What do transformers learn from solving routing problems ?

Roozbeh Sanaei[1], Shahin Gelareh[2,3], Rahimeh N Monemi[4,5]

[1] SUTD - MIT International Design Centre, Singapore
roozbeh.sanaei@gmail.com
[2] Paris School of Business
[3] Universite d'Artois
[4] IESEG Lille, France
[5] Predictim Globe Ltd, Lille, France

Mots-clés : *Routing, Traveling salesman problem, Reinforcement learning, Transformers, Explainable Machine learning, SHAP, LIME*

1 Introduction

Combinatorial optimization problems such as routing, scheduling, planning are being increasingly embedded in a variety of engineering and logistics systems with remarkable momentum in realizing their ultimate performance. As day-to-day problems, objectives, or constraints surface in evolving engineering and logistic systems, crafting hand-engineered heuristic algorithms demand substantial domain knowledge, advanced analytical skills, and considerable development effort, the absence of which may inhibit potential optimality of these systems to materialize. This has inspired a surge of investigations in utilizing deep neural networks for combinatorial optimization problems which made machine learning a driving player in devising innovative solutions for operations research problems of the last decade.

Transformers, grounded entirely on self-attention mechanisms have turned to become the centerstone of this research owing to their prolific performance in image and language synthesis and their prominent status as the de facto model of choice for sequence modeling tasks [1].

Despite the extent of research in putting deep learning into operation in this area and promising gains witnessed through elaborately designed deep structures, there is little consensus as to what sort of knowledge these networks gain over their learning course and to what extent their acquired knowledge is generalizable to instances structurally distinct from their training distribution [2]. Furthermore, whether the current state-of-the-art deep learning machinery is fundamentally apt to meet the intricacies of symbolic problem-solving or complementary components are crucial for achieving this remains unclear amongst the AI community [3] .

This paper will endeavor to get the ball rolling in addressing these questions by resorting to explainable machine-learning tools. We utilize these techniques for explaining the learning of transformers applied to the traveling salesman problem as an exemplary case of routing. To the best of our knowledge, this is the first effort to interpret learnings of deep neural networks used for combinatorial optimization [4]

2 Problem Description

We have grounded our work on the deep learning framework proposed by [5] to directly learn combinatorial tasks where the output is an ordering of the input. Specifically, our focus is to find effective heuristics for solving traveling salesman problems (TSP). TSP consists of finding the shortest possible tour that visits each city only once and returns to the original city. Here we consider 2D Euclidean TSP, where city locations are represented by their 2D coordinates. The aim is to learn the parameters θ of a policy function over city permutations $p_\theta(\pi|s)$, using

Transformer and Policy Gradient. In other words, the transformer is supposed to assign higher probabilities to the order of cities constructing shorter tours.

The transformer architecture follows the conventional encoder-decoder paradigm, where the decoder maps the city locations to the latent space and the encoder predicts the next city index provided this latent space and previous cities in the tour by assigning a higher probability to the order it foresees to produce a shorter tour.

The transformer is trained using the REINFORCE learning rule with a critic to bring down the variance of the gradients. We use the opposite of the tour length as the reward and learn actor parameters θ by starting from a random policy and iteratively optimizing it with the REINFORCE learning rule and Stochastic Gradient Descent (SGD). The critic is computed then by feeding a weighted summation of action vectors into a fully connected neural network followed by a ReLu activation which is trained through minimizing the mean square error between predicted and actual actor rewards on a set of instanced generated on the fly. We benchmarked our results against Google OR Tools [6] on a test set of 5000 euclidean graphs , generated by random drawing of 20 to 50 points from a 2d unit square at uniform distribution.

SHAP (SHapley Additive exPlanations) [4] and LIME (Local interpretable Model-agnostic Explanations) [4] are model-agnostic interpretation methods vastly used in explainable machine learning. Here we propose a variation of LIME and SHAP for routing problems through which we interpret local and global variations of suggested tours. We also employed activation maximization, rollout, and gradient rollout to understand how mid-level features rely on the tour points and how final ordering associates with the mid-level features as well as assessing the impact of local changes on the high-level order of the tour.

3 Conclusions and Perspectives

By and large, based on our observations, transformers do not grow a sense of distance between points but similar clusters of points (or clusters of clusters and so on) appear in the tour in a similar order observed more frequently in the training data. Thus the network may produce highly sub-optimal solutions (compared to the benchmark) when the point set can not be broken down into clusters of points similar to that observed in the training or similar order of clusters does not produce high-quality results. Each layer of the encoder works in principle as a weighted hash table categorizing the clusters of the down-level layer and the other way around for the decoder. Accordingly, going upward higher levels of the encoder and lower levels of the decoder perform at a higher level of abstraction. Output is found robust against input order, possibly owing to the permutation invariance of the self-attention operator and local changes in city locations exhibiting little impact on the high-level order of the tour.

Références

[1] Vesselinova, Natalia, et al. *Learning combinatorial optimization on graphs : A survey with applicathe tions to networking.* IEEE Access 8 (2020) : 120388-120416.

[2] Marcus, Gary. *Deep learning : A critical appraisal.* arXiv preprint arXiv :1801.00631 (2018).

[3] Garcez, Artur d'Avila, and Luis C. Lamb. *Neurosymbolic AI : the 3rd wave.* arXiv preprint arXiv :2012.05876 (2020).

[4] Belle, Vaishak, and Ioannis Papantonis. *Principles and practice of explainable machine learning.* Frontiers in big Data (2021) : 39.

[5] Deudon, Michel, et al. *Learning heuristics for the tsp by policy gradient.* International conference on the integration of constraint programming, artificial intelligence, and operations research. Springer, Cham, 2018.

[6] Perron, Laurent. *Operations research and constraint programming at google.* International Conference on Principles and Practice of Constraint Programming. Springer, Berlin, Heidelberg, 2011.