Planning Your Trip Using Graph Neural Networks

Sirui Chen*
Stanford University
siruic@stanford.edu

Wei Zhao*
Stanford University
wzhao18@stanford.edu

Shutong Zhang*
Stanford University
tonyzst@stanford.edu

1 Project Overview

Trip planning is an important application that helps users in organizing and managing their travel itineraries (Souffriau and Vansteenwegen, 2010). It allows users to specify a series of intended destinations, including the start and end locations, and then generate an optimal route based on specific criteria, such as minimizing travel time or reducing costs. The geographical data utilized in trip planning is generally modeled as a graph, where nodes denote locations (e.g., cities), and edges represent the connections between these locations (e.g., roads). Traditional trip planning models have relied on graph-traversal algorithms (Little et al., 1963; Held and Karp, 1962), which can be computationally intensive and slow due to their iterative nature. One promising alternative is to use graph neural networks (GNNs) (Scarselli et al., 2009; Kipf and Welling, 2017), which have demonstrated efficacy across a wide variety of graph-based tasks. GNNs leverage deep learning to capture intricate relationships within graph structures, offering the potential to predict optimal routes with greater efficiency and precision. In this work, we explore the application of GNNs to enhance trip planning methods.

2 Application Domain

Dataset. The dataset utilized in this project is the OpenStreetMap (OSM) dataset (OpenStreetMap contributors, 2024), an open geographical database that provides detailed global geospatial information, encompassing road networks, buildings, and other infrastructure. The OSM dataset is graph-structured, with nodes representing geographical locations (e.g., landmarks, intersections, points of interest) and edges denoting connections between nodes (e.g., roads, sidewalks, bike lanes). Additionally, both nodes and edges contain feature attributes, such as location coordinates for nodes and path distances for edges. For this project, we limit the scope of the OSM dataset to the geographic area of San Francisco, resulting in graph consists of 9,929 nodes and 27,437 edges.

Problem Statement. While trip planning encompasses a variety of objectives, this project focuses explicitly on the shortest-path trip planning problem. The problem is formulated as follows: given a specified starting point, a destination, and a set of intermediate waypoints, the objective is to determine an optimal sequence of waypoints that forms a travel route that minimizes the total travel distance.

3 Method

Trip Data Generation. Since the OSM dataset does not include predefined travel routes, we generate synthetic subgraph inputs, as illustrated on the left of Figure 1. Each subgraph represents a plausible travel route within a specific region. To construct each subgraph, we randomly selecting K nodes, including a start node, a destination, and a set of intermediate nodes of interest, along with all nodes within the triangular region encompassing the route. We then use a solver (dmishin, 2017) to compute high-quality approximate routes for each subgraph within a fixed time constraint (e.g., 1 minute). These generated subgraphs and approximate routes are subsequently used as training data for the

Preprint. Under review.

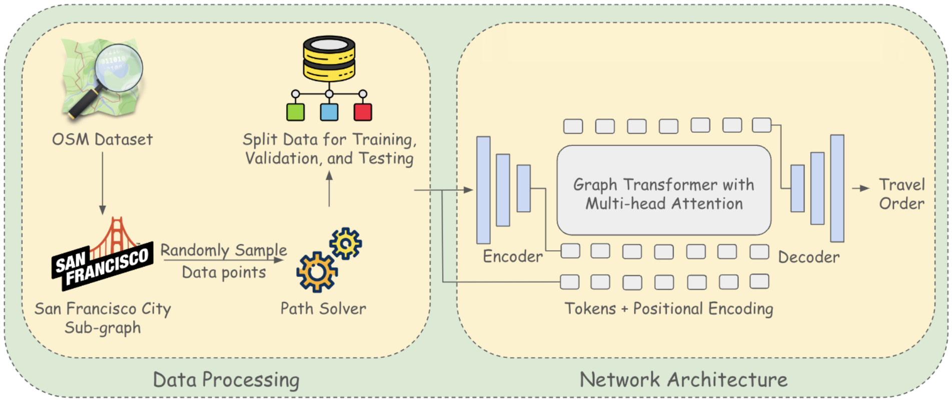

Figure 1: Overview of our trip planning method. The left side illustrates the synthetic trip data generation process used to create training data for the trip planning model. The right side demonstrates how we utilize a GNN to compute the travel order.

development of the trip planning model. We plan to generate 10,000 data points, splitting them into training, validation, and testing sets in an 8/1/1 ratio.

Problem Formulation. We formulate the shortest trip planning problem as an optimization problem on a graph, aiming to determine the optimal ordering of nodes such that the total route traversal cost is minimized. Let $G = (V, E)$ represent the graph, where $V = \{v_1, .., v_N\}$ denotes the set of nodes corresponding to all locations on the map, and $E = \{e_{(u,v)}, \forall u, v \in V\}$ represents the set of edges defining routes between these locations. Given a subset of locations to be visited, denoted by $V' = \{s, t, v'_1, ..., v'_M\} \subseteq V$ where s and t are the specified start and end nodes, the objective is to find the shortest path that covers all required nodes in V' while minimizing total travel cost. The solution is represented as a reordered sequence $\mathcal{S} = (u_0 = s, u_1, u_2, ..., u_M, uM + 1 = t)$, satisfying $set(\mathcal{S}) = V'$. The optimization objective is expressed as

$$min_{\mathcal{S}} \sum_{i=0}^{M} e_{u_i, u_{i+1}} \tag{1}$$

.

Algorithm. The right side of Figure 1 presents an overview of our approach. It utilizes a GNN to generate an embedding vector for each waypoint node $v_i \in V'$ in the input subgraph. These embeddings are then fed into a decoder network that outputs a numerical score for each node, which is used to determine their order. Nodes are sorted based on these scores to establish the final sequence. Once the node order is identified, we apply graph traversal algorithms, such as BFS, to compute the shortest paths between consecutive node pairs, constructing the complete trip.

We employ the Graph Transformer (Yun et al., 2019) as our GNN, which is particularly suited for trip planning as it can effectively capture both local and global relationships within graph-structured data. This capability allows it to identify the most relevant nodes and edges for optimal routing. Specifically, the initial node embeddings are calculated as $h_i^{(0)} = \text{Linear}(x_i)$, where x_i represents the features of node i. Subsequent embeddings are updated according to the rule $h_i^{(l+1)} = \sum_{j=1}^{N} \alpha_{ij} \cdot W_v h_j^{(l)}$, where α_{ij} is the attention score between nodes i and j, defined as:

$$\alpha_{ij} = \frac{\exp\left(\frac{(W_q h_i)^\top (W_k h_j)}{\sqrt{d_k}} + W_e g_{ij}\right)}{\sum_{k=1}^{N} \exp\left(\frac{(W_q h_i)^\top (W_k h_k)}{\sqrt{d_k}} + W_e g_{ik}\right)}, \tag{2}$$

where W_q, W_k, and W_v are learnable projection matrices in the attention mechanism, and g_{ij} represents the edge feature between nodes i and j.

To optimize the network, we compute the loss by the distance between the predicted sequence and the ground truth sequence. We intend to explore Kendall's Tau Distance (Kendall, 1938) and Spearman's Rank Correlation Loss (Spearman, 1904) as potential distance functions, as they provide a robust evaluation of sequence prediction accuracy.

4 Evaluation Methodology

Baselines. We will evaluate our Graph Transformer model against two baselines: (1) The Greedy Algorithm, which selects the nearest location at each step, achieving locally optimal decisions but often failing to find a globally optimal solution; (2) The Traversal-based Approximate Solver, which employs a heuristic approach to approximate the shortest path by exploring potential routes more comprehensively than the Greedy Algorithm, though at the cost of significantly longer processing times. Given that the solver allows for adjustable time constraints, we will evaluate its performance across multiple time levels.

Metrics. We will employ two metrics to evaluate our trip planning model's performance: trip distance and computation time. Trip distance measures the total travel distance of a route generated by the model, providing an indication of how well the model minimizes route length compared to baseline methods. Computation time measures the time taken to generate each route, reflecting the model's efficiency in finding solutions. Together, these metrics will offer a comprehensive evaluation of both the accuracy and computational efficiency of the model.

References

dmishin. tsp-solver: Solving traveling salesman problem using dynamic programming and bitmasking. `https://github.com/dmishin/tsp-solver`, 2017. Accessed: 2024-10-22.

Michael Held and Richard M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial and Applied Mathematics*, 10(1):196–210, 1962. doi: 10.1137/0110015. URL `https://doi.org/10.1137/0110015`.

Maurice G Kendall. A new measure of rank correlation. *Biometrika*, 30(1/2):81–93, 1938.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

John D. C. Little, Katta G. Murty, Dura W. Sweeney, and Caroline Karel. An algorithm for the traveling salesman problem. *Operations Research*, 11(6):972–989, 1963. ISSN 0030364X, 15265463. URL `http://www.jstor.org/stable/167836`.

OpenStreetMap contributors. Openstreetmap: Free World Map. `https://www.openstreetmap.org`, 2024. Accessed: 2024-10-20.

Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. doi: 10.1109/TNN.2008.2005605.

Wouter Souffriau and Pieter Vansteenwegen. Tourist trip planning functionalities: State–of–the–art and future. In Florian Daniel and Federico Michele Facca, editors, *Current Trends in Web Engineering*, pages 474–485, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. ISBN 978-3-642-16985-4.

Charles Spearman. The proof and measurement of association between two things. *The American Journal of Psychology*, 15(1):72–101, 1904.

Seongjun Yun, Minbyul Jeong, Raehyun Kim, Jaewoo Kang, and Hyunwoo J Kim. Graph transformer networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL `https://proceedings.neurips.cc/paper_files/paper/2019/file/9d63484abb477c97640154d40595a3bb-Paper.pdf`.