https://pubsonline.informs.org/journal/trsc

TRANSPORTATION SCIENCE

Vol. 00, No. 0, Xxxxx 0000, pp. 000–000

Submitted to *Transportation Science*

ISSN 0041-1655, EISSN 1526-5447

Reduce-then-Optimize for the Fixed-Charge Transportation Problem – Appendix

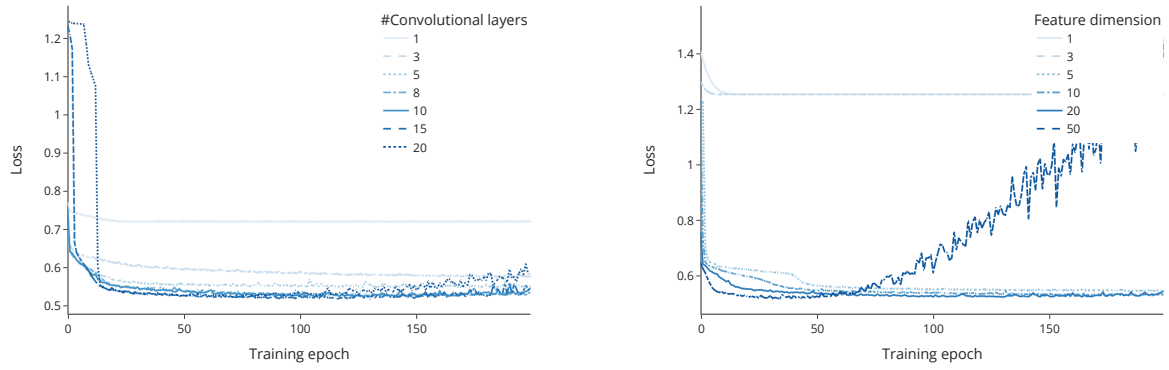
Caroline Spieckermann
Logistics and Supply Chain Management, TUM School of Management, Technical University of Munich, Munich, Germany
Advanced Optimization in a Networked Economy (GRK 2201), Technical University of Munich, Munich, Germany
caroline.spieckermann@tum.de

Stefan Minner
Logistics and Supply Chain Management, TUM School of Management, Technical University of Munich, Munich, Germany
Munich Data Science Institute (MDSI), Technical University of Munich, Munich, Germany
stefan.minner@tum.de

Maximilian Schiffer
Business Analytics and Intelligent Systems, TUM School of Management, Technical University of Munich, Munich, Germany
Munich Data Science Institute (MDSI), Technical University of Munich, Munich, Germany
schiffer@tum.de

Appendix A: Hyperparameter Tuning and Training Data Set Size

Figure 1 shows the validation loss of the GNN for different model hyperparameters. All models were trained on small BASE instances. To better assess the risk for overfitting, the models were trained for a fixed number of epochs without learning rate decay and early stopping. The results show that the model performance improves with an increasing number of layers and features, but when the number of parameters becomes too large, the GNN might overfit. As a good compromise between under- and overfitting, we select the number of convolutional layers to be 10 and the feature



(a) Number of convolutional layers (b) Feature dimensionality

Figure 1 Hyperparameter screening (15×15, base configuration). The default values are #convolutional layers=10, #dense layers=2, feature dimension=20.

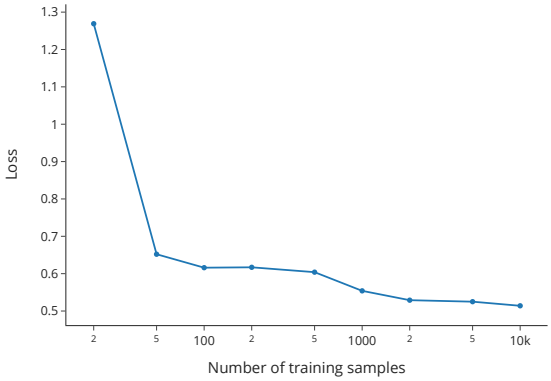


Figure 2 Validation loss for different training data set sizes (BASE_15x15)

dimensionality to be 20. We find that the additional dense layers after the convolutional block hardly affect model performance and use only two dense layers before the output layer.

To generate neither too little nor too much training data, we further evaluate the GNN performance for different training data set sizes (Figure 2). The results show that the performance starts stagnating at a training data set size of 2000 instances and we use a training data set of 4000 instances for all models and data sets.

Appendix B: Details on Prediction Performance of Different ML Models

Table 1 compares the different ML models (GNN, MLP, LR) in terms of validation loss, accuracy, recall, precision and F_2-score. The GNN dominates the baseline models in all metrics.

Table 1 Comparison of different ML models in terms of prediction performance

	Loss	Accuracy [%]	Recall [%]	Precision [%]	F_2 [%]
LR - basic	0.762	76.88	85.53	27.49	60.14
LR - adv.	0.733	77.39	86.89	28.16	61.32
LR - adv. + stat.	0.711	78.31	87.04	29.08	62.23
MLP - basic	0.720	77.45	87.15	28.26	61.51
MLP - adv.	0.705	77.68	87.98	28.59	62.16
MLP - adv. + stat.	0.694	77.90	88.17	28.83	62.46
GNN	0.534	83.85	91.04	36.46	70.07
GNN - adv.	0.532	84.00	91.19	36.69	70.30
GNN - adv. + stat.	0.534	83.92	90.90	36.55	70.06

Appendix C: Feasibility Edges

To ensure feasibility of the reduced problem, we generate a greedy solution and add the solution edges to the edge set. Figure 3 compares the size of the predicted edge set to the size of the edge set including the additional feasibility edges. The results show that the feasibility edges increase the edge sets only marginally.

Table 2 reports the number of instances that need the additional edges to be feasible and the frequency with which feasibility edges are used in the final solution. Only for the smallest edge set (20%), two instances rely on the feasibility

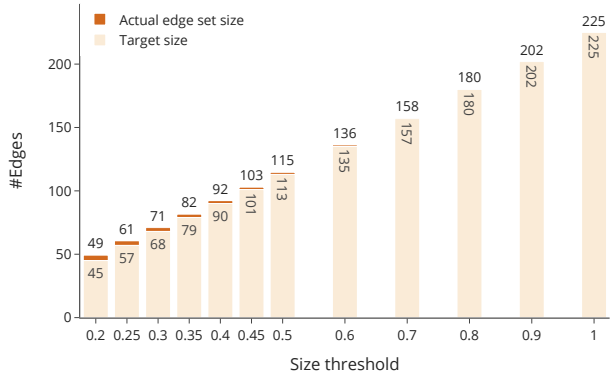


Figure 3 Share of feasibility edges in edge set for different edge set size thresholds (BASE_15x15)

edges, and for edge set sizes above 30%, the additional edges are never used for the optimal solution of the reduced problem, thus only serving as a feasibility guarantee and not contributing significantly to the solution quality.

Table 2 Number of instances that need feasibility edges and share of feasibility edges in optimal solutions for different edge set size thresholds (BASE_15x15)

	0.20	0.25	0.30	0.35	0.40	0.45	0.50	0.60	0.70	0.80	0.90	1.00
#instances needing feasibility edges	2	0	0	0	0	0	0	0	0	0	0	0
#instances using feasibility edges	13	5	1	0	0	0	0	0	0	0	0	0
#feasibility edges used (avg.)	0.53	0.17	0.03	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

Appendix D: Maximum Likelihood Solutions for Capacitated FCTPs and FCTPs with Blending

The greedy maximum likelihood procedure described in Algorithm 1 in Section 3.2 might fail in case of edge capacities or blending constraints. Therefore, we solve a helper MILP based on the FCTP model formulation presented in Section 2.1 that minimizes the total inverse prediction value. Let $\hat{y}_{ij}$ be the prediction for edge (i, j). Then a feasible solution with high edge probabilities will be obtained by solving the following helper problem:

$$\min_{\boldsymbol{x},\boldsymbol{y}} \sum_{i \in N_S} \sum_{j \in N_D} \frac{y_{ij}}{\hat{y}_{ij}}, \quad \text{s.t. problem constraints (2) - (6)} \ .$$

This again is an FCTP, but in practice it solves significantly faster than the original problem. Alternatively, one can solve the helper problem heuristically. Based on the results in Appendix C, the effect on the overall performance can be expected to be low.

Appendix E: Generalization across Network Structures

To analyze the generalization capabilities across network structures, we evaluate a GNN that was trained on symmetric instances (BASE_15x15) on instances of similar size but with an asymmetric network structure. Figure 4 summarizes the pipeline performance in terms of optimality gaps and runtimes on BASE_10x20 and BASE_20x10 instances. In both cases, the optimality gaps and relative runtime savings are similar to those of the symmetric instances. Note that

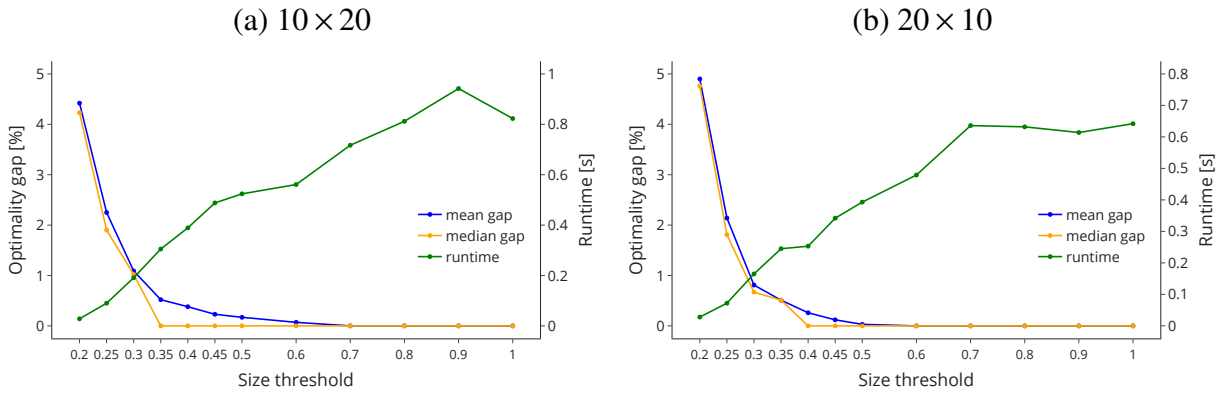


Figure 4 **Optimality gaps and runtimes when evaluating GNN+GRB-pipeline on instances with a different network structure. The GNN was trained on BASE_15x15 instances and evaluated on BASE_10x20 and BASE_20x10 instances.**

absolute runtimes are smaller for the asymmetric instances than for the symmetric instances, which can be explained by the reduced value range that was used for sampling demand and supply quantities when the number of demand nodes exceeded the number of supply nodes and vice versa. The effect of the value range on absolute runtimes is discussed in Section 5.2.4.

Appendix F: Generalization to Smaller Instances

Although generalization to smaller instances is less relevant since they solve quickly, we also test the GNN, trained on BASE_15x15 instances, on BASE_10x10 instances to better assess its robustness across different instance sizes. Figure 5 summarizes the pipeline performance in terms of optimality gaps and runtimes, demonstrating a good balance between solution quality and runtime for edge set sizes around 40%.

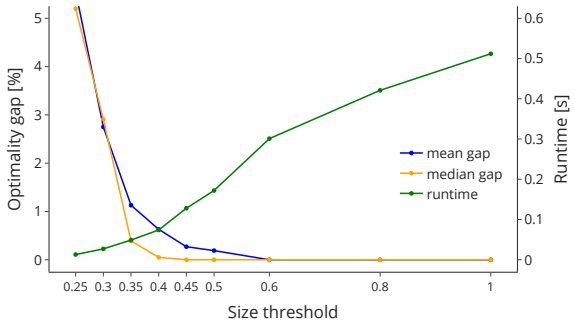


Figure 5 **Optimality gaps and runtimes when evaluating GNN+GRB-pipeline on smaller instances. The GNN was trained on BASE_15x15 instances and evaluated on BASE_10x10 instances.**