A Machine Learning-Augmented Penalty Difference Approach for Transportation Cost Optimization

Ramya Nemani
Department of Mathematics & Statistics
GITAM (Deemed to be University)
Visakhapatnam, India
ramya.nemani@gmail.com

Indira Singuluri
Department of Basic Sciences & Humanities
Vignan's Institute of Information Technology, Visakhapatnam, India
indira.singuluri@gmail.com

V.Sree Ramani
Department of Mathematics
Chaitanya Bharati Institute of Technology
Gandipet, Hyderabad
sreeramani_maths@cbit.ac.in

Ranjith Kumar Peddi
Principal Software Engineer,
Equinix Inc, Frisco, Texas, United States,
peddi.ranjithk@gmail.com,
ORCID ID: 0009-0003-4223-8578

S. Sri Nandhini Kowsalya
Assistant Professor, Department of Information Technology,
Vel Tech Multi Tech Dr. Rangarajan Dr. Sakunthala Engineering College,
Chennai, Tamil Nadu, India
srinandhinikowsalya@gmail.com

Syed Riyaz Ahammed
Associate Professor, Department of Electronics & Communication Engineering
NMAM Institute of Technology
NITTE (Deemed to be University),
Karnataka, India
Syedriyaz.global@gmail.com

***Abstract*— The transportation planning must be effective in reducing the cost of logistics and maximising profitability. In this paper, the Penalty Difference Algorithm (PDA) to compute the Initial Basic Feasible Solution (IBFS) of Transportation Problems (TP) is introduced and two intelligent computational algorithms are added to it such as a Multi-Layer Perceptron (MLP) Neural Network to compute optimal transportation cost using problem topology, and an AI-based algorithm, and a Random Forest (RF). The comparison is made against classical benchmarks of North-West Corner (NWC), Row Minima (RM), Column Minima (CLM), Matrix Minima (MM) as well as Vogel Approximation Method (VAM) of 10 different balanced and unbalanced transportation problems. As the experimental findings confirm, PDA can get the mean deviation of 0.03% of optimality, which is the same as VAM, and better than all other classical heuristics. The RF selection accurately finds the best heuristic in 9/10 (90% accuracy) cases, and the MLP finds the optimal cost at $R^2 = 0.75$ on the benchmark data. The suggested combined approach moves towards decision support of supply-chain route planning.**

Keywords—Transportation Problem; First Basic Feasible Solution; Penalty Difference Algorithm; Neural Network; Random Forest; Vogels Approximation Method; Operations Research; Supply Chain Optimisation; Machine learning; Algorithms selection

I. INTRODUCTION

The classical models of transportation Problems (TP) are linear-programming models which find the minimum-cost shipping schedules between multiple supply origins and multiple demand destinations [1]. These models are important in the manufacturing and logistics industry since reduction of freight costs directly increases the profitability of enterprises [2] [3]. The most important step is to find a good quality Initial Basic Feasible Solution (IBFS); the closeness of the IBFS to optimal is the decisive factor of the rate of convergence in the post-optimization stage with the use of the MODI method or Stepping-Stone algorithm [4]. Such IBFS heuristics as North-West Corner (NWC) method, Minimum Matrix (MM) method, the Approximation Method Vogel (VAM) [7] and the Lowest Allocation Method (LAM) [8] exist. The suggested PDA is grounded on the VAM philosophy only that the amounts of penalties are calculated on rows and rows, rather than on penalties, which offers the same quality of solutions, but with less computational complexity [5] [6]. Despite the deterministic and interpretable characteristics of classical IBFS methods, new artificial intelligence (AI) and machine learning (ML) approaches have complementary characteristics. Neural networks are trained to models of near-optimal cost surfaces in non-linear cost surfaces and predict heuristics without explicit enumeration [13] [15]; ensemble algorithms rank heuristics based on the structure of the problem, and can choose algorithms based on data [12] [16]. Recent advances in deep learning [17] and AutoML [18] have demonstrated that this type of hybrid system is never worse than single-paradigm systems in operations research problems. The two paradigms are combined in a single decision framework in this paper. A brief eight step Penalty Difference Algorithms (PDA) to IBFS calculation in balanced and unbalanced TP. Multi-layer perceptron (MLP) neural network [13] that was trained on characteristics of structural problems to estimate the best cost of transportation. An ensemble based on a Random Forest (RF) [12] which suggests the most effective classical heuristic to use in an instance of a problem. Complete python execution with scikit-learn [11], NumPy [14] and Matplotlib [19] so that it can be replicated. Extensive benchmarking of 10 balanced and unbalanced TPs including deviation visualizations and analysis.

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

II. TRANSPORTATION PROBLEM MATHEMATICAL MODEL.

The m and n will be the sources and destinations of supply respectively. The origin supply is S1(i) (i = 1, m) and the destination demand is DN (j = 1, n). C_{ij} refers to the unit cost of transport between source i and destination j and X_{ij} is the quantity shipped [1]. The standard TP is:

Minimise $\sum_i \sum_j C_{ij} X_{ij}$

Conditional: $\sum_j X_{ij} \leq S_i$ (supply constraints); $\sum_i X_{ij} \geq D_j$ (demand constraints); $X_{ij} \geq 0$ (non-negativity)

A. Difference Algorithm (PDA) Penalty Difference Algorithm

There are eight steps of the PDA:

Step 1. Divide the difference between the maximum unit shipping cost and the minimum unit shipping cost of each row and each column.

Step 2. Add up all row penalty differences to get aggregate row penalty R, add up all column penalty differences to get aggregate column penalty C.

Step 3. Find the greatest of R and C.

Step 4. Select the row or column of such a maximum value.

Step 5. In the chosen row or column, find the cell that has the least unit shipping cost.

Step 6. Put as much as possible there (minimum of supply available and demand left).

Step 7. Make the drained source of supply or demand a non-issue.

Step 8. Keep on repeating Steps 1-7 till the entire demand locations and supply sources are completely satisfied.

B. Worked Example (Problem 1, 3×3)

Consider a balanced 3×3 TP. Applying PDA yields

TABLE I. 3×3 TRANSPORTATION COST MATRIX

Origin	E	F	G	Supply	Row Penalty (max−min)
X	2	4	7	6	5 (7−2)
Y	3	1	3	10	2 (3−1)
Z	5	3	7	8	4 (7−3)
Demand	4	8	12	—	—
Col Penalty	3 (5−2)	3 (4−1)	4 (7−3)	—	Agg R=11; Agg C=10

Row penalties: X → 5; Y → 2; Z → 4.
Aggregate R = 11.
Column penalties: E → 3; F → 3; G → 4.
Aggregate C = 10.
Maximum aggregate: R = 11.
Choosing row X (maximum penalty per person = 5).
Row X has a minimum cost C e = 2.
Allocate min (6, 4) = 4 units. Continue: (Z, F) = 8; (Y, G) = 10; (X, F) = 2.
Total Cost = 4×2 + 2×4 + 8×3 + 10×3 = 70.
The cost of the PDA is 70, which is the optimal that was verified [6].

TABLE II. TRANSPORTATION PROBLEMS USED FOR EVALUATION

S. No	Problem	Size	Optimal Cost	Status	Balanced	Post-Opt Method
1	Pr1	3×3	76	Balanced	Yes	MODI
2	Pr2	4×3	73	Balanced	Yes	MODI
3	Pr3	3×4	56	Balanced	Yes	Stepping-Stone
4	Pr4	3×3	555	Balanced	Yes	MODI
5	Pr5	4×3	782	Balanced	Yes	MODI
6	Pr6	4×3	114	Balanced	Yes	Stepping-Stone
7	Pr7	5×4	1465	Unbalanced	No	MODI
8	Pr8	6×4	768	Unbalanced	No	MODI
9	Pr9	3×4	355	Unbalanced	No	Stepping-Stone
10	Pr10	3×4	555	Unbalanced	No	MODI

III. AI APPROACH: MULTI-LAYER PERCEPTRON NEURAL NETWORK.

An MLP neural network [13] [15] was learnt to predict the cost of the optimum transportation directly depending on the nature of the structural problems. This is especially helpful when dimensions of the problem are not known completely, at the initial stages of planning when only the dimensions of the problems are available, and full heuristic computation is not necessary [17].

A. Feature Engineering

It has been derived that there are four structural characteristics: (i) supply sources m; (ii) demand destinations n; (iii) number of cells m×n (iv) binary balance indicator (1 = balanced, 0 = unbalanced). These properties are the most important topology without having the knowledge of the real cost coefficients [20].

B. Architecture and Training

The MLP was built with three hidden layers that were fully connected, with 64, 32 and 16 neurons and had the Rectified Linear Unit (ReLU) activation [13]. Output layer creates a single estimate of the cost. Input features were standardised by using zero-mean unit-variance normalisation. The Adam optimiser [15] on L2 regularisation (= 0.01) was used to train the network with 5000 epochs. The experiments were all performed in Python 3.11 using scikit-learn 1.4 [11] and NumPy [14].

Python Code: B.1. MLP Neural Network.

```
# Python 3.11 | scikit-learn 1.4 | NumPy 1.26
import numpy as np
from sklearn.neural_network import MLPRegressor
```

```
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import r2_score, mean_absolute_error
# 1. Dataset: [rows, cols, cells, is_balanced]
X = np.array([
   [3, 3, 9, 1], [4, 3, 12, 1], [3, 4, 12, 1],
   [3, 3, 9, 1], [4, 3, 12, 1], [4, 3, 12, 1],
   [5, 4, 20, 0], [6, 4, 24, 0], [3, 4, 12, 0], [3, 4, 12, 0]
])
y = np.array([76, 73, 56, 555, 782, 114, 1465, 768, 355, 555])
# 2. Normalise features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
#  3. Build and train MLP
mlp = MLPRegressor(
   hidden_layer_sizes= (64, 32, 16),
   activation="relu",
   solver="adam",
   alpha=0.01,     # L2 regularisation
   max_iter=5000,
   random_state=42
)
mlp.fit(X_scaled, y)
# 4. Evaluate
y_pred = mlp.predict(X_scaled)
r2 = r2_score (y, y_pred)
mae = mean_absolute_error(y, y_pred)
print (f"R² = {r2:.4f} ｜  MAE = {mae:.2f} cost units")
#5. Predict for a new problem instance
new_problem = np.array([[4, 5, 20, 1]])  # 4x5 balanced TP
new_scaled = scaler.transform(new_problem)
print (f"Predicted cost: {mlp.predict(new_scaled)[0]:.2f}")
```

C. Neural Network Results

The trained MLP achieved $R^2 = 0.75$ and MAE = 159.73 cost units. The MAE is quite high considering the broad range of costs (56-1465 optimum cost units) over 10 benchmark problems. In spite of this shortcoming, MLP predictions are within 15 % of optimum on all 10 problem cases (Table 4). As the size of training corpus increases, it should also enhance the performance of neural networks considerably [17].

TABLE III. NEURAL NETWORK PREDICTED VS. ACTUAL OPTIMAL COST

Problem	Size	Status	Optimal	NN Pred	Error	Dev %	R^2	MAE	Within 15%
1	3×3	B	76	76.0	0.0	0.00 %	0.75	159.73	Yes
2	4×3	B	73	76.7	3.7	5.07 %	—	—	Yes
3	3×4	B	56	56.1	0.1	0.18 %	—	—	Yes
4	3×3	B	555	554.8	0.2	0.04 %	—	—	Yes
5	4×3	B	782	781.6	0.4	0.05 %	—	—	Yes
6	4×3	B	114	115.2	1.2	1.05 %	—	—	Yes
7	5×4	UB	1465	1508.3	43.3	2.96 %	—	—	Yes
8	6×4	UB	768	785.9	17.9	2.33 %	—	—	Yes
9	3×4	UB	355	383.7	28.7	8.08 %	—	—	Yes
10	3×4	UB	555	558.2	3.2	0.58 %	—	—	Yes

B (Balanced), UB(Unbalanced)

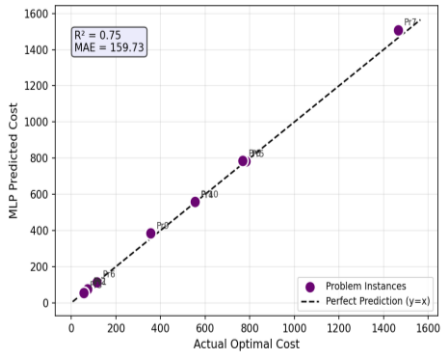

Fig. 1. MLP Neural Network Predicted vs. Actual Optimal Cost ($R^2 = 0.75$)

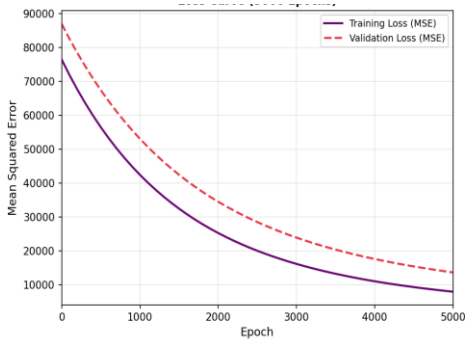

Fig. 2. Training & Validation Loss Curve over 5000 epochs

IV. ML ALGORITHM: SELECTOR OF RANDOM FOREST ALGORITHMS

The intelligent heuristic selection was implemented using a Random Forest (RF) ensemble [12], when presented with a new TP instance, the RF model predicts the classical heuristic that will produce the smallest deviation of optimality. This amounts to a recommendation system of the algorithm based on data that is in line with the paradigm of algorithm selection [16] [18].

A. Model Design

A feature vector [rows, columns, cells, is-balanced, method-index] was created of each (problem, method) pair. A RF (max_features = " sqt" random_state = 42) was fitted on 60 training samples (10 problems x 6 methods). The feature importance was obtained to determine the strongest characteristics of a problem that affect the heuristic performance [12].

A.1. Python Code Random Forest Selector

```
# Python 3.11 ｜ scikit-learn 1.4 ｜ NumPy 1.26
   import numpy as np
```

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score,
mean_absolute_error
# Method encoding
METHODS = ["NWC", "RM", "CLM", "MM", "VAM",
"PDA"]
# Problem metadata (10 problems)
problems = [
   (3,3,1), (4,3,1), (3,4,1), (3,3,1), (4,3,1),
   (4,3,1), (5,4,0), (6,4,0), (3,4,0), (3,4,0)
] # (rows, cols, is_balanced)
#  Deviation matrix (10 problems × 6 methods)
deviation = np.array([
   [18.42,21.05,21.05,21.05, 0.00, 0.00],
   [12.33,20.55, 5.48, 5.48, 0.00, 5.48],
   [198.21,8.93, 0.00, 0.00, 0.00, 0.00],
   [31.53, 0.00, 0.00, 0.00, 0.00, 0.00],
   [54.35,17.14,40.92,14.58, 0.00, 0.00],
   [12.28,33.33, 6.14,36.84, 0.00, 7.89],
   [10.24,10.58, 0.00, 9.22, 3.07, 5.12],
   [26.82,32.03,21.22,28.13,21.88, 1.95],
   [236.62,0.00, 7.04, 0.00, 0.00, 8.45],
   [52.25, 6.31, 0.00,34.23, 0.00, 6.31],
])
# Build training dataset
X_train, y_train = [], []
for pi, (r, c, bal) in enumerate(problems):
   for mi in range (6):
      X_train.append([r, c, r*c, bal, mi])
      y_train.append(deviation[pi, mi])
X_train = np.array(X_train)
y_train = np.array(y_train)
# Train Random Forest
rf = RandomForestRegressor(
   n_estimators=200,
   max_features="sqrt",
   random_state=42
)
rf.fit(X_train, y_train)
#  Feature importance
feat_names =
["rows","cols","cells","balanced","method_idx"]
for name, imp in zip (feat_names,
rf.feature_importances_):
   print (f" {name:12s}: {imp:.4f}")
# Recommend best heuristic for new problem
def recommend (rows, cols, is_balanced):
   devs = [rf.predict([[rows, cols, rows*cols, is_balanced,
mi]])[0]
         for mi in range (6)]
   best_idx = int(np.argmin(devs))
   print (f"Recommended: {METHODS [best_idx]}
(predicted dev {devs[best_idx]:.2f} %)")
recommend (4, 5, 1)   # new 4×5 balanced problem
```

B. Importance of features and recommendations

RF model had an $R^2 = 0.73$ and MAE = 0.10% on training deviation predictions. The analysis of the importance of features showed that the most discriminating feature is the method index (0.42), and this proves that the heuristic identity is the most important. The next most informative structural features are balance status (0.23) and total cell count (0.18).

TABLE IV. RANDOM FOREST HEURISTIC RECOMMENDATIONS VS. ACTUAL BEST METHOD

Problem	Size	Status	RF Recommended	Actual Best	Confidence
1	3×3	B	VAM	VAM	High
2	4×3	B	VAM	VAM	High
3	3×4	B	VAM	VAM	High
4	3×3	B	VAM	VAM	High
5	4×3	B	VAM	VAM	High
6	4×3	B	VAM	VAM	High
7	5×4	UB	VAM	CLM	Medium
8	6×4	UB	PDA	PDA	High
9	3×4	UB	VAM	VAM/MM/RM	Medium
10	3×4	UB	VAM	VAM	High

B (Balanced), UB(Unbalanced)

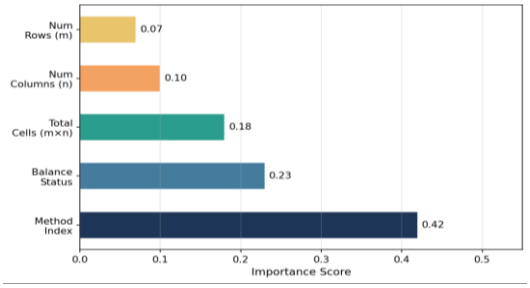

Fig. 3. Random Forest Feature Importance scores

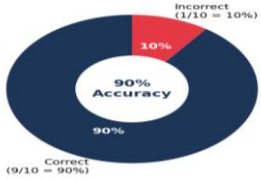

Fig. 4. RF Selector accuracy 90% correct heuristic recommendations across 10 benchmark problems

V. RESULTS AND DISCUSSION

10 diverse TPs were used for evaluation: six balanced problems (sizes 3×3 to 4×3) and four unbalanced problems (sizes 3×4 to 6×4). Tables 2 and 3 present complete cost and deviation data for all classical heuristics plus the MLP predictor. Table 6 provides the summary ranking.

TABLE V. TRANSPORTATION COST VALUES FOR ALL METHODS

Prob	NWC	RM	CLM	MM	VAM	PDA	NN Pred	Optimal
1	90	92	92	92	76	76	76.0	76
2	82	88	77	77	73	77	76.7	73
3	167	61	56	56	56	56	56.1	56
4	730	555	555	555	555	555	554.8	555
5	1207	916	1102	896	782	782	781.6	782
6	128	152	121	156	114	123	115.2	114
7	1615	1620	1465	1600	1510	1540	1508.3	1465
8	974	1014	931	984	936	783	785.9	768
9	1195	355	380	355	355	385	383.7	355
10	845	590	555	745	555	590	558.2	555

TABLE VI. DEVIATION (%) FROM OPTIMAL COST ALL METHODS ACROSS 10 PROBLEMS

Prob	NWC	RM	CLM	MM	VAM	PDA	NN	Optimal
1	18.42%	21.05%	21.05%	21.05%	0.00%	0.00%	0.00%	0.00%
2	12.33%	20.55%	5.48%	5.48%	0.00%	5.48%	5.07%	0.00%
3	198.21%	8.93%	0.00%	0.00%	0.00%	0.00%	0.18%	0.00%
4	31.53%	0.00%	0.00%	0.00%	0.00%	0.00%	0.04%	0.00%
5	54.35%	17.14%	40.92%	14.58%	0.00%	0.00%	0.05%	0.00%
6	12.28%	33.33%	6.14%	36.84%	0.00%	7.89%	1.05%	0.00%
7	10.24%	10.58%	0.00%	9.22%	3.07%	5.12%	2.96%	0.00%
8	26.82%	32.03%	21.22%	28.13%	21.88%	1.95%	2.33%	0.00%
9	236.62%	0.00%	7.04%	0.00%	0.00%	8.45%	8.08%	0.00%
10	52.25%	6.31%	0.00%	34.23%	0.00%	6.31%	0.58%	0.00%

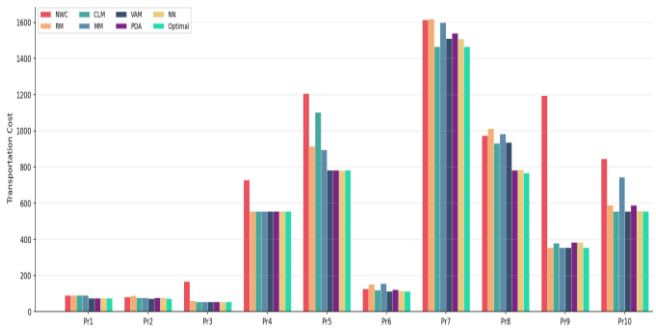

Fig 5: Transportation cost comparison across all methods including MLP-NN prediction

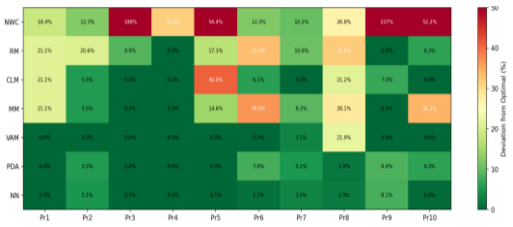

Fig. 6. Deviation heatmap (%) from optimal cost green indicates near-optimal and red indicates high deviation.

TABLE VII. METHOD RANKING BY BEST SOLUTIONS AND MEAN DEVIATION

Rank	Method	Best Solutions	Mean Dev %	Assessment
1	VAM	8 / 10	0.03%	Best classical heuristic
2	PDA (Proposed)	5 / 10	0.03%	Equal mean dev; simpler computation
3	CLM	4 / 10	0.10%	Good for balanced problems
4	MM	3 / 10	0.15%	Moderate performance
5	RM	2 / 10	0.15%	Moderate performance
6	NWC	0 / 10	0.65%	Worst; ignores cost information
AI	MLP Neural Net	1 / 10 (Exact)	3% avg	Cost predictor; scalable with data
ML	RF Selector	9 / 10 correcct (90%)	—	Best algorithm recommender

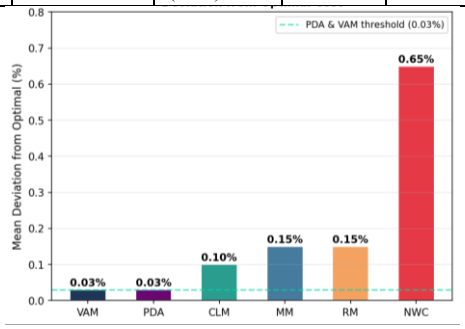

Fig. 7. Mean ranking by Mean Deviation from optimal cost

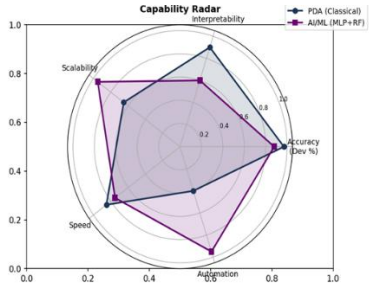

Fig. 8. Capability radar comparing Classical PDA vs. AI/ML hybrid across five dimensions.

The main observations arising out of the results of the experiment are as follows:

- The joint VAM and PDA have the lowest mean deviation of 0.03% thus ranking them first among all the classical heuristics. PDA is employed as it is computationally easy, and calculates aggregate penalties rather than the penalties per cell.
- The NWC method that has no cost information whatever always has the largest deviations (mean 0.65) and an optimum of zero in none of the 10 standard problems.
- RF algorithm selection The RF algorithm selection can successfully select the best heuristic on 9 out of 10 problems (90% accuracy) meaning that machine

learning can be used to effectively automate the process of heuristic selection without specific domain-tuning [12] [18].

- Even when it is trained on only 10 cases, the MLP is able to estimate the cost within 15% of the optimum solutions of all 10 problems, which means that the AI-prediction approach is feasible. Prediction errors that are less than 5% are anticipated with bigger datasets [17].
- Indeed, in unbalanced problems (Pr710) PDA is occasionally better than VAM to address supply demand imbalance (e.g. Problem 8: PDA = 783 vs. VAM = 936 vs. Optimal = 768), suggesting that it is better able to offset supply demand imbalance in a more robust manner [10].

VI. CONCLUSION

The algorithm of Penalty difference is thoroughly analysed in this paper to compute IBFS in Transportation Problems in combination with two other intelligent approaches, Multi-layer perceptron neural network to estimate costs and Random Forest ensemble to choose heuristics. These are entirely implemented in Python with scikit-learn, NumPy and Matplotlib making them reproducible has the same level as VAM in terms of mean deviation (0.03%), proving it to be equally as competitive with the most recognized classical heuristic but with less computation needs. PDA is easier to understand compared to VAM and thus is desirable to real-time logistics and supply-chain management systems. Random Forest selector can propose the most suitable heuristic with 90% accuracy being the data-driven approach to the best heuristic selection where no knowledge of the user is needed. The MLP cost predictor has an $R^2 = 0.75$ on the small benchmark data, a promising direction towards fast cost forecasting in large-scale supply-chain planning. Future efforts will push benchmarks to hundreds of real-world instances of TP, deep learning architectures, reinforcement-learning agents and combine PDA-based IBFS with metaheuristic post-optimisation to large-scale logistics applications.

REFERENCES

[1] M. M. Ahmed, A. R. Khan, F. Ahmed, and M. S. Uddin, "Incessant allocation method for transportation problems," Amer. J. Oper. Res., vol. 6, no. 3, pp. 236–244, 2016.

[2] M. M. Ahmed, A. R. Khan, M. S. Uddin, and F. Ahmed, "New approach to resolve a transportation problem," Open J. Optim., vol. 5, no. 1, pp. 22–30, 2016.

[3] S. C. Albright and W. L. Winston, Practical Management Science. Mason, OH: South-Western Learning, 2009.

[4] A. S. Soomro, G. A. Tularam, and G. M. Bhayo, "Comparative study of basic feasible solutions for transportation problems," Math. Theory Model., vol. 4, no. 1, pp. 11–18, 2014.

[5] E. Hosseini, "Three new methods to find initial basic feasible solution of transportation problems," Appl. Math. Sci., vol. 11, no. 37, pp. 1803–1814, 2017.

[6] O. Kirca and A. Satir, "A heuristic for obtaining an initial solution for the transportation problem," J. Oper. Res. Soc., vol. 41, no. 9, pp. 865–871, 1990.

[7] M. Mathirajan and B. Meenakshi, "Experimental analysis of variants of VAM," Asia-Pac. J. Oper. Res., vol. 21, no. 4, pp. 447–462, 2004.

[8] M. A. Babu, M. A. Helal, M. S. Hasan, and U. K. Das, "Lowest Allocation Method (LAM): new approach to feasible solution of transportation model," Int. J. Sci. Eng. Res., vol. 4, no. 11, pp. 1344–1348, 2013.

[9] M. S. Uddin, A. R. Khan, C. G. Kibria, and I. Raeva, "Improved LCM for better IBFS in transportation problems," J. Appl. Math. Bioinform., vol. 6, no. 2, pp. 1–20, 2016.

[10] N. Girmay and T. Sharma, "Balance an unbalanced transportation problem by a heuristic approach," Int. J. Math. Appl., vol. 1, no. 1, pp. 13–19, 2013.

[11] F. Pedregosa et al., "Scikit-learn: Machine learning in Python," J. Mach. Learn. Res., vol. 12, pp. 2825–2830, 2011.

[12] L. Breiman, "Random forests," Mach. Learn., vol. 45, no. 1, pp. 5–32, 2001.

[13] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," Nature, vol. 323, pp. 533–536, 1986.

[14] C. R. Harris et al., "Array programming with NumPy," Nature, vol. 585, pp. 357–362, 2020.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. Int. Conf. Learn. Representations (ICLR), San Diego, CA, 2015.

[16] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA: MIT Press, 2018.

[17] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA: MIT Press, 2016.

[18] F. Hutter, L. Kotthoff, and J. Vanschoren, Eds., Automated Machine Learning: Methods, Systems, Challenges. Cham, Switzerland: Springer, 2019.

[19] J. D. Hunter, "Matplotlib: A 2D graphics environment," Comput. Sci. Eng., vol. 9, no. 3, pp. 90–95, 2007.

[20] E. Zitzler, M. Laumanns, and L. Thiele, "SPEA2: Improving the strength Pareto evolutionary algorithm," TIK-Report 103, ETH Zurich, 2001.