Efficient Representation of Location Features for Predictive Models: A Geospatial Embedding Approach

Maya Rosen & Yatir Gross

March 11, 2025

Abstract

Location features such as city names and addresses are crucial in predictive modeling but are traditionally represented as categorical strings. These representations are inefficient for machine learning models, leading to high cardinality and loss of spatial relationships. In this research, we propose an alternative approach by encoding geographic locations into 3D Cartesian coordinates on a spherical model of the Earth. By converting latitude and longitude into three-dimensional vectors, we maintain geospatial proximity while reducing feature dimensionality. Our experiments show that this representation improves predictive accuracy and computational efficiency compared to traditional encoding methods. However, the effectiveness varies across different datasets. In this document, we analyze when and why our approach outperforms existing methods, such as one-hot encoding and geohashing, and discuss cases where traditional methods may still be preferable.

1 Problem Description

1.1 Motivation

During this semester, our project focused on predicting weather conditions in various cities across the world. Early in our process, we noticed that location was treated as a simple one-hot encoded categorical feature, which raised the question: Could we find a better way to represent spatial information—and would that lead to improved predictions? This curiosity led us to choose to explore more advanced spatial encoding techniques, with the goal of enriching our model's understanding of location beyond basic city labels.

1.2 Challenges of Location Encoding

Traditional location encoding methods suffer from:

- **High Cardinality:** One-hot encoding leads to an explosion in feature dimensions.
- **Arbitrary Ordering:** Label encoding assigns numerical values that introduce unintended ordinal relationships.

- **Loss of Spatial Meaning:** Conventional encodings fail to capture the real-world distance between locations.

To address these issues, we considered two possible solutions: geohashing and a 3D vector representation of geographical coordinates. In this research, we explore both approaches, evaluating their effectiveness in predictive modeling.

2 Geohash Approach

2.1 What Is Geohash?

Geohashing[1] converts latitude and longitude into a short alphanumeric string using Base-32 encoding. The world is recursively divided into 32 cells, and each additional character refines the precision of the area being represented.

- The first character of a geohash represents a large geographical region.
- Adding more characters subdivides the area into smaller and smaller grids.
- The more characters a geohash has, the more precise the location becomes.

This hierarchical structure allows geohashing to be used at different scales, making it ideal for applications requiring both coarse and fine-grained spatial representation.

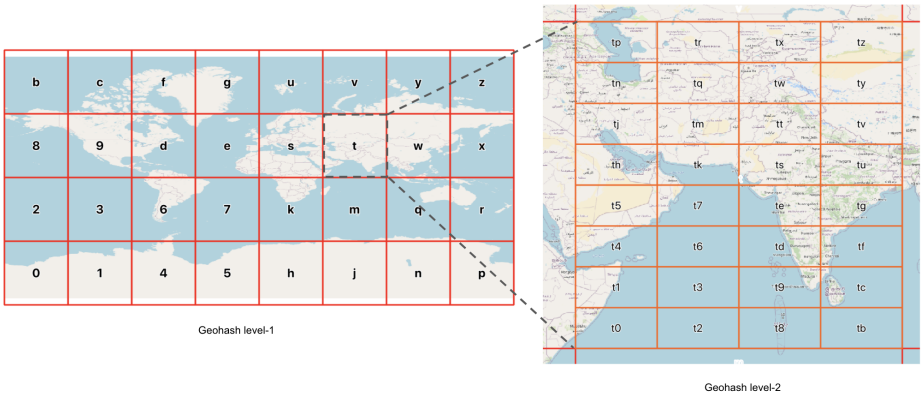

Figure 1: Illustration of geohashing and its hierarchical structure.

2.2 The Method

This method converts raw geographic coordinates into geohash strings at various precision levels (4 for broader areas and 6 for detailed regions) and calculates geohashes for neighboring cells to capture

[1] `https://en.wikipedia.org/wiki/Geohash`

adjacent influences. It then one-hot encodes these geohashes into a binary feature matrix that is merged back into the dataset, enabling efficient spatial feature engineering.

2.3 Advantages

- **Capturing Local Spatial Relationships:** By including the neighboring geohashes, the method acknowledges that geographical effects are not isolated. Locations close to one another tend to share similar characteristics, so incorporating nearby areas can improve predictive performance.
- **Flexibility Across Scales:** Using multiple geohash precisions allows the method to explore different levels of spatial granularity. This is useful in identifying whether broader spatial regions (low precision) or more localized areas (high precision) better capture the variability in the target variable.
- **Improved Model Robustness:** Including neighboring geohashes mitigates the risk of misclassification at cell boundaries and improves the robustness of the spatial features used in the prediction.

3 Solution Overview: 3D Geospatial Encoding

3.1 Mathematical Formulation

We model the Earth as a sphere of radius $R = 6371$ km and transform each location's latitude ϕ and longitude λ into Cartesian coordinates (x, y, z):[2][3]

$$\begin{aligned} x &= R\cos(\phi)\cos(\lambda) \\ y &= R\cos(\phi)\sin(\lambda) \\ z &= R\sin(\phi) \end{aligned} \tag{1}$$

where:

- ϕ is the latitude in radians.
- λ is the longitude in radians.
- R is the Earth's approximate radius in kilometers.

This transformation ensures that geographically close locations are also close in numerical representation, making it easier for machine learning models to utilize spatial relationships.

The transformation of latitude and longitude to a 3D representation can be visualized as follows:

[2] https://en.wikipedia.org/wiki/Spherical_coordinate_system#In_geography
[3] https://en.wikipedia.org/wiki/Geographic_coordinate_system

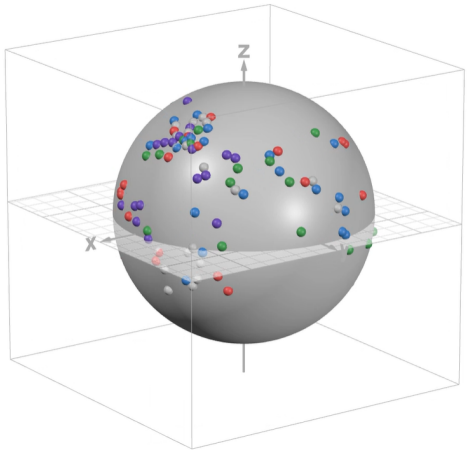

Figure 2: Illustration of the 3D location encoding on a sphere.

3.2 Advantages of the 3D Representation

The proposed 3D encoding method offers several advantages:

- **Preserves Spatial Relationships:** Unlike categorical encodings, our method ensures that distances between points in feature space reflect real-world geographic distances.
- **Efficient for Machine Learning:** The compact 3D representation significantly reduces dimensionality compared to one-hot encoding.
- **Universal Generalization and Spatial Continuity:** A 3D location representation lets the model handle any place on Earth without predefined dataset limits, enabling better generalization, smooth interpolation, and support for unseen locations.
- **Avoids Discontinuities:** Unlike geohashing, which suffers from boundary issues, our method represents location as a continuous function.

3.3 Automated 3D Coordinate Transformation for Location Data

We created the script `location_to_db.py` to process location data and store it in a database. It automates geocoding and converts location data into 3D Cartesian coordinates. It retrieves latitude and longitude using Geopy, applies spherical-to-Cartesian transformation, and updates a CSV file with the new data. The script requires a CSV file path, a city column, and optionally an address or country column. The output enhances location datasets by adding geolocation data and 3D coordinates while ensuring compatibility with geographic tools.

4 Experimental Evaluation

4.1 Datasets

We evaluate our method on datasets containing location-based categorical features, such as: We apply this method to four different datasets, where location features are crucial for prediction:

Dataset	**Location Feature**	**Target**	**Goal**
Global Weather[1]	City, Country	Rainfall (mm)	Predict rainfall
Israel Bagrut Scores[2]	School City	Avg. Grade	Predict performance
London Housing[3]	Address, Neighbourhood	Price (GBP)	Predict price
Cost of Living[4]	City, Country	Avg. Salary (USD)	Predict salary

Table 1: Datasets for 3D geospatial encoding evaluation.

4.2 Baseline Approaches

We compare our methods against:

- **One-Hot Encoding**
- **Label Encoding**

4.3 Evaluation Metrics

We measure:

- **Prediction Performance:** R-squared (R^2), Mean Squared Error (MSE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE).
- **Computational Efficiency:** Memory usage, training time.

4.4 Results

4.4.1 Weather

Method	R^2	MSE	MAE	RMSE
One-Hot Encoding	0.2093	0.2153	0.1792	0.4602
Label Encoding	0.2093	0.2153	0.1792	0.4640
Geohashing	0.2097	0.2152	0.1790	0.4639
3D Vector Encoding	**0.3313**	0.2168	0.1796	0.4656

Table 2: Performance comparison of location encoding methods on the Weather dataset.

- Geohash ($R^2 = 0.2097$) slightly improves over One-Hot ($R^2 = 0.2093$).

- 3D ($R^2 = 0.331$) gives the best result.

Why? Weather conditions are influenced by proximity, and 3D coordinates help capture distance better than simple geohashing.

4.4.2 London Houses

Method	R^2	MSE (10^{10})	MAE (10^5)	RMSE (10^5)
One-Hot Encoding	**0.9435**	**4.5537**	**1.5682**	**2.1339**
Label Encoding	**0.9435**	**4.5537**	**1.5682**	**2.1339**
Geohashing	0.9423	4.6459	1.5798	2.1554
3D Vector Encoding	0.9421	4.6639	1.5927	2.1596

Table 3: Performance comparison of location encoding methods on the London Houses dataset (values scaled for readability).

All methods perform very well ($R^2 \sim 0.94$). Geohash and 3D perform slightly worse than One-Hot and Label Encoding. **Why?** Property prices in London are highly localized to neighborhoods. One-hot encoding works well because neighborhoods already define price regions.

4.4.3 Israel Bagrut

Method	R^2	MSE	MAE	RMSE
One-Hot Encoding	**0.3396**	**53.5904**	**5.6029**	**7.3205**
Label Encoding	**0.3396**	**53.5904**	**5.6029**	**7.3205**
Geohashing	0.3388	53.6580	5.6069	7.3216
3D Vector Encoding	0.2523	59.3280	6.0144	7.7025

Table 4: Performance comparison of location encoding methods on the Israel Bagrut dataset.

Geohash and 3D perform slightly worse than One-Hot and Label Encoding. **Why?** Educational performance is influenced by school-specific factors rather than spatial location. One-hot encoding better preserves this.

4.4.4 Cost of Living

Method	R^2	MSE	MAE	RMSE
One-Hot Encoding	-735.1170	1,949,250,864	2154	44150
Label Encoding	-24.7444	68,171,636	803	8257
Geohashing	0.6428	986,381	711	993
3D Vector Encoding	**0.7959**	**563,648**	**484**	**751**

Table 5: Performance comparison of location encoding methods on the Cost of Living dataset (rounded for readability).

One-Hot has a catastrophic R^2 (-735.1), and Label Encoding is also bad (-24.74). Geohash and 3D massively improve performance (0.64 and 0.79 R^2). **Why?** Cost of living is highly location-dependent. Encoding city names doesn't help much, but geohashing captures regional differences well.

4.4.5 Conclusion

In our research, we explored different methods for integrating location features across diverse datasets. Our findings indicate that there is no universal approach—optimal performance depends on the dataset's scale and structure. The 3D coordinate system proved particularly advantageous for global datasets, such as weather, where spatial continuity is crucial. This highlights 3D encoding as a powerful tool for worldwide applications, capturing complex geographic relationships more effectively than other methods.

For localized datasets, however, neither 3D coordinates nor geohash provided significant improvements. In cases where location functions more as a categorical variable—such as neighborhoods in London—one-hot encoding or label encoding may be more suitable, as they emphasize distinct regional attributes rather than spatial proximity. While geohash can effectively model proximity-based patterns, its advantages diminish in smaller, well-defined areas where traditional categorical encoding methods are often sufficient.

5 Related Work

Geohashing has been widely used for spatial indexing and geographic data representation geohash, converting latitude and longitude into hierarchical grid cells. While effective, geohashing isn't primarily designed for predictive models as it suffers from boundary issues where adjacent geohashes may not fully capture spatial relationships. Our method has tried to apply this ecoding representation and also improves upon this by considering neighboring geohashes, which enhances predictive accuracy.

Existing work using the representation of location as a pair of longitude and latitude encodes a specific point on the Earth's surface by defining its horizontal (longitude) and vertical (latitude) coordinates

through two numerical values. Unlike this method, our approach uses a 3D Cartesian encoding of locations, ensuring spatial continuity while reducing dimensionality compared to one-hot encoding.

In contrast to traditional methods, our solution provides a smooth and compact representation of geographic locations that mitigates the limitations of discrete encodings like geohashing, offering better model generalization and improved robustness.

5.1 Inspiration for 3D representation

Our inspiration came from the advancements in natural language processing (NLP), specifically the breakthrough with large language models (LLMs). In traditional methods, words were represented as one-hot vectors, where each word was uniquely encoded without capturing any relationships between them. The innovation of embedding words into continuous vector spaces, where words with similar meanings are closer together, was a game-changer. This shift enabled models to understand the semantic connections between words, not just their individual identities. Similarly, we were motivated by the idea of representing data in a way that reflects inherent relationships, leading us to explore a more meaningful way to represent locations around the world, ultimately choosing to model locations in a 3D space, treating the Earth as a sphere.

6 Conclusion

We proposed a novel approach for encoding location features in predictive models using 3D Cartesian coordinates. By transforming latitude and longitude into a numerical representation that preserves spatial relationships, we achieved improved model efficiency and performance on global-scale datasets. However, this approach primarily enhanced performance for worldwide datasets by effectively capturing global spatial relationships. It did not significantly improve the encoding of location-specific features within smaller geographic areas, such as individual cities. Future work may explore hybrid methods combining learned embeddings with 3D vector encoding to better capture both global and local spatial characteristics.

References

1 Global Weather Dataset: `https://www.kaggle.com/datasets/nelgiriyewithana/global-weather-repos`

2 Israel Bagrut Scores Dataset: `https://www.kaggle.com/datasets/emachlev/bagrut-israel`

3 London Housing Dataset: `https://www.kaggle.com/datasets/oktayrdeki/houses-in-london`

4 Cost of Living Dataset: `https://www.kaggle.com/datasets/mvieira101/global-cost-of-living`