augmented as disks with nominal radii of 1 m. Conversely, GeoEncoders trained on complex geometry datasets demonstrated excellent performance in predicting the contain and overlap relations, underscoring the K-Delta encodings' and the embeddings' ability to capture spatial relationships. However, the performance in the touch relation was notably weaker. This outcome

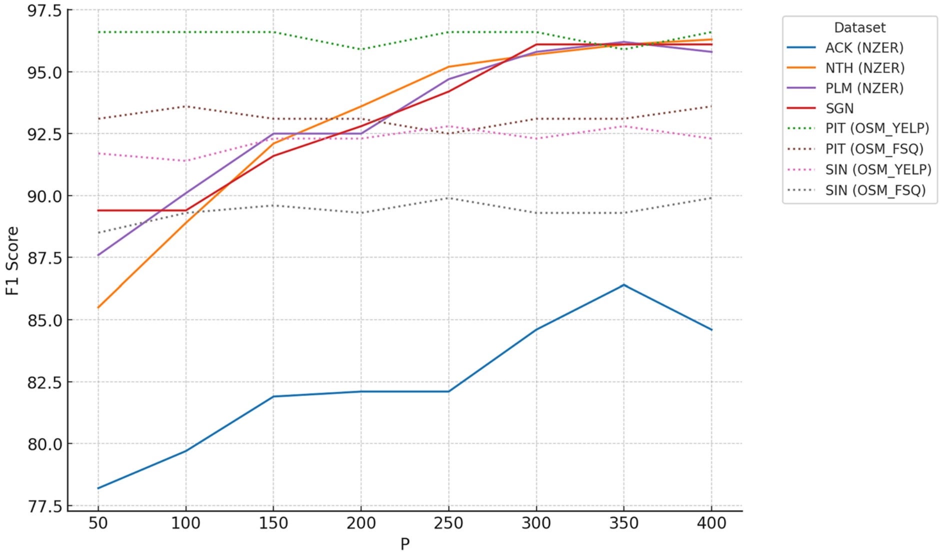


FIGURE 5 | Best performance of Omni on select sub-datasets with varying *P* values.

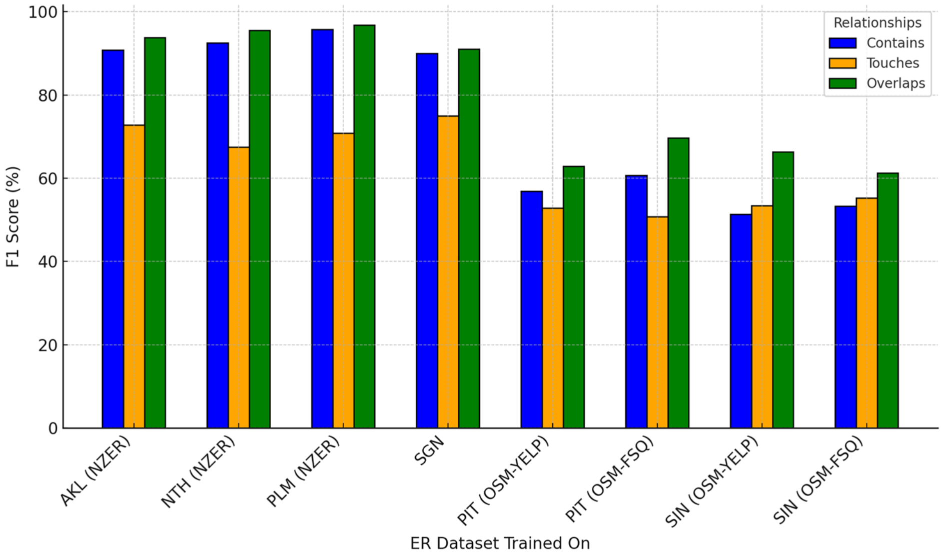


FIGURE 6 | Performance of the Omni-GeoEncoder on three generic spatial relation prediction tasks.

is unsurprising, as the touch relation requires the boundaries of two geometries to coincide, which can be as minimal as a single vertex intersection. This loss of information is caused by the simplification of geometries during the modified Douglas–Peucker decimation process.

5.7 | Model Efficiency (RQ4)

Naively comparing all records across databases when merging geospatial databases is computationally expensive, resulting in $O(n \times m)$ complexity. While various blocking techniques like spatial blocking can mitigate this cost, they still lead to a computationally expensive ER task. Consequently, efficiency becomes a critical factor in evaluating ER solutions. Table 7 compares the size and inference time of each model. Omni is significantly lighter compared to GeoER. This is primarily due to GeoER's bulky neighborhood attention mechanism. GTMiner is slightly lighter and comparable to Omni-small, though the latter usually outperforms it. What is impressive about Omni is that the added functionality of geometry encoding does not compromise inference time, as Omni remains the fastest model at inference clocking almost 50 times faster than GeoER.

The Llama-3-8B-Instruct model tested in this experiment comprises 8 billion parameters, making it significantly more resource-intensive than PLM-based solutions. Although the use of QLoRA reduces the number of trainable parameters to fewer than 200 million, in general, inference remains almost 100 times slower than Omni. In addition to their demanding VRAM requirements, this positions LLMs at the bottom of the list in terms of efficiency.

5.8 | Qualitative Analysis

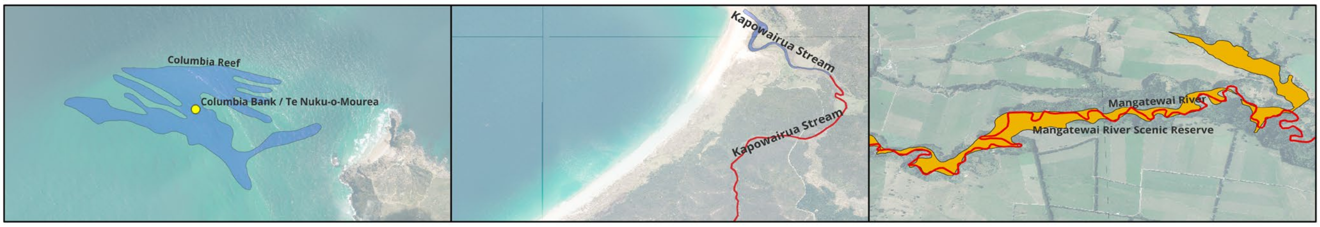
Figure 7 illustrates some examples from the NZER dataset with their predictions from the Omni model. Figure 7a resolves a point feature and a polygon feature that bear minimal textual similarity. It is also an instance of a multilingual place name where one name consists of the English name and the other is a concatenation of a different version of the English name and a Māori name. Figure 7b shows an instance of a correct prediction with minimal geospatial overlap between the polygon and the line. Figure 7c also presents an interesting case of non-matches between a reserve and a river that flows through it. While the complete Omni model correctly predicts a true negative, ablation studies reveal that removing the Attribute Affinity results in a false positive. This misclassification is attributed to the high textual and geo-footprint similarity. In such challenging cases, insight provided by Attribute Affinity on specific attributes plays a crucial role in the model's correct interpretation of the relation between places.

To investigate the exceptionally high performance of the LLMs on the NZER's Auckland sub-dataset, we designed a simple experiment: We posed the base Llama-3-8B-Instruct model the following question: “Answer the following question. What is an alternative name for <PLACE> in Auckland?” where <PLACE> was replaced with a name from our test set. Even for challenging cases such as “Te Wāpū o Queen,” the model's response, although verbose and simply predicting next token, consistently included the correct answer, “Queens Wharf,” every time (Te Wāpū o Queen is the Māori name for Queen's Wharf). Unlike PLMs, LLMs are better equipped to deal with multilingual challenges owing to their vast pre-trained knowledge. This outcome highlights two key points: the extensive knowledge LLMs acquire during pre-training on large-scale corpora and the

TABLE 7 | Table compares the weight and inference speed of the models.

Methods	Total # parameters	# Trainable parameters	Average inference time per 1000 samples
GeoER	221M	221M	80.2 s
GTMiner	112M	112M	1.83 s
In-context LLM	8B	—	158.3–208.3 s
Fine-tuned LLM	8B	167M	253.33 s
Omni-small	125M	125M	1.25 s
Omni	132M	132M	1.66 s

Note: All inference times are calculated on the NZER's Auckland sub-dataset. In-context LLMs report two inference times: Zero-shot and Few-shot.


Columbia Reef
Columbia Bank / Te Nuku-o-Mourea
Kapowairua Stream
Kapowairua Stream
Mangatewai River
Mangatewai River Scenic Reserve

(a) Point - poly : True positive

(b) Line - poly : True positive

(c) Poly - line : True negative

FIGURE 7 | Examples from the NZER dataset with predictions from the Omni model.

inherent limitations in evaluating and comparing LLMs using test datasets they may have been indirectly exposed to during pre-training.

6 | Conclusion

This work introduces a novel omni-geometry encoder and the use of a novel attribute affinity generation concept for ER in geospatial databases. Our solution is the first deep learning-based approach to perform ER on geospatial databases with complex geometries, seamlessly encoding diverse geometry types in a single encoder. The affinity generation concept shows improvement in results over a simple summary representation of the entities' textual attributes and can be generalized to generic ER. Evaluated on existing point datasets and our manually annotated diverse geometry dataset, Omni-GeoEncoder demonstrates the ability to learn and represent geometries and how these representations can be effectively used to detect spatial relationships between entities in downstream tasks. Experiments on LLMs reveal that they lack true spatial understanding in zero-shot settings. Albeit being computationally expensive, they perform competitively in few-shot and fine-tuned settings. Although they fall behind Omni in truly understanding spatial relations, LLMs demonstrate superior language capability coupled with vast prior knowledge of places. Distilling LLMs' language understanding in combination with spatial embeddings is an interesting avenue of future research.

Acknowledgments

Open access publishing facilitated by Massey University, as part of the Wiley - Massey University agreement via the Council of Australian University Librarians.

Funding

The research was funded by the New Zealand Ministry of Business, Innovation and Employment (MBIE), grant number MAUX2104.

Conflicts of Interest

The authors declare no conflicts of interest.

Data Availability Statement

This framework was programmed in Python using Jupyter Notebooks and PyTorch. The code for the Omni model can be accessed using the following link: https://figshare.com/s/3a4ebcb6c255e40d76f5. The code for the Large language model experiments can be found here: https://figshare.com/s/f45389595a58fcb669dd. The NZER dataset is hosted separately: https://figshare.com/s/e0e0481d62a3e411178b. We have also hosted the entity enhanced third party datasets here: https://figshare.com/s/7858aa81a88b2347d09d.

Endnotes

[1] https://figshare.com/s/f45389595a58fcb669dd.

[2] https://huggingface.co/.

[3] https://www.llama.com/.

[4] https://www.swisstopo.admin.ch/de/landschaftsmodell-swissnames3d.

[5] https://www.geonames.org/.

[6] https://www.openstreetmap.org/.

[7] https://developer.foursquare.com/.

[8] https://www.yelp.com/developers.

[9] https://figshare.com/s/e0e0481d62a3e411178b.

[10] https://www.linz.govt.nz/.

[11] https://qgis.org/.

References

Acheson, E., J. Villette, M. Volpi, and R. S. Purves. 2017. "Gazetteer Matching for Natural Features in Switzerland." In *Proceedings of the 11th Workshop on Geographic Information Retrieval*, 1–2.

Acheson, E., M. Volpi, and R. S. Purves. 2020. "Machine Learning for Cross-Gazetteer Matching of Natural Features." *International Journal of Geographical Information Science* 34, no. 4: 708–734.

Achiam, J., S. Adler, S. Agarwal, et al. 2023. "Gpt-4 Technical Report." arXiv Preprint arXiv:2303.08774.

Ahlers, D. 2013. "Assessment of the Accuracy of Geonames Gazetteer Data." In *Proceedings of the 7th Workshop on Geographic Information Retrieval*, 74–81.

Balsebre, P., D. Yao, G. Cong, and Z. Hai. 2022. "Geospatial Entity Resolution." In *Proceedings of the ACM Web Conference 2022*, 3061–3070.

Balsebre, P., D. Yao, G. Cong, W. Huang, and Z. Hai. 2023. "Mining Geospatial Relationships From Text." *Proceedings of the ACM on Management of Data* 1, no. 1: 1–26.

Berjawi, B., E. Chesneau, F. Duchateau, et al. 2014. "Representing Uncertainty in Visual Integration." In *DMS*, 365–371.

Bojanowski, P., E. Grave, A. Joulin, and T. Mikolov. 2017. "Enriching Word Vectors With Subword Information." *Transactions of the Association for Computational Linguistics* 5: 135–146.

Bronstein, M. M., J. Bruna, Y. LeCun, A. Szlam, and P. Vandergheynst. 2017. "Geometric Deep Learning: Going Beyond Euclidean Data." *IEEE Signal Processing Magazine* 34, no. 4: 18–42.

Brunner, U., and K. Stockinger. 2020. "Entity Matching With Transformer Architectures—A Step Forward in Data Integration." In *23rd International Conference on Extending Database Technology, Copenhagen, 30 March–2 April 2020. OpenProceedings*, 463–473.

Chowdhery, A., S. Narang, J. Devlin, et al. 2023. "Palm: Scaling Language Modeling With Pathways." *Journal of Machine Learning Research* 24, no. 240: 1–113.

Christophides, V., V. Efthymiou, T. Palpanas, G. Papadakis, and K. Stefanidis. 2020. "An Overview of End-to-End Entity Resolution for Big Data." *ACM Computing Surveys* 53, no. 6: 1–42.

Chu, G., B. Potetz, W. Wang, et al. 2019. "Geo-Aware Networks for Fine-Grained Recognition." In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 247–254.

Cousseau, V., and L. Barbosa. 2021. "Linking Place Records Using Multi-View Encoders." *Neural Computing and Applications* 33, no. 18: 12103–12119.

Deng, Y., A. Luo, J. Liu, and Y. Wang. 2019. "Point of Interest Matching Between Different Geospatial Datasets." *ISPRS International Journal of Geo-Information* 8, no. 10: 435.

Dettmers, T., A. Pagnoni, A. Holtzman, and L. Zettlemoyer. 2024. "Qlora: Efficient Finetuning of Quantized Llms." *Advances in Neural Information Processing Systems* 36: 10088–10115.

Devlin, J., M. W. Chang, K. Lee, and K. Toutanova. 2018. “Bert: Pre-Training of Deep Bidirectional Transformers for Language Understanding.” arXiv Preprint arXiv:1810.04805.

Dong, Q., L. Li, D. Dai, et al. 2024. “A Survey on In-Context Learning.” In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, 1107–1128.

Douglas, D. H., and T. K. Peucker. 1973. “Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature.” *Cartographica* 10, no. 2: 112–122.

Fan, M., X. Han, J. Fan, et al. 2024. “Cost-Effective In-Context Learning for Entity Resolution: A Design Space Exploration.” In *2024 IEEE 40th International Conference on Data Engineering (ICDE)*, 3696–3709. IEEE.

Fleuret, F., T. Li, C. Dubout, E. K. Wampler, S. Yantis, and D. Geman. 2011. “Comparing Machines and Humans on a Visual Categorization Test.” *Proceedings of the National Academy of Sciences* 108, no. 43: 17621–17625.

Gao, S., L. Li, W. Li, K. Janowicz, and Y. Zhang. 2017. “Constructing Gazetteers From Volunteered Big Geo-Data Based on Hadoop.” *Computers, Environment and Urban Systems* 61: 172–186.

Ghosh, A., B. Rozemberczki, S. Ramamoorthy, and R. Sarkar. 2018. “Topological Signatures for Fast Mobility Analysis.” In *Proceedings of the 26th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 159–168.

Hastings, J. 2008. “Automated Conflation of Digital Gazetteer Data.” *International Journal of Geographical Information Science* 22, no. 10: 1109–1127.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. “Deep Residual Learning for Image Recognition.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 770–778.

He, L., H. Li, and R. Zhang. 2024. “A Semantic-Spatial Aware Data Conflation Approach for Place Knowledge Graphs.” *ISPRS International Journal of Geo-Information* 13, no. 4: 106.

Isaj, S., E. Zimányi, and T. B. Pedersen. 2019. “Multi-Source Spatial Entity Linkage.” In *Proceedings of the 16th International Symposium on Spatial and Temporal Databases*, 1–10.

Kasinikos, I. I. A., and G. Papadakis. 2024. “Entity Resolution With Small-Scale LLMS: A Study on Prompting Strategies and Hardware Limitations.” MSc thesis, National and Kapodistrian University of Athens.

Kim, J., M. Vasardani, and S. Winter. 2017. “Similarity Matching for Integrating Spatial Information Extracted From Place Descriptions.” *International Journal of Geographical Information Science* 31, no. 1: 56–80.

Kojima, T., S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa. 2022. “Large Language Models Are Zero-Shot Reasoners.” *Advances in Neural Information Processing Systems* 35: 22199–22213.

Köpcke, H., A. Thor, and E. Rahm. 2010. “Evaluation of Entity Resolution Approaches on Real-World Match Problems.” *Proceedings of the VLDB Endowment* 3, no. 1–2: 484–493 Available from.

Laurini, R. 2015. “Geographic Ontologies, Gazetteers and Multilingualism.” *Future Internet* 7, no. 1: 1–23.

Lei, T. L., and Z. Lei. 2022. “Harmonizing Full and Partial Matching in Geospatial Conflation: A Unified Optimization Model.” *ISPRS International Journal of Geo-Information* 11, no. 7: 375.

Lewis, M. 2019. “Bart: Denoising Sequence-To-Sequence Pre-Training for Natural Language Generation, Translation, and Comprehension.” arXiv Preprint arXiv:1910.13461.

Li, H., S. Li, F. Hao, C. J. Zhang, Y. Song, and L. Chen. 2024. “Booster: Leveraging Large Language Models for Enhancing Entity Resolution.” In *Companion Proceedings of the ACM on Web Conference 2024*, 1043–1046.

Li, Y., R. Bu, M. Sun, W. Wu, X. Di, and B. Chen. 2018. “Pointcnn: Convolution on x-Transformed Points.” *Advances in Neural Information Processing Systems* 31.

Li, Y., J. Li, Y. Suhara, A. H. Doan, and W.-C. Tan. 2020. “Deep Entity Matching With Pre-Trained Language Models.” *Proceedings of the VLDB Endowment* 14, no. 1: 50–60.

Liu, Y., M. Ott, N. Goyal, et al. 2019. “Roberta: A Robustly Optimized Bert Pretraining Approach.” ArXiv: abs/1907.11692.

Mac Aodha, O., E. Cole, and P. Perona. 2019. “Presence-Only Geographical Priors for Fine-Grained Image Classification.” In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 9596–9606.

Mai, G., W. Huang, J. Sun, et al. 2023. “On the Opportunities and Challenges of Foundation Models for Geospatial Artificial Intelligence.” arXiv Preprint arXiv:2304.06798.

Mai, G., K. Janowicz, Y. Hu, et al. 2022. “A Review of Location Encoding for Geoai: Methods and Applications.” *International Journal of Geographical Information Science* 36, no. 4: 639–673.

Mai, G., K. Janowicz, B. Yan, R. Zhu, L. Cai, and N. Lao. 2020. “Multi-Scale Representation Learning for Spatial Feature Distributions Using Grid Cells.” arXiv Preprint arXiv:2003.00824.

Mai, G., C. Jiang, W. Sun, et al. 2023. “Towards General-Purpose Representation Learning of Polygonal Geometries.” *GeoInformatica* 27, no. 2: 289–340.

Manguinhas, H., B. Martins, and J. Borbinha. 2008. “A Geo-Temporal Web Gazetteer Integrating Data From Multiple Sources.” In *2008 Third International Conference on Digital Information Management*, 146–153. IEEE.

Martins, B. 2011. “A Supervised Machine Learning Approach for Duplicate Detection Over Gazetteer Records.” In *International Conference on GeoSpatial Sematics*, 34–51. Springer.

McKenzie, G., K. Janowicz, and B. Adams. 2013. “Weighted Multi-Attribute Matching of User-Generated Points of Interest.” In *Proceedings of the 21st ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 440–443.

Meng, L., R. Huang, and J. Gu. 2013. “A Review of Semantic Similarity Measures in Wordnet.” *International Journal of Hybrid Information Technology* 6, no. 1: 1–12.

Mikolov, T., I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. 2013. “Distributed Representations of Words and Phrases and Their Compositionality.” *Advances in Neural Information Processing Systems* 26.

Morana, A., T. Morel, B. Berjawi, and F. Duchateau. 2014. “Geobench: A Geospatial Integration Tool for Building a Spatial Entity Matching Benchmark.” In *Proceedings of the 22nd ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 533–536.

Narayan, A., I. Chami, L. Orr, and C. Ré. 2022. “Can Foundation Models Wrangle Your Data?” *Proceedings of the VLDB Endowment* 16, no. 4: 738–746.

Novack, T., R. Peters, and A. Zipf. 2018. “Graph-Based Matching of Points-Of-Interest From Collaborative Geo-Datasets.” *ISPRS International Journal of Geo-Information* 7, no. 3: 117.

Paganelli, M., D. Tiano, and F. Guerra. 2023. “A Multi-Facet Analysis of Bert-Based Entity Matching Models.” *VLDB Journal* 33: 1–26.

Peeters, R., and C. Bizer. 2021. “Dual-Objective Fine-Tuning of Bert for Entity Matching.” *Proceedings of the VLDB Endowment* 14: 1913–1921.

Peeters, R., A. Steiner, and C. Bizer. 2023. “Entity Matching Using Large Language Models.” arXiv Preprint arXiv:2310.11244.

Qi, C. R., H. Su, K. Mo, and L. J. Guibas. 2017. “Pointnet: Deep Learning on Point Sets for 3d Classification and Segmentation.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 652–660.

Raper, J., G. Gartner, H. Karimi, and C. Rizos. 2007. “A Critical Evaluation of Location Based Services and Their Potential.” *Journal of Location Based Services* 1, no. 1: 5–45.

Reimers, N. 2019. “Sentence-Bert: Sentence Embeddings Using Siamese Bert-Networks.” arXiv Preprint arXiv:1908.10084.

Santos, R., P. Murrieta-Flores, P. Calado, and B. Martins. 2018. “Toponym Matching Through Deep Neural Networks.” *International Journal of Geographical Information Science* 32, no. 2: 324–348.

Sehgal, V., L. Getoor, and P. D. Viechnicki. 2006. “Entity Resolution in Geospatial Data Integration.” In *Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems*, 83–90.

Smart, P. D., C. B. Jones, and F. A. Twaroch. 2010. “Multi-Source Toponym Data Integration and Mediation for a Meta-Gazetteer Service.” In *Proceedings of the International Conference on Geographic Information Science*, 234–248. Springer.

Sun, K., Y. Hu, Y. Ma, R. Z. Zhou, and Y. Zhu. 2023. “Conflating Point of Interest (Poi) Data: A Systematic Review of Matching Methods.” *Computers, Environment and Urban Systems* 103: 101977.

Tang, K., M. Paluri, L. Fei-Fei, R. Fergus, and L. Bourdev. 2015. “Improving Image Classification With Location Context.” In *Proceedings of the IEEE International Conference on Computer Vision*, 1008–1016.

Touvron, H., L. Martin, K. Stone, et al. 2023. “Llama 2: Open Foundation and Fine-Tuned Chat Models.” arXiv Preprint arXiv:2307.09288.

Valsesia, D., G. Fracastoro, and E. Magli. 2018. “Learning Localized Generative Models for 3d Point Clouds via Graph Convolution.” In *International Conference on Learning Representations*.

Vaswani, A. 2017. “Attention Is All You Need.” *Advances in Neural Information Processing Systems* 30: I.

Veer, R. V., P. Bloem, and E. Folmer. 2018. “Deep Learning for Classification Tasks on Geospatial Vector Polygons.” arXiv Preprint arXiv:1806.03857.

Wang, J., T. Kraska, M. J. Franklin, and J. Feng. 2012. “Crowder: Crowdsourcing Entity Resolution.” *Proceedings of the VLDB Endowment* 5, no. 11: 1483–1494.

Wang, L., N. Yang, X. Huang, L. Yang, R. Majumder, and F. Wei. 2024. “Improving Text Embeddings With Large Language Models.” In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, edited by L. W. Ku, A. Martins, and V. Srikumar, 11897–11916. Association for Computational Linguistics.

Wang, S., X. Sun, X. Li, et al. 2023. “Gpt-Ner: Named Entity Recognition via Large Language Models.” arXiv Preprint arXiv:2304.10428.

Wang, T., X. Chen, H. Lin, et al. 2024. “Match, Compare, or Select? An Investigation of Large Language Models for Entity Matching.” arXiv Preprint arXiv:2405.16884.

Wijegunarathna, K. I., K. Stock, and C. B. Jones. 2025. “Digital Gazetteers: Review and Prospects for Place Name Knowledge Bases.” *ACM Computing Surveys* 58, no. 3: 1–39.

Xavier, E. M., F. J. Ariza-López, and M. A. Urena-Camara. 2016. “A Survey of Measures and Methods for Matching Geospatial Vector Datasets.” *ACM Computing Surveys* 49, no. 2: 1–34.

Xu, Y., Z. Piao, and S. Gao. 2018. “Encoding Crowd Interaction With Deep Neural Network for Pedestrian Trajectory Prediction.” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5275–5284.

Yan, X., T. Ai, M. Yang, and X. Tong. 2021. “Graph Convolutional Autoencoder Model for the Shape Coding and Cognition of Buildings in Maps.” *International Journal of Geographical Information Science* 35, no. 3: 490–512.

Yang, C., D. H. Hoang, T. Mikolov, and J. Han. 2019. “Place Deduplication With Embeddings.” In *The World Wide Web Conference*, 3420–3426.

Yang, K., O. Russakovsky, and J. Deng. 2019. “Spatialsense: An Adversarially Crowdsourced Benchmark for Spatial Relation Recognition.” In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2051–2060.

Yin, Y., Z. Liu, Y. Zhang, S. Wang, R. R. Shah, and R. Zimmermann. 2019. “Gps2vec: Towards Generating Worldwide GPS Embeddings.” In *Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 416–419.

Zeakis, A., G. Papadakis, D. Skoutas, and M. Koubarakis. 2023. “Pre-Trained Embeddings for Entity Resolution: An Experimental Analysis.” *Proceedings of the VLDB Endowment* 16, no. 9: 2225–2238.

Zhang, P., W. Ouyang, P. Zhang, J. Xue, and N. Zheng. 2019. “Sr-Lstm: State Refinement for Lstm Towards Pedestrian Trajectory Prediction.” In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12085–12094.

Zheng, Y., X. Fen, X. Xie, S. Peng, and J. Fu. 2010. “Detecting Nearly Duplicated Records in Location Datasets.” In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 137–143.

Zhou, Y., M. Wang, C. Zhang, F. Ren, X. Ma, and Q. Du. 2021. “A Points of Interest Matching Method Using a Multivariate Weighting Function With Gradient Descent Optimization.” *Transactions in GIS* 25, no. 1: 359–381.

Zhu, J. P., P. Cai, K. Xu, et al. 2024. “Autotqa: Towards Autonomous Tabular Question Answering Through Multi-Agent Large Language Models.” *Proceedings of the VLDB Endowment* 17, no. 12: 3920–3933.