Review of An Efficient Graph Convolutional Network Technique for the Travelling Salesman Problem

Adib Habbou, Teddy Alexandre

ENS Paris-Saclay

Context

- The Travelling Salesman Problem (TSP) focuses on finding a minimal distance path that visits every node in a given graph
- NP-hard problems, such as TSP, pose challenges for large-scale solutions due to their factorial time complexity
- TSP has widespread applications in areas such as vehicle routing, supply chain, telecommunications, scheduling, and planning
- Various heuristics have historically been employed to address TSP, but recent advancements have introduced novel approaches, including **Graph ConvNet**

Encoding Step

The Graph ConvNet is a Convolutional Neural Network adapted to 2D graphs and decomposed in three main layers :

- **Input Layer :** embeds nodes and edges by mapping nodes linearly and computing edge features based on distance weights through a TSP edge indicator function
- **Graph Convolution Layer :** perform feature extraction in the graph, applying linear combinations, batch normalization, and ReLU activation functions
- **MLP Classifier :** utilizing the edge embedding from the last graph convolution layer to compute probabilities for each edge's inclusion in the TSP tour

Decoding Step

At the end, we obtain a probabilistic heat map over the adjacency matrix of tour connections. We use a search strategy to efficiently retrieve the solution to TSP :

- **Greedy Search :** begins at the first node and iteratively selects the next node based on the highest probability of an edge's presence, masking visited nodes until all nodes are visited
- **Beam Search :** explores the heat map by expanding the b most probable edge connections among neighbors, and iteratively extends the top-b partial tours until all nodes are visited. The final prediction is the tour with the highest probability
- **Beam Search with Shortest Tour Heuristic :** similar to Beam Search, but the final prediction is the shortest tour among the set of b complete tours

Reproduction for 20 nodes

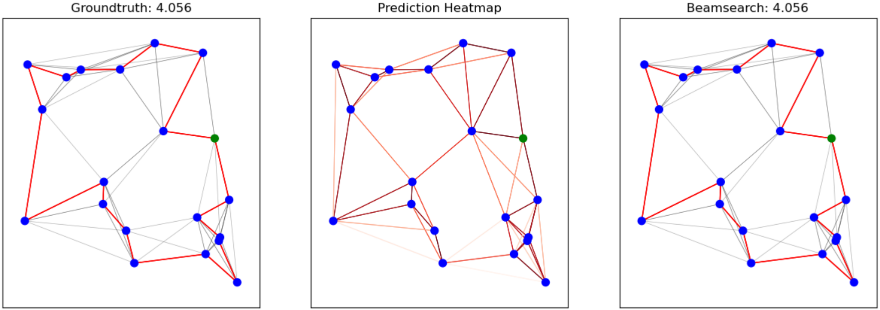

Figure – Reproduction for $N = 20$

Reproduction for 50 nodes

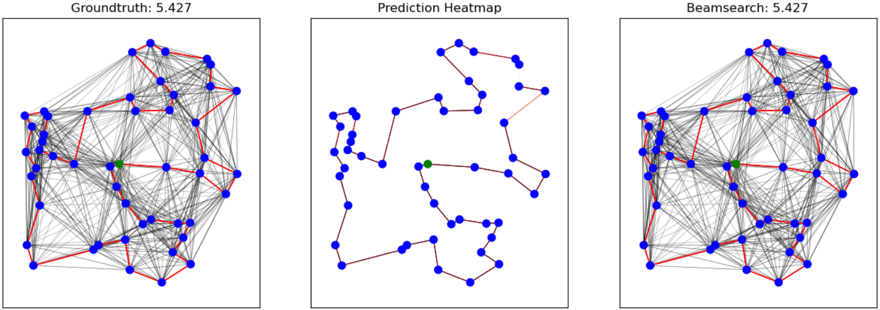

Figure – Reproduction for $N = 50$

Reproduction for 100 nodes

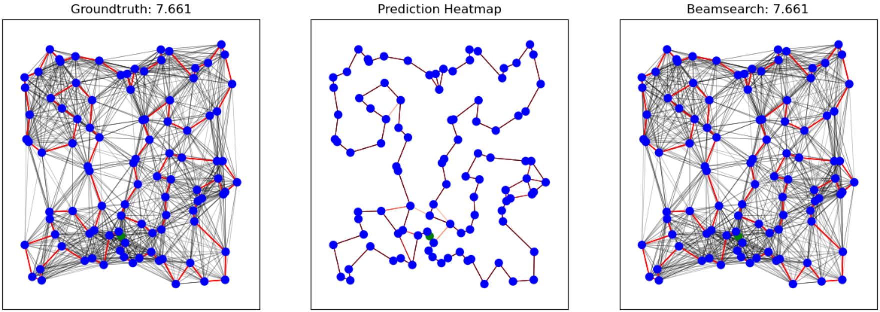

Figure – Reproduction for $N = 100$

Generalization for larger graphs

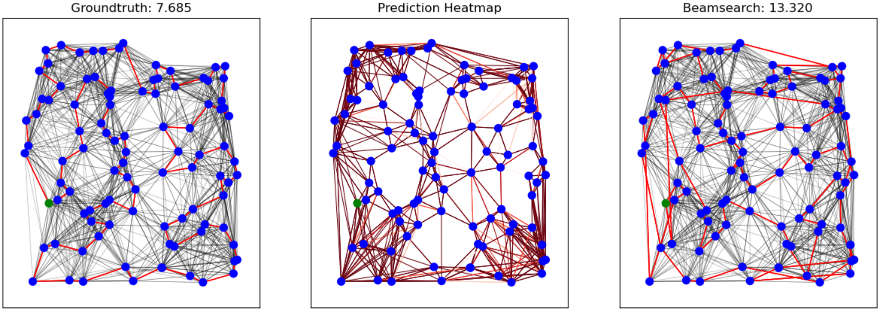

Figure – Training for $N = 20$, testing for $N = 100$

Generalization for smaller graphs

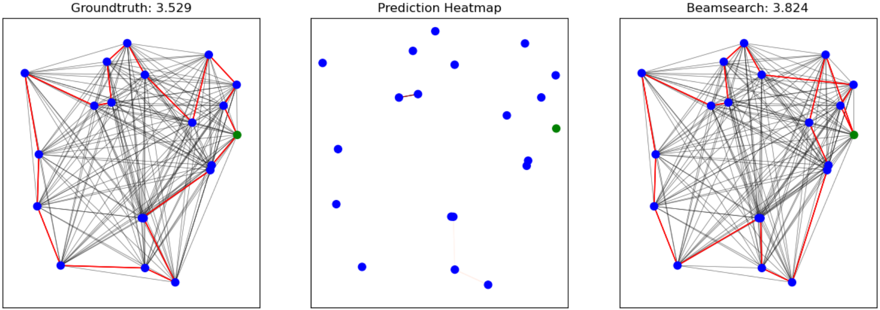

Figure – Training for $N = 100$, testing for $N = 20$

Less number of batches per epoch

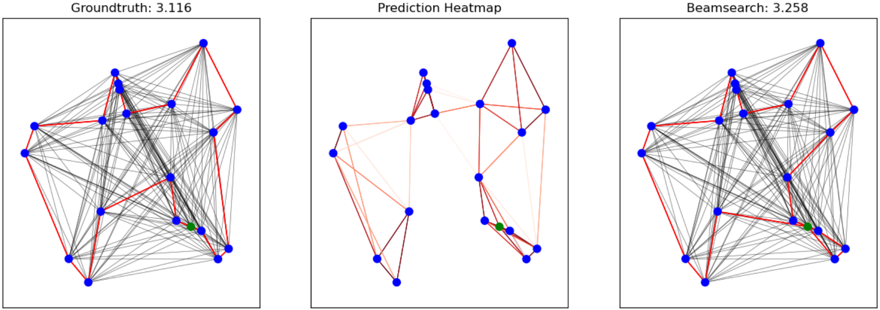

Figure – $N = 20$, 50 batches per epoch, batch size of 20

Less instances in each epoch

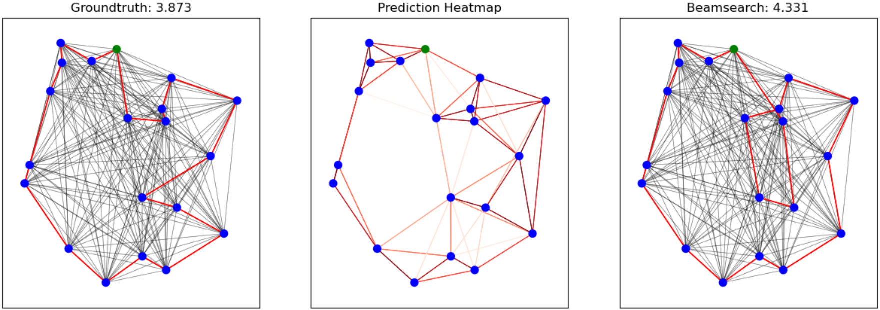

Figure – $N = 20$, 500 batches per epoch, batch size of 2

Limitations

- **When training on smaller graphs :** Graph ConvNet generates a complex and erratic heatmap, hence the prediction is worse, and far from ground-truth
- **When training on larger graphs :** Graph ConvNet generates a sparse heatmap, where all edge are unlikely to figure in the final tour, so the predictions are bad
- **When training on fewer data :** Model performance is sensitive to the size of the train dataset, less data concludes to predictions far from ground-truth

Extensions

- **Leveraging equivariance and symmetries :** Rotations, Reflections, Translations
- **Improved graph search :** Dynamic Programming, Monte-Carlo Tree Search
- **Learning with local search heuristics :** Alternative approach to constructive method such as AR and NAR using TSP heuristics
- **Learning paradigms that promote generalization :** Enhancing fast-adaptation, fine-tuning, multi-task pre-training for other routing problems