Pontifícia Universidade Católica
do Rio de Janeiro

Arthur Monteiro Ferraz

Districting and Vehicle Routing: Learning the Delivery Costs

Dissertação de Mestrado

Dissertation presented to the Programa de Pós–graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática.

Advisor : Prof. Thibaut Victor Gaston Vidal
Co-advisor: Prof. Quentin Cappart

Rio de Janeiro
September 2022

PUC-Rio - Certificação Digital Nº 2012383/CA

Pontifícia Universidade Católica
do Rio de Janeiro

Arthur Monteiro Ferraz

Districting and Vehicle Routing: Learning the Delivery Costs

Dissertation presented to the Programa de Pós–graduação em Informática of PUC-Rio in partial fulfillment of the requirements for the degree of Mestre em Informática. Approved by the Examination Committee:

Prof. Thibaut Victor Gaston Vidal
Advisor
Departamento de Informática – PUC-Rio

Prof. Quentin Cappart
Polytechnique Montréal

Prof. Alberto Maria Santini
ESSEC Business School

Prof. Marcus Vinicius Soledade Poggi de Aragao
Departamento de Informática – PUC-Rio

Rio de Janeiro, September 19th, 2022

PUC-Rio - Certificação Digital Nº 2012383/CA

All rights reserved.

Arthur Monteiro Ferraz

Obtained bachelor's degree in Computer Science from the Universidade Federal Fluminense (UFF) in July 2019.

PUC-Rio - Certificação Digital Nº 2012383/CA

Bibliographic data

Ferraz, Arthur Monteiro

Districting and Vehicle Routing: Learning the Delivery Costs / Arthur Monteiro Ferraz ; advisor: Thibaut Victor Gaston Vidal ; co-advisor: Quentin Cappart. – 2022.
50 f: il. color. ; 30 cm

Dissertação (mestrado) - Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Informática, 2022.
Inclui bibliografia

1. Informática – Teses. 2. Roteamento de veículos. 3. Deep learning. 4. Metaheuristicas. 5. Aprendizado em grafos. I. Vidal, Thibaut Victor Gaston. II. Cappart, Quentin. III. Pontifícia Universidade Católica do Rio de Janeiro. Departamento de Informática. IV. Título.

CDD: 004

Acknowledgments

I would like to thank my advisors Thibaut Vidal and Quentin Cappart for being so helpful and guiding me to think in a scientific way.

Also to thank professor Anand Subramanian that was fundamental for me to consider doing the masters.

Thanks professors Simone de Lima Martins, Alexandre Plastino, and Isabel Rossetti for introducing me to computer science and for all guidance you gave me on my bachelor's.

Thanks to my family and friends for being so supportive and comprehensive in this hard journey.

To CNPq and PUC-Rio, for the aid granted, without which I would not work have been carried out.

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

PUC-Rio - Certificação Digital Nº 2012383/CA

Abstract

Ferraz, Arthur Monteiro; Vidal, Thibaut Victor Gaston (Advisor); Cappart, Quentin (Co-Advisor). **Districting and Vehicle Routing: Learning the Delivery Costs**. Rio de Janeiro, 2022. 50p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

The districting-and-routing problem is a strategic problem in which basic geographical units (e.g., zip codes) should be aggregated into delivery regions, and each delivery region is characterized by a routing cost estimated over an extended planning horizon. The objective is to minimize the expected routing costs while ensuring regional separability through the definition of the districts. Repeatedly simulating routing costs on a set of scenarios while searching for good districts can be computationally intensive, so existing solution approaches for this problem rely on approximation functions. In contrast, we propose to rely on a graph neural network (GNN) trained on a set of demand scenarios, which is then used within an optimization approach to infer routing costs while solving the districting problem. Our computational experiments on various metropolitan areas show that the GNN produces accurate cost predictions. Moreover, using this better estimator during the search positively impacts the quality of the districting solutions and leads to 10.35% delivery-cost savings over the commonly-used Beardwood estimator and similar gains compared to other approximation methods.

PUC-Rio - Certificação Digital Nº 2012383/CA

Keywords

Routing; Deep Learning; Metaheuristics; Graph Neural Networks.

Resumo

Ferraz, Arthur Monteiro; Vidal, Thibaut Victor Gaston; Cappart, Quentin. **Districting e Roteamento de Veículos: Aprendendo a Estimar Custos de Entrega.**. Rio de Janeiro, 2022. 50p. Dissertação de Mestrado – Departamento de Informática, Pontifícia Universidade Católica do Rio de Janeiro.

O problema de *Districting-and-routing* é um problema estratégico no qual porções geográficas devem ser agregadas em regiões de entrega, e cada região de entrega possui um custo de roteamento estimado. Seu objetivo é de minimizar esses custos, além de garantir a divisão da região em distritos. A simulação para obter uma boa aproximação é muito custosa computacionalmente, enquanto mecanismos como buscas locais exigem que esse cálculo seja feito de forma muito eficiente, tornando essa estratégia de aproximação inviável para uma solução metaheurística. Grande parte das soluções existentes para esse problema utilizam de formulas de aproximação contínua para mensurar os custos de roteamento, funções essas que são rápidas de serem calculadas porém cometem erros significativos. Em contraste, propomos uma Rede Neural em Grafo (*Graph Neural Network* - GNN) que é usada como oráculo por um algoritmo de otimização. Nossos experimentos computacionais executados com dados de cidades do Reino Unido mostram que a GNN é capaz de produzir previsões de custos mais precisas em tempo computacional aceitável. O uso desse estimator na busca local impacta positivamente a qualidade das soluções, levando a uma economia de 10,35% no custo de entrega estimado em relação a função Beardwood, que é comumente usada nesse cenários, e ganhos similares em comparação com outros métodos de aproximação.

Palavras-chave

Roteamento de veículos; Deep Learning; Metaheuristicas; Aprendizado em Grafos.

PUC-Rio - Certificação Digital Nº 2012383/CA

Table of contents

PUC-Rio - Certificação Digital Nº 2012383/CA

List of figures

PUC-Rio - Certificação Digital Nº 2012383/CA

List of tables

PUC-Rio - Certificação Digital Nº 2012383/CA

1
Introduction

Districting is the process of partitioning a service region, represented as a collection of basic geographical units, into larger clusters called districts. This practice is ubiquitous in large-scale transportation and last-mile delivery systems for mail delivery [1], home care services [2], and maintenance services [3]. A delivery policy in fixed districts has several interests: (i) allowing the separation and the aggregation of the requests in advance before all information is available, (ii) reducing the complexity of the task thanks to the decomposition of the routing optimization process, (iii) stimulating the familiarity of drivers and thus their efficiency within their respective geographical regions [4], and (iv) increasing the satisfaction of customers thanks to a higher familiarity with their drivers [5].

Districting decisions are strategic and linked with major financial and societal stakes. These decisions typically hold for a few months or years, whereas operational routes occur on a daily or weekly basis and are subject to variations. Optimizing or even evaluating districting decisions is a very complex task. Demands are typically uncertain and volatile. Moreover, routing cost evaluations typically translate into large-scale vehicle routing problems, which are time-consuming to solve and highly sensitive to the spatial distribution of the requests. Because of these two different classes of decisions and planning horizons, the related *districting-and-routing* problems are subject to important challenges [6, 7].

A common way to estimate costs for a long plan horizon is by solving sample demand scenarios. This method offers the benefit to obtain a good estimation but at the cost of a more intense execution time. As the efficiency of the evaluation step is critical when used in optimization techniques, such as in the local search for driving the neighbor selection, it is impractical to generate optimal routes for each scenario at each optimization step. For this reason, a number of solutions rely on continuous approximation formula to evaluate costs. It is for instance the case of the Beardwood approximation formula on the traveling salesman problem, which roughly estimates routing costs through n independently distributed points in a compact area of size A as $\alpha\sqrt{nA}$, where α is a constant [8]. There are also other solutions based on machine learning

PUC-Rio - Certificação Digital Nº 2012383/CA

methods to evaluate routing costs. To the best of our knowledge, the most efficient related methods are based on shallow neural networks [9], and do not exploit the recent progress in deep learning [10] and are then limited to small-sized case studies.

Based on this context, this work proposes to leverage recent methods in the deep learning toolbox, such as graph neural networks [11] in order to build a model able to learn routing cost. Specifically, any geographical area can be split into a set of contiguous geographical units, called Basic Units (BU), and thus be represented as a network of BUs, where edges link contiguous areas. The goal is to approximate the routing costs associated with a network, given a specific districting plan. Input data for training are randomly generated feasible districts with their expected routing cost. This cost is obtained by solving a number of TSP scenarios, with Lin–Kernighan algorithm [12], and taking their average value as expected cost, a method known as Sample Approximation Approach (SAA) [13]. Once the model has been trained, it is subsequently used as a predictor in the context of a districting solution method. Our solution approach creates an initial districting solution based on a mathematical programming method for graph partitioning, and then subsequently applies local-improvement techniques and perturbation steps (as in the classic ILS framework) to obtain better districting plans. The metaheuristic can use both the graph neural networks or a more traditional approach, like the Beardwood formula, as cost function, in a way that is possible to compare the impact of the different cost estimators in the final solution. In turn, our methodology permits to provide high-quality districting plans for practitioners and permits us to evaluate the importance of accurate routing-cost estimates in the solution process.

The contributions of this work are as follows:

– A deep learning model based on a graph neural network architecture for estimating routing costs in strategic districting problems. Unlike the previous approaches based on shallow neural networks, this new architecture leverages the topological structure of a geographic area.
– An Iterated Local Search (ILS) algorithm that leverages the predicted routing cost for generating high-quality districting plans within minutes. The construction of the initial solution is done by solving an adapted flow formulation for the balanced connected k-partition problem. The local search procedure considers both relocating and swapping between the whole border of each pair of districts. The perturbation mechanism is done by randomly applying moves. This ILS implementation does not rely on any specific cost function, so we can easily compare districting plans obtained with different cost evaluation approaches.

PUC-Rio - Certificação Digital Nº 2012383/CA

– Experiments on five important metropolitan areas in the United Kingdom (Bristol, Manchester, Leeds, London, and West-Midlands), up to 120 basic geographical units. The quality of the routing cost estimation and the performance of the full solution for the districting task are evaluated. Experimental results show that the learning-based approach can reduce significantly the prediction error compared to other commonly used approaches, and enables a districting strategy providing economical gains of 10.40% on average and up to more than 20% in some scenarios. The obtained solutions are evaluated using the SAA in not previously seen scenarios.

This dissertation is organized as follows. Chapter 2 gives a formal definition of our problem and reviews the literature. Next, Chapter 3 details the graph neural network for routing-cost estimation and describes our solution scheme for the tactical districting problem. Chapter 4 reports our numerical experiments, finally Chapter 5 concludes and proposes perspectives for future research.

PUC-Rio - Certificação Digital Nº 2012383/CA

2
Problem statement and Literature Review

The *districting-and-routing* is a problem that arises from the necessity of dividing geographical areas into districts for routing purposes. Workers are assigned to areas on the long-plan horizon without prior knowledge of the customer's demands. That comes with the challenge of estimating transportation costs. A precise way to do it is by solving demand scenarios. Although this method can be used to estimate some solution costs, still not fast enough to be used in a local search mechanism, thus not suitable for a heuristic approach. A widespread way of solving it is by using some continuous approximation formula, that are faster to evaluate but lacks precision. Machine Learning approaches for routing problems are arising in the last few years, especially with the development of graph neural networks. Similarly to the continuous approximation functions, they can be used as an oracle for a higher level heuristic.

The rest of this Chapter is organized as follows. Section 2.1 explores some applications of districting in different contexts. Section 2.2 states the studied problem. Section 2.3 gives a definition for this transportation cost and reviews some common approaches to estimate it in the context of districting. Section 2.4 presents works that rely on Machine Learning techniques to solve routing problems.

PUC-Rio - Certificação Digital Nº 2012383/CA

2.1
Districting Problem and applications

Districting is a very broad class of problems coming in different flavors. The most common ones are political, servicing, sales, and distribution [7]. Political districting is especially important for democracies where each district elects a single member of parliament. Horn [14] compares different compactness measurements in the context of politics. Bozkaya [15] implements a tabu search metaheuristic to solve a multi-criteria version of the problem, considering a weighted sum of some of the popular optimization objectives, like compactness and population equality, for instance. Webster [16] presents a review of different districting measurements and their socio-economical impacts. Servicing is generally associated with social or public facilities. Benzarti [2] presents different

formulations for home health care (HHC) districting, solving it with a MIP solver. Garcia-Ayala [3] proposes an arc-based approach to solve problems where paths (e.g. a road) are a relevant factor, like postal delivery, meter readings, winter gritting, road maintenance, and municipal solid waste collection. Bruno [1] solves the problem of reorganizing postal delivery services in Bologna. Sales territory arises from the necessity of assigning salespeople to client accounts. Zoltners [17] reviews the historical work in sales districting, arguing its importance and presenting the most relevant works until then. Lei [18] solves a variation of the multiple traveling salesmen and the districting problem with multi-periods and multi-depots aiming for sales territory applications. It has implemented an adaptive large neighborhood search with a multi-objective function balancing between some important measurements, similarly to what Bozkaya [15] did in the political context.

Distribution districting is the design of areas focusing on pick-up and delivery services, which is often considered a vehicle routing problem variant. Novaes [19] solves the problem of optimizing vehicles fleet building districts first with ring-radial topology and then optimizing it with a genetic algorithm. Galvao [20] extends this last work using Voronoi diagrams to refine districts. Zhong [4] provides a tabu search heuristic that considers drivers familiarity to estimate routing costs. Carlsson [21] uses geometric arguments to draw lines to determine district boundaries aiming to have a balanced workload.

PUC-Rio - Certificação Digital Nº 2012383/CA

2.2 Districting and Vehicle Routing

Generally speaking, the goal to achieve depends on the application context and can involve different measures such as compactness, balance, service levels, and costs. Our work studies a *districting-and-routing* problem in which the districts are designed in such a way to optimize transportation costs on an extended planning horizon.

Consider a geographical region subdivided into n geographical units called Basic Units (BU). In each BU, transportation operations will be independently performed over a long-term planning horizon. A district d is a set of BUs, i.e., $d \subseteq \{1, \ldots, n\}$. The operational cost of a district is a function $\Phi : 2^{\{1,\ldots,n\}} \to \mathbb{R}$ representing the expected (i.e., long-term) daily cost of delivering customers in this district. The districting-and-routing problem studied in this work consists in partitioning the region into k districts in such a way that (i) each BU belongs to exactly one district, (ii) the number of BUs inside each district belongs to an admissible range $[n_l, n_u]$, (iii) the districts are connected, and (iv) the sum

of the long-term operational costs of the districts is minimized.

The districting-and-routing problem can be formally cast as a graph partitioning problem with a non-separable objective. Let $G(V, E)$ be an undirected graph, where each vertex $i \in V$ is a BU and edges $e \in E$ represent the contiguity between adjacent BUs (i.e., sharing a border). Let $\Omega = 2^V$ be the set of all possible feasible districts respecting size and connectivity constraints. Then, the problem can be formulated as the following integer program:

$$\min \sum_{d \in \Omega} \Phi(d) \lambda_d \tag{2-1}$$

$$\text{s.t.} \sum_{d \in \Omega} e_{id} \lambda_d = 1 \qquad i \in V \tag{2-2}$$

$$\sum_{d \in \Omega} \lambda_d = k \tag{2-3}$$

$$\lambda_d \in \{0, 1\} \qquad d \in \Omega. \tag{2-4}$$

For each d, the binary variable λ_d takes value 1 if district d is selected in the solution. Parameter $e_{id} = 1$ if BU i appears in district d, and 0 otherwise. Objective (2-1) corresponds to the total cost of the selected districts. Constraint (2-2) ensure that each BU is present in one district, and Constraint (2-3) fixes the number of districts.

This formulation is essentially for descriptive use, as there are two main barriers to its direct solution by MIP solvers. First, it includes an exponential number of variables $d \in \Omega$. Moreover, the non-separability of Φ makes it difficult to design decomposition approaches. For these reasons, we can use this formulation only for smaller problems, to produce baseline solutions. For larger cases, more efficient methods are needed. Designing scalable solution methods requires (i) efficient algorithms to estimate districts operational costs, and (ii) faster search strategies for the districting problem.

2.3 Long-term Operational Cost

The long-term operational cost $\Phi(d)$ of any district d is defined as follows. Any BU $i \in \{1, \ldots, n\}$ has ξ_i inhabitants which are spread over its geographical area. On any given day, each inhabitant may request a delivery with probability p. Therefore, the number of demands on any given day for a BU is given by a binomial distribution $\mathcal{B}(\xi_i, p)$, and the locations of these demands are selected randomly with uniform probability within the BU. The cost of the district corresponds to the expected cost of a TSP tour, leaving and returning from

PUC-Rio - Certificação Digital Nº 2012383/CA

the depot, and visiting the customers generated by the aforementioned random process in all its BUs. With this definition of the district costs, function Φ is monotonic, i.e., $\Phi(d') \leq \Phi(d)$ if $d' \subseteq d$. However, contrary to intuition, Φ is not submodular, by direct consequence of the non-submodularity of TSP costs (see [22]).

Drivers will be assigned to a specific region and work within it for months or even years. Given the uncertainty of the demand, it is not an easy task to get the value of $\Phi(d)$. A precise way to do it is by using Monte Carlo simulation, a method known as Sample Average Approximation (SAA). It consists in generating demand scenarios and taking their average TSP cost as the expected cost. When the number of scenarios grows to infinity, this estimation approach converges towards the true cost $\Phi(d)$ with probability one [23]. Although this method is practical to evaluate a single districting solution, becomes a considerable bottleneck when numerous districting solutions must be evaluated.

For this reason, several applications have replaced the estimations based on simulations with surrogate measures of routing cost, cheaper to compute. A useful tool to tackle this problem is the Beardwood–Halton–Hammersley (`BHH`) theorem [24]. It states that the optimal routing cost of R randomly distributed points within an area A is:

$$\texttt{BHH}(d) = \alpha\sqrt{RA}, \text{ where } \alpha \in \mathbb{R}_{\geq 0} \tag{2-5}$$

This formula has been further extended by Daganzo (`DAG`) [25] for vehicle routing problems. This extension proposes the usage of a new variable L, defined as the average distance between points inside the district and the depot. It also finds the best α value for a number of vehicle routing scenarios.

$$\texttt{DAG}(d) = 2L + 0.57\sqrt{RA} \tag{2-6}$$

The `BHH` and `DAG` equations assume *a priori* knowledge of the area of the service region in which the customers are located. Chien [26] proposes the use of Monte Carlo simulations to determine the best α constant for different estimators, a pretty similar approach we use in our work. It shows that `DAG` with a right α could estimate the TSP costs with 5–10% of prediction error in scenarios limited up to 30 customers.

A widespread way to estimate routing costs for districting purposes is by using `DAG` equation [18, 19, 20, 27, 28, 29]. There are other continuous approximation formulas in the broader literature [8, 30], but we could not find a study that use any formula other than `DAG` to directly estimate long-term routing costs. Another common approach [1, 2] is to consider some compactness

PUC-Rio - Certificação Digital Nº 2012383/CA

formula as a proxy measure for estimating the traveling distance. It is reasonable to assume that less compact areas will lead to higher routing costs.

2.4 Machine learning approaches for routing cost estimations

The first use of machine learning for approximating routing costs has been proposed by Kwon [9]. To do so, they construct several regression models and shallow neural networks that are able to produce accurate estimates of the optimal length of a traveling salesman tour of customers located in a rectangular region. It shows that even using the same features as a simpler regression model, the neural networks were able to enhance the prediction. Also it was able to outperform `DAG` formula and Chiens [26] approach. Because of computational limitations, no instance larger than 80 customers were considered and only a shallow neural network was built (1 hidden layer with 3 neurons).

More recently, Nicola [31] relies on several positional features of customers to approximate the optimal cost for TSP, Capacitated Vehicle Routing Problem with Time Window (CVRP-TW), and Multi-Region Multi-Depot Pickup and Delivery Problem (MR-MDPDP). It was able to produce estimators with errors of about 2% when customer positions are given. Akkerman [32] tested multiple regression models to run customer selection on a multi-period vehicle routing problem. Using a Gradient Boost Regressor, it was able to reduce up to 17% of the waste collection routing costs when compared to the usage of `DAG` as an estimator in scenarios where customer positions are previously known.

In another context, the last decade has shown the rise of deep learning architectures [10]. Thanks to the increased computational power and the number of available data, deeper and more sophisticated neural network architecture could achieve exceptional performances on many different tasks, from computer vision [33] to natural language processing [34]. In this context of deep learning, Belo [35] proposes a reinforcement learning with an LSTM approach to solve combinatorial problems. It was able to optimally solve instances with 200 items for the Knapsack Problem. Also, they have solved TSP instances up to 100 nodes, but could not consistently find optimal solutions. Deudon [36] relies on an attention mechanism [34, 37] instead of using an LSTM to improve the solving process, enhancing the results obtained by Belo [35]. Attention is also used by Kool [38], who proposes an extended model able to solve different routing problems, like Prize Collecting TSP (PCTSP) and Stochastic PCTSP (SPCTSP).

A recent and popular architecture for solving approximately routing problems is the so-called Graph Neural Network (`GNN`) [11, 39]. Similar to

PUC-Rio - Certificação Digital Nº 2012383/CA

convolutional neural networks that are dedicated to learn from spatial data such as images, graph neural networks are specialized to learn from data having a graph structure, such as in many combinatorial problems [40, 41, 42, 43]. Joshi [44] has implemented a Graph Neural Network (GNN) with beam search to solve the TSP, achieving an average gap of 1.39% for 100 nodes, improving the previous learning mechanisms to solve the problem but still remaining far from standard optimization approaches. An extensive study of graph neural networks in combinatorial optimization is proposed by Cappart [45].

Interestingly, despite the interest in using deep learning to solve routing problems, we could not find a work that proposes an approach for the *districting-and-routing* problem that relies on these advances.

PUC-Rio - Certificação Digital Nº 2012383/CA

3
Methodology

The proposed methodology consists in training a Graph Neural Network (GNN) to estimate routing costs in a long plan horizon, and then use it as an oracle for an Iterated Local Search (ILS) metaheuristic that produces district designs. The hypothesis is that GNN provides a better estimator than the classic approaches, and still fast enough to be used in the heuristic, thus creating solutions with lower routing costs. Section 3.1 provides details about the GNN, and Section 3.2 explains the ILS algorithm.

3.1
Delivery-Cost Estimations with a GNN

We focus on the estimation of delivery costs for the districts. Given the generative model for demand distribution described in Section 2.3, Sample Average Approximation (SAA) provides a simple estimation approach which generates a set of scenarios and calculates the average TSP costs over them. When the number of scenarios grows to infinity, SAA estimates converge towards the true costs $\Phi(d)$ with probability one [23]. Clearly, in practical situations, a finite sample size is used (e.g., 50 scenarios per BU in our experiments in Chapter 4). Still, while this approach can be practical to evaluate one solution of the districting-and-routing problem (represented as a fixed list of districts), it becomes impractically slow within a search method for the districting problem (e.g., a local search), due to the large number of candidate districts considered through the search during move evaluations.

To provide faster and accurate estimations in this context, we now explore the option to *learn* an approximation of the delivery costs by supervised learning, more specifically by relying on the formidable methodological progress recently made on deep learning and graph neural networks (GNN). This approach is described in the remainder of the section, starting from the feature information used for training, the architecture of the network, and the training process.

PUC-Rio - Certificação Digital Nº 2012383/CA

3.1.1 Features

Our GNN leverages the same undirected graph $G(V, E)$ as the one depicted in Section 2.2, with the vertices V corresponding to BUs and edges in E correspond to adjacency relations. The GNN will be trained on a set of district samples $\Omega^{\text{TRAIN}} \subset \Omega$, which consists of randomly-generated districts on which we have previously evaluated delivery costs with SAA. Each district sample $d \in \Omega^{\text{TRAIN}}$ will be characterized by a vector of features on each node of the graph and an estimated cost value. We use eight features $\boldsymbol{f}_{vd} = (\xi_v, \sqrt{\xi_v}, q_v, a_v, \sqrt{a_v}, \rho_v, \delta_v, e_{vd})^\top \in \mathbb{R}^8$ for each vertex v (associated to a BU) and each district sample d:

1. the **population** ξ_v of the BU;
2. the **sqrt-population** $\sqrt{\xi_v}$ of the BU;
3. the **perimeter** q_v of the BU;
4. the **area** a_v of the BU;
5. the **sqrt-area** $\sqrt{a_v}$ of the BU;
6. the **density** $\rho_v = \xi_v / a_v$ of the BU;
7. the **distance to the depot** δ_v, corresponding to the minimum distance between the depot and any point in the BU;
8. an **inclusion variable** e_{vd}, taking value 1 if BU v belongs to d and 0 otherwise.

It is noteworthy that the first seven features remain fixed when considering examples generated in the same metropolitan area. Only the last feature changes, according to the current district considered in the sample.

3.1.2 Architecture of the Graph Neural Network

The neural network used for the prediction includes three main parts:

1. a **node embedding layer**, taking as input a district sample in the format described previously and whose final output is a vector of features for each node of the graph. These outputs are also referred to as *node embedding*. Intuitively, it is a function that aggregates feature information defined over a graph into a vectorial representation of the features, by aggregating information from neighboring nodes. This aggregation is done several times and corresponds to a layer of the GNN.

2. an **graph embedding layer** whose aggregates each node embedding into a single vector through a non-linear transformation. Intuitively, this

PUC-Rio - Certificação Digital Nº 2012383/CA

vector is a latent representation of the input graph. It is also referred to as the *graph embedding*.

3. a **fully-connected neural network** whose purpose is to fit predicted values from the graph embedding. Its output is the districting cost that we want to estimate.

A high-level representation of this architecture is provided in Figure 3.1. It predicts the associated routing cost $\hat{\Phi}(d)$ of a district d given as input and represented as a graph. Detailed information about the three components of the architecture are proposed in this section.

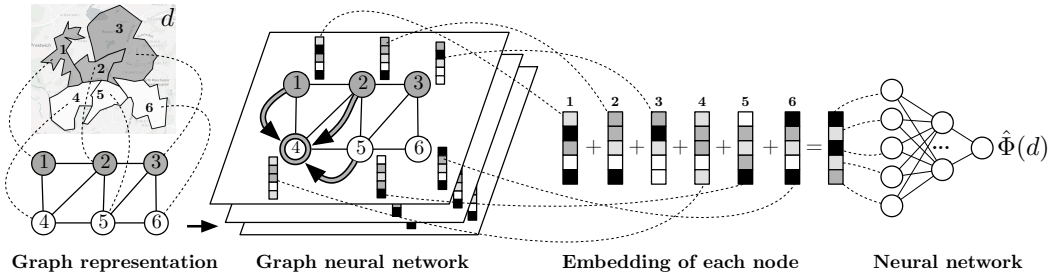


Figure 3.1: Neural architecture dedicated to approximate delivery costs.

Node Embedding Layers.

Let $G(V, E)$ be the graph representation of a metropolitan area, and $\boldsymbol{f}_{vd}$ be the features of each nodes $v \in V$ for a district sample $d \in \Omega$. Formally, a `GNN` computes a m-dimensional features embedding $\boldsymbol{\mu}_v \in \mathbb{R}^m$ for each node $v \in V$ in G (i.e., the node embedding). The node features $\boldsymbol{f}_{vd}$ are aggregated iteratively with the neighboring nodes in the graph. After a predefined number of aggregation steps, the embedding of each node is produced and encompasses both local and global characteristics of the graph. These operations can be carried out in different ways [46, 47, 48], and public implementations are available for many of these architectures. Following [49] who considered graph neural networks solving combinatorial problems over graphs, this paper is based on their implementation referred to as STRUCTURE2VEC [46].

The behavior is as follows. Let T be the number of aggregation steps, $\boldsymbol{\mu}_v^t$ be the node embedding of v obtained after t steps, and $\mathcal{N}(v)$ the set of neighboring nodes of $v \in V$ in G. The computation of an embedding $\boldsymbol{\mu}_v^{t+1}$ is presented in Equation (3-1), where $\boldsymbol{\theta}_1 \in \mathbb{R}^{p \times w}$ and $\boldsymbol{\theta}_2 \in \mathbb{R}^{p \times p}$ are tensors of weights that are learned during the training phase, and $\texttt{ReLU}(x) = \max(0, x)$ is a non-linear activation function commonly used in deep neural networks [50]:

$$\boldsymbol{\mu}_v^{t+1} = \texttt{ReLU}\left(\boldsymbol{\theta}_1 \boldsymbol{f}_{vd} + \boldsymbol{\theta}_2 \sum_{u \in \mathcal{N}(v)} \boldsymbol{\mu}_u^t\right) \quad \forall t \in \{1, \ldots, T\}. \tag{3-1}$$

PUC-Rio - Certificação Digital Nº 2012383/CA

The idea is to compute the new embedding $\boldsymbol{\mu}_v^{t+1}$ as a parametrized sum of the previous embedding ($\boldsymbol{\mu}_v^{t+1}$) and the node features ($\boldsymbol{f}_{vd}$). Following the recommendations of the initial implementation, four aggregation steps are done ($T = 4$), each hidden embedding is a vector of 64 values ($\boldsymbol{\mu}_v^2, \boldsymbol{\mu}_v^3, \boldsymbol{\mu}_v^4 \in \mathbb{R}^{64}$). Then, a last aggregation step is performed on the last hidden embedding. This is presented in Equation (3-2), where $\boldsymbol{\theta}_3 \in \mathbb{R}^{k \times p}$ is another weight tensor. In our case, k is set to 1024, which yields a 1024-dimensional vector as the output embedding ($\boldsymbol{\mu}_v^{\mathtt{out}} \in \mathbb{R}^{1024}$):

$$\boldsymbol{\mu}_v^{\mathtt{out}} = \mathtt{ReLU}\Big(\boldsymbol{\theta}_3 \sum_{u \in \mathcal{N}(v)} \boldsymbol{\mu}_u^T\Big). \tag{3-2}$$

Graph Embedding layer.

Once a vectorized representation $\boldsymbol{\mu}_v^{\mathtt{out}}$ has been computed for each vertex v, this information is used to compute $\boldsymbol{z}$, a vectorized representation of the entire graph. This is done by summing together the embedding of each node and applying a non-linear transformation (e.g., `ReLU`) to the result. This is illustrated in Equation (3-3) where $\boldsymbol{\theta}_4 \in \mathbb{R}^{k \times k}$ and $\boldsymbol{\theta}_5 \in \mathbb{R}^{k \times k}$ are two other tensors of weights that must be learned ($k = 1024$). This is also referred to as a *pooling* operation:

$$\boldsymbol{z} = \boldsymbol{\theta}_4 \mathtt{ReLU}\Big(\boldsymbol{\theta}_5 \sum_{u \in V} \boldsymbol{\mu}_u^T\Big). \tag{3-3}$$

The transformations carried out by Equations (3-1–3-3) can be summarized as a parametrized function $\mathtt{GNN} : (G \times \mathbb{R}^{8V}) \to \mathbb{R}^{1024}$, which takes as input a graph decorated with eight features at each node, and returns a vector of 1024 features characterizing the graph.

Fully-Connected Neural Network.

Finally, the embedding $\boldsymbol{z}$ goes through a standard fully-connected neural network of 2 layers, with 100 neurons for the hidden layer and a single neuron for the output layer. This neural network can be represented as a function $\mathtt{FCNN} : \mathbb{R}^{1024} \to \mathbb{R}$, that computes the expected delivery cost $\hat{\Phi}(d)$ inside a district d thanks to the pre-computed graph embedding and two additional tensor of weights: $\boldsymbol{\theta}_6 \in \mathbb{R}^{100 \times 1024}$ and $\boldsymbol{\theta}_7 \in \mathbb{R}^{1 \times 100}$. Assembling all the pieces together, the delivery cost of a district d, represented as a graph G and the features $\boldsymbol{f}_{vd}$ for each $v \in V$ is computed as follows:

$$\hat{\Phi}(d) = \mathtt{FCNN}\left(\boldsymbol{z}\Big(G, \Big[\boldsymbol{f}_{vd} \;\Big|\; v \in V\Big]\Big) \;;\; \boldsymbol{\theta}_6, \boldsymbol{\theta}_7\right). \tag{3-4}$$

PUC-Rio - Certificação Digital Nº 2012383/CA

3.1.3 Training

The network is trained using back-propagation using the mean absolute error as loss function (L1 loss) to find good values of the tensors $\boldsymbol{\theta}_1$, $\boldsymbol{\theta}_2$, $\boldsymbol{\theta}_3$ $\boldsymbol{\theta}_4$, $\boldsymbol{\theta}_5$, $\boldsymbol{\theta}_6$, and $\boldsymbol{\theta}_7$. The training is carried out for a maximum of 24 hours or $10,000$ epochs (with a batch size of 64) using Adam optimizer [51]. As an additional early stopping criterion, the training is aborted when no reduction in the loss is observed after $1,000$ consecutive epochs. We used the default values for the optimizer ($\beta_1 = 0.9$, $\beta_2 = 0.999$, and no weight decay), except for the learning rate which was set to 10^{-4} instead of 10^{-3} to better stabilize the training.

3.2 ILS for the districting-and-routing problem

Having trained a `GNN` to approximate the delivery costs, we now have a fast prediction oracle capable of calculating the cost of any district. We can now focus on the solution of the districting-and-routing problem given this information.

As previously discussed, the Graph partition formulation (Section 2.2) is not adequate for a direct solution. We will opt for a heuristic solution approach, described in the following. For the heuristic, we rely on the Iterated Local Search (ILS) principle, which is a well-known metaheuristic dedicated to solve a large set of combinatorial optimization problems [52]. The general approach is presented in Algorithm 1. First, an initial solution is generated and a local search is executed in order to improve the solution until a local minimum has been reached. Then, the solution is perturbed (e.g., by applying random moves or by destroying partially the solution) in order to find another solution. Finally, the local search is executed again on the new solution in order to find, hopefully, a better local minimum. This process is iteratively repeated until a stopping criterion has been reached. An important and non-trivial design choice is the perturbation scheme. It must be strong enough to be able to discover another local minimum, but also preservative enough to prevent a too strong degradation of the solution. Thanks to its wide range of applications, iterated local search appears as a natural candidate to perform the districting optimization on top of the district-cost estimation method provided by the graph neural network. This section describes the different components of our solving approach: the construction of an initial solution, the local search moves, and the perturbation scheme.

PUC-Rio - Certificação Digital Nº 2012383/CA

```
1  s ← buildInitialSolution()
2  s ← localSearch(s)
3  s* ← s
4  while stop criteria not achieved do
5  |   s ← pertubate(s)
6  |   s ← localSearch(s)
7  |   if cost(s) < cost(s*) then
8  |   |   s* ← s
9  |   end
10 end
11 return s*
```

Algorithm 1: Iterated Local Search (ILS)

3.2.1 Initial Solution

Finding a feasible initial solution to the districting problem is NP-complete, and relying only on constructive heuristics do not provide a guarantee that the solution obtained will satisfy all the constraints. Consequently, we propose to use an exact solver to compute a feasible initial solution. This is done by extending a network flow formulation of a balanced connected k-partition problem [53] and by solving it thanks to a mixed-integer programming solver.

Given the initial graph $G(V, E)$, we can extend it to a digraph $G'(V \cup S, E')$, where S is the set of k vertices, each one representing a district. The set E' of edges is built by replacing every edge $e \in E$ with a pair of arcs connecting the same two nodes, each one in different directions, and then adding an arc from each vertex $s \in S$ to each original vertex $v \in V$. Variable $y_{i,j}$ is a binary value that is 1 if (i, j) has some flow and $f_{i,j}$ is the amount of flow. Variable δ_v is the distance from a vertex v to the depot. Also, we define $V' = \{V \cup S\}$ as the set of all nodes in an instance, grouping basic units and flow sources.

PUC-Rio - Certificação Digital Nº 2012383/CA

$$\min \sum_{s \in S} \sum_{v \in V} \delta_v y_{s,v} \tag{3-5}$$

$$\text{s.t.} \sum_{v \in V} f_{s,v} \leq \sum_{v \in V} f_{s+1,v} \qquad \forall s \in [0, k-1] \tag{3-6}$$

$$\sum_{i \in V'} f_{i,v} - \sum_{i \in V'} f_{v,i} = 1 \qquad \forall v \in V \tag{3-7}$$

$$f_{i,j} \leq n_u y_{i,j} \qquad \forall i, j \in V' \tag{3-8}$$

$$\sum_{v \in V} f_{s,v} \geq n_l \qquad \forall s \in S \tag{3-9}$$

$$\sum_{v \in V} y_{s,v} \leq 1 \qquad \forall s \in S \tag{3-10}$$

$$\sum_{s \in S} y_{s,v} \leq 1 \qquad \forall v \in V \tag{3-11}$$

$$y_{i,j} \in \{0, 1\} \qquad \forall i, j \in V' \tag{3-12}$$

$$f_{i,j} \in \mathbb{R}_{\geq} \qquad \forall i, j \in V' \tag{3-13}$$

The objective function (3-5) aims to minimize the sum of the minimum depot distance in each district. It favors that the initial flow from any i goes to some basic unit that is close to the depot. The idea behind this objective is to have the lowest depot travel cost possible. Constraint (3-6) order the sources by flow, it is useful to break symmetry between solutions. Constraints (3-7) guarantee that each vertex $v \in V$ consume 1 unit o flow. Constraint (3-8) sets the maximum flow of any arc, if it is being used. Constraint (3-9) defines the minimum flow coming out of each source. (3-10) says that at most one arc leaves each source. (3-11) defines that at most one source arc must come in any $v \in V$. Finally (3-12) defines $y_{i,j}$ as a binary and (3-13) defines $f_{i,j}$ as a real positive number.

After solving this formulation using a MIP solver, we need to transform the result into a solution for the districting problem. It is done by taking the set of nodes achieved by each source and grouping them into a district. The flow guarantees that districts are contiguous sets of basic units and the set of constraints (3-8, 3-9) guarantees that bounds are respected.

3.2.2 Local Search

The local search improves the solution until a local minimum has been found. The neighborhood is defined as the union of two moves: (1) RELOCATE(u, d), which reassign the basic unit u to a new district d, and (2) SWAP(u, v), which permutes the district of two basic units u and v. Besides, the moves generating infeasible solutions are not allowed. Assuming a total of $|D|$

PUC-Rio - Certificação Digital Nº 2012383/CA

districts and $|B|$ basic units, the neighborhood has a size of $\mathcal{O}(|D| \times |B| + |B|^2)$ neighbours.

```
1 while Improvement Found do
2 |   for all pair of neighbor districts (d_i, d_j) in random order do
3 |   |   Calculate borders B_{i,j} and B_{j,i};
4 |   |   Find and apply, if improving, the best move on s among:
5 |   |     • all feasible RELOCATE moves for (B_{i,j}, d_j) ∪ (B_{j,i}, d_i);
6 |   |     • all feasible SWAP movements for (B_{i,j}, B_{j,i});
7 |   end
8 end
```

Algorithm 2: Local search procedure

The local search procedure is illustrated by Algorithm 2 and it runs as it follows. We iterate over every pair of districts (d_i, d_j), such that d_i and d_j are neighbors, in random order. For each pair it is evaluated all possible moves in the borders. A border $\mathcal{B}(d_i, d_j)$ is defined as the set of nodes of a district d_i that have at least one neighbor that belongs to district d_j. For every $u \in \mathcal{B}(d_i, d_j)$ it evaluates the RELOCATION of u to district d_j, the same happens for every $v \in \mathcal{B}(d_j, d_i)$ to d_i. Also it evaluates SWAP movements between every basic unit in both borders. For every pair (d_i, d_j) it is applied the best movement. The procedure runs until no improving solution is found.

Let's take Figure 3.2 to illustrate local search procedure. Suppose that we have 6 BUs and three different districts. The first figure shows us the map figure that is transformed into a graph for didactic reasons. Assuming that the current iteration is evaluating moves between districts I and II, the set of candidate RELOCATE movements in this scenario is {(A,II), (B,II), (C,II), (D,I), (E,I), (F,I)} and the set of candidate SWAP movements is {(A, D), (A, E), (B, D), (B, E), (C, D), (C, E)}. Let's assume that SWAP(C, D), a movement that keeps the solution feasible, is the one that produces the cheapest solution, then this move is applied. Notice that even that C and D are not neighbors it is possible to swap them because C $\in \mathcal{B}_{\text{I,II}}$ and D $\in \mathcal{B}_{\text{II,I}}$.

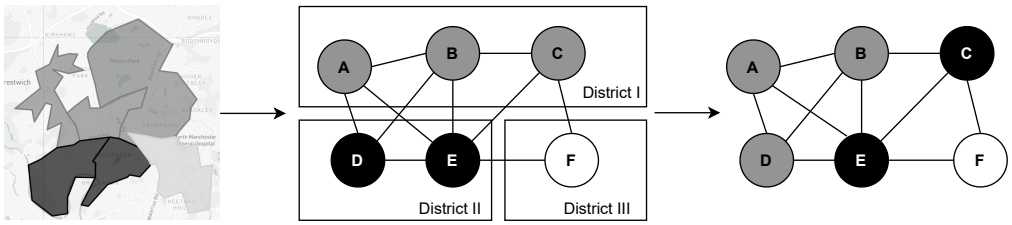


Figure 3.2: Local Search example

PUC-Rio - Certificação Digital Nº 2012383/CA

3.2.3 Perturbation

The perturbation procedure is based on the same set of moves as the local search, but instead of evaluating the cost of a solution, they are randomly applied based on a probability p_{RM}. In this scenario, many movements can be applied to the same pair of districts (d_i, d_j). Constant p_{RM} is a parameter of the `ILS` algorithm, details about its tuning are in Section 4.3.

PUC-Rio - Certificação Digital Nº 2012383/CA

4
Experimental Analyses

To conduct our experimental analyses, we study districting-and-routing problems that occur in five metropolitan areas in the UK (London, Bristol, Manchester, Leeds, and West-Midlands) with very diverse geographical characteristics. The goal of our experiments is twofold. First, we aim to evaluate how accurate the proposed GNN is in terms of routing-cost estimation accuracy. Next, we measure to which extent cost-estimation accuracy impact the ability to take good strategic decisions, i.e., to what extent solutions of districting problems using the GNN or other cost predictors differ in terms of their partitions and operational efficiency.

In the rest of this Chapter, we discuss the data collection and generation of test instances (Section 4.1), as well as the baseline methods considered for routing-cost estimation (Section 4.2). Then, we evaluate the ILS algorithm to tune its perturbation parameter and check if it has any bias towards a cost estimator (Section 4.3). Next, we analyze the accuracy of the different prediction cost-estimation methods (Section 4.4), we measure the impact of different estimation algorithms when optimizing districts (Section 4.5) and finally we discuss the characteristics (e.g., compactness and balance) of the partitions thereby generated (Section 4.6).

PUC-Rio - Certificação Digital Nº 2012383/CA

4.1
Data Collection and Experimental Setup

Geographical data and test instances. We base our studies on five metropolitan areas located in the UK (London, Bristol, Manchester, Leeds, and West-Midlands). The selection of these areas was driven by their diversity and availability of precise geographical boundaries from `https://movement.uber.com`, as well as population statistics from the UK government 2018 census database [54]. The BUs correspond to Middle Layer Super Output Areas (MSOAs), which are designed to contain roughly the same population (each MSOA contains over 5,000 inhabitants, and 8,000 inhabitants on average over the UK). Finally, to obtain data sets with a different number of BUs ($n = \{60, 90, 120\}$), we selected a center point in each region and retained the n closest BUs.

Table 4.1 provides general statistics (population, area, density, and compactness) on the $n = 120$ BUs of each metropolitan area. The area of each BU was measured using Monte Carlo method with 50,000 samples. Compactness scores have been obtained using Reocks formula [55, 56], which divides the area of the BU by the area of the smallest circumscribed circle. Therefore, this compactness measure assigns higher scores to areas that are more round-shaped.

		Bristol	**Leeds**	**London**	**Manchester**	**West-Mid.**
Population (thousands)	average	8.32	7.53	9.68	8.69	8.71
	std	2.29	1.64	2.03	2.36	2.06
	min	5.55	5.20	6.58	5.26	5.44
	median	7.68	7.26	9.40	8.32	8.19
	max	18.16	14.06	16.17	15.87	17.12
Area (km^2)	average	10.34	4.67	0.75	2.22	1.81
	std	22.42	7.22	0.47	1.17	0.75
	min	0.63	0.35	0.30	0.59	0.53
	median	1.99	2.51	0.64	1.90	1.68
	max	171.21	51.79	3.58	6.69	4.39
Density (thousands/km^2)	average	3.92	3.61	15.15	4.87	5.51
	std	2.89	3.41	4.91	2.58	2.41
	min	0.06	0.14	2.76	1.12	1.99
	median	3.74	2.90	15.17	4.41	5.21
	max	12.72	25.20	28.27	16.36	17.06
Compactness	average	0.40	0.43	0.43	0.42	0.44
	std	0.10	0.09	0.09	0.10	0.09
	min	0.19	0.25	0.21	0.20	0.22
	median	0.40	0.43	0.43	0.42	0.44
	max	0.65	0.67	0.63	0.66	0.61

Table 4.1: Population and geography statistics for the BUs

Besides the characteristics of the BUs, other factors impact the structure of the district-and-routing problems: the admissible range for the number of BUs in each district $[n_{\mathrm{L}}, n_{\mathrm{U}}]$, and the location of the depot. We generated five different configurations for the district-size constraints by setting $n_{\mathrm{L}} = \lfloor 0.8 \times t \rfloor$ and $n_{\mathrm{U}} = \lceil 1.2 \times t \rceil$ with $t \in \{3, 6, 12, 20, 30\}$. Then, for each configuration, the probability π that an inhabitant will make a request has been set to $\pi = 96/(8000t)$ where 96 corresponds to the targeted number of requests in the related routing problem, and 8000 corresponds to the average population of a BU. The values for parameter π have been selected to reflect realistic scenarios with delivery routes that cover close to a hundred stops (typical for parcel deliveries). Finally, five possible locations for the depot D were considered: at the center of the metropolitan area (`C`), at the north-east (`NE`), at the north-west (`NW`), at the south-east (`SE`), and at the south-west (`SW`) of the center ($D = \{\texttt{C}, \texttt{NE}, \texttt{NW}, \texttt{SE}, \texttt{SW}\}$). All these factors are summarized in in Table 4.2. For each of the five considered metropolitan areas, we therefore generated $3 \times 5 \times 5 = 75$ instances covering all the possible combinations of these factors.

PUC-Rio - Certificação Digital Nº 2012383/CA

Factor	Values
Total number of BUs	$n \in \{60, 90, 120\}$
Target number of BUs in a district	$t \in \{3, 6, 12, 20, 30\}$
Depot location	$\{\texttt{C}, \texttt{NE}, \texttt{NW}, \texttt{SE}, \texttt{SW}\}$
Minimum number of BUs in a district	$\lfloor 0.8 \times t \rfloor$
Maximum number of BUs in a district	$\lceil 1.2 \times t \rceil$
Request probability	$\pi = 96/(8000t)$

Table 4.2: Summary of the different instance parameters

Scenarios and solution evaluation. The districting-and-routing problem has an objective function that must be *simulated* over an extended planning horizon. Consequently, to standardize the evaluation procedures among the different evaluation and solution methods, it was essential to (i) create a common set of demand scenarios (positions of customers demands) for training and testing and (ii) to adopt a gold standard for solution measurement. We did this by sampling, for each BU, $n_{\text{TRAIN}} = 50$ and $n_{\text{TEST}} = 50$ random scenarios for training and testing, respectively. Therefore, let $\mathcal{R}^{\text{X}}_{it}$ be the set of customer requests, characterized by their positions, for any BU i and scenario t for $\text{X} \in \{\text{TRAIN}, \text{TEST}\}$. All methods have only access to the training set of scenarios during the learning and solution process. We reserve the other scenarios for the final evaluation and comparison of the solutions. For each demand scenario of the BUs, we obtain a corresponding demand scenario for any district by compounding all the demand locations of the BUs it contains. With this, the training and testing cost of a solution $\mathcal{S}$ described as a set of districts are defined as the average cost of the TSP tours over the scenarios, calculated as:

$$\Phi^{\text{X}}_{\text{SAA}}(\mathcal{S}) = \frac{1}{n_{\text{X}}} \sum_{t=1}^{n_{\text{X}}} \sum_{d \in \mathcal{S}} C_{\text{TSP}} \left(\bigcup_{i \in d} \mathcal{R}^{\text{X}}_{it} \right), \tag{4-1}$$

for $\text{X} \in \{\text{TRAIN}, \text{TEST}\}$, and where $C_{\text{TSP}}(\mathcal{R})$ calculates the cost of a TSP tour visiting the depot and all customers in $\mathcal{R}$. We rely on Lin-Kernighan (LKH) algorithm (from [57], available at `http://webhotel4.ruc.dk/~keld/research/LKH/`) to measure TSP distances.

Computational Environment. All experimental analyses (including training, calibration, and optimization) have been conducted on a computer with an Intel E5-2683 v4 Broadwell 2.1GHz CPU, 124GB RAM, and an Nvidia P100 Pascal (12GB HBM2 memory) GPU. To implement the `GNN`, we use the official implementation of `structure2vec` through its `Python` interface, available at `https://github.com/Hanjun-Dai/pytorch_structure2vec`. The ILS is implemented in C++, compiled with G++ v9.3.0.

PUC-Rio - Certificação Digital Nº 2012383/CA

4.2 Baselines for Routing Cost Estimation

We identified three main estimation approaches for routing costs in districting problems: Beardwoods's formula [24] extended by Daganzo [25] (called BHHD in the rest of this paper), the variant of this formula by Figliozzi [30] (FIG), and the shallow neural network designed by Kown [9] (SNN). We will rely on these methods for our experimental comparisons.

– **BHHD**: Let d be a district composed of n BUs. For any BU i, we recall that a_i is the area of the BU and ξ_i is its population. Moreover, π is the probability of a delivery request for any inhabitant. With this, the total area of the district is calculated as $A_d = \sum_{i \in d} a_i$ and the expected number of deliveries within the district is $R_d = \pi \sum_{i \in d} \xi_i$. Extending the formula of [24] to account for the distance from the depot as in [25], we obtain:

$$\Phi_{\text{BD}}(d) = \beta \sqrt{A_d R_d} + 2\Delta_d, \tag{4-2}$$

where β is a hyper-parameter that needs calibration, and Δ_d is the average distance between the depot and a request, calculated via Monte-Carlo estimation on the training scenario set. In our experiments, we set β to minimize the mean-squared error over a training set (discussed at the end of this section) through least-squares regression. We note that Beardwood's formula converges almost surely to the true expected distance in an asymptotic regime where the number of delivery requests tends towards infinity. Moreover, its usual application in Equation (4-2) also assumes that demand density is uniform over the district. As a consequence, approximation errors will naturally occur as we deviate from these assumptions.

– **FIG**: This continuous approximation formula of [30] is a direct extension of Equation (4-2), which was designed to cope with the practical non-uniformity of real observed demands. The formula is defined in Equation (4-3), and includes four hyper-parameters (β_1, β_2, β_3, β_4) requiring calibration. As previously, Δ_d represents the average distance between the depot and the customer requests. As previously, the values of the hyperparameters are selected through least-squares regression.

$$\texttt{FIG}(d) = \beta_1 \sqrt{A_d R_d} + \beta_2 \Delta_d + \beta_3 \sqrt{\frac{A_d}{R_d}} + \beta_4 \tag{4-3}$$

– **SNN**: This estimator is based on a neural network with a single hidden layer of three neurons. Five features are used as input of the network:

(1) The expected number of deliveries in the district (R_d);

PUC-Rio - Certificação Digital Nº 2012383/CA

(2) The ratio between the length d^l and the height d^h of a minimal-area rectangle covering the district: (d^l/d^h);

(3) The average distance of a customer from the district d to the depot (Δ_d);

(4) Feature (2) divided by Feature (1).

(5) The BHH distance estimate $(\sqrt{A_d R_d})$.

Let $\boldsymbol{x}_d \in \mathbb{R}^5$ be the vector of these five features for a district d. The routing cost estimation is given by Equation (4-4) where $\boldsymbol{w}_1 \in \mathbb{R}^{3\times5}$ and $\boldsymbol{w}_2 \in \mathbb{R}^{1\times3}$ are the network weights, and where $\boldsymbol{b}_1 \in \mathbb{R}^3$ and $b_2 \in \mathbb{R}$ are the biases. This leads to a total of 22 free parameters to learn. Finally, $\gamma : \mathbb{R} \to [0, 1]$ is a non-linear activation function.

$$\texttt{SNN}(d) = \boldsymbol{w}_2\gamma\Big(\boldsymbol{w}_1\boldsymbol{x}_d + \boldsymbol{b}_1\Big) + b_2 \tag{4-4}$$

The original work [9] relied on the sigmoid function for activation and used standard backpropagation for training. However, it is noteworthy that this architecture was designed in 1995, and consequently did not leverage extensive recent improvements of deep learning. Consequently, our preliminary experiments showed that this network performed poorly compared to the other baselines. To obtain a fair comparison leveraging newer training strategies: we decided to rely on `ReLU` activation functions (instead of sigmoid) and carry out training using Adam optimizer [51] with a learning rate of 10^{-3}. The SNN model is therefore trained for 50,000 epochs considering the mean square error as loss function.

Calibration and Training. To calibrate and train the different cost-estimation methods, we sampled for each instance 9,000 random connected districts respecting the size constraints $[n_{\text{L}}, n_{\text{U}}]$. For each of these districts, we calculated the expected TSP cost by SAA on the Train demand scenarios. For `GNN` and `SNN`, we further subdivided this set into 8,000 districts for training and 1,000 districts for validation which permit to control the convergence.

4.3 Tuning and Validation of ILS

In order to have a baseline of comparison to different settings of the `ILS`, we have solved some easier instances to optimality using the graph partitioning formulation presented in 2.2, namely, the instances with target district size equal to three and six. With the optimal results in hands, the ILS tuning consisted in finding the best value for the perturbation parameter p_{RM}. Preliminary

PUC-Rio - Certificação Digital Nº 2012383/CA

experiments showed that executing the instances of size 60 for 180 seconds, 90 for 600, and 120 for 1200 seconds is enough to obtain near-optimal results. Also, we have discovered that the objective function (Equation 3-5) was adding too much overhead to the construction and not improving the results. That way, we have decided to remove it and the formulation is used only to generate any feasible solution. The evaluated values for p_{RM} are 0.500, 0.250, 0.150, 0.100, 0.050, 0.025, 0.020, 0.015, 0.010 and 0.005.

		p_{RM}									
n	***t***	**0.500**	**0.250**	**0.150**	**0.100**	**0.050**	**0.025**	**0.020**	**0.015**	**0.010**	**0.005**
60	3	0.014	0.009	0.007	0.004	0.004	0.004	0.003	0.003	0.003	0.005
	6	0.023	0.014	0.012	0.009	0.009	0.007	0.007	0.008	0.006	0.009
90	3	0.040	0.030	0.023	0.020	0.015	0.014	0.014	0.013	0.013	0.014
	6	0.058	0.041	0.035	0.032	0.027	0.026	0.023	0.025	0.025	0.023
120	3	0.051	0.038	0.033	0.027	0.021	0.019	0.016	0.017	0.017	0.018
	6	0.082	0.062	0.052	0.045	0.043	0.037	0.042	0.044	0.040	0.043
Average		0.045	0.032	0.027	0.023	0.020	**0.018**	**0.018**	**0.018**	**0.018**	0.019

Table 4.3: Gap (%) for different values of p_{RM}

Table 4.3 has the results for some variations of p_{RM}. The gap column shows how far from the optimal cost the solution is. Each value contains the average result grouping every method, city, and depot positioning by instance size and target district size. Using lower values has improved quality until the limit of 0.025 when it starts to get stuck in the 0.018% gap until p_{RM} equals 0.005 when the quality starts to deteriorate. To decide which value of p_{RM} to take, the number of iterations is considered. Since the probability of a random move to be executed is p_{RM} when its value is lower fewer random moves are applied and the perturbation mechanism does not take the solution too far away from the local optimal, leading to less time running the local search. We selected the configuration $p_{\text{RM}} = 0.010$ on the remainder of the experiments as it led to the best performance.

It is important to evaluate if this ILS configuration is not biased to favor some of the cost estimators. Table 4.4 shows the gap and the number of iterations of each method with $p_{\text{RM}} = 0.010$. It is possible to see that the performances of `FIG`, `SNN`, and `BHHD` were pretty similar, but `GNN` has achieved a worse performance when compared to its optimal. This is expected as the `GNN` is more time consuming than other approaches, so it runs for far fewer iterations (about 7 times less).

PUC-Rio - Certificação Digital Nº 2012383/CA

		GNN		FIG		SNN		BHHD	
n	***t***	**Gap**	**Iter.**	**Gap**	**Iter.**	**Gap**	**Iter.**	**Gap**	**Iter.**
60	3	0.044	108241.6	0.004	533983.0	0.010	505233.6	0.010	542054.8
	6	0.209	8400.0	0.009	92962.5	0.011	76506.8	0.009	91046.4
90	3	0.123	86872.5	0.032	581387.5	0.033	552336.8	0.033	577662.5
	6	0.419	12556.6	0.055	108508.7	0.056	104712.5	0.059	114875.1
120	3	0.156	85246.2	0.045	683166.6	0.042	718791.6	0.045	738281.1
	6	0.457	14009.2	0.102	186465.3	0.093	180756.0	0.100	204835.0
Average		0.235	52554.34	0.041	364412.3	0.041	356389.5	0.043	378125.8

Table 4.4: Gap (%) and number of iterations for each each cost estimator

4.4 Results – Predictive performance of different cost-estimation models

Our first set of experiments aims to evaluate the accuracy of the different models (`BHHD`, `FIG`, `SNN`, as well as the proposed `GNN`) for estimating the routing costs. We therefore use the trained models described in the previous section, and evaluate them on an additional set of 1,000 random districts that are distinct from the ones used during training. We compare with a ground truth value for the routing costs, obtained again by SAA over the 50 Test demand scenarios. In the rest of this section, we analyze how the proposed estimation approaches deviate from the ground-truth measurements in terms of their root-mean-square error (RMSE – the lower, the better). For a given test instance, the RMSE is calculated as:

$$\text{RMSE} = \sqrt{\frac{1}{|\mathcal{D}|}\sum_{d\in\mathcal{D}}(\Phi(d) - \Phi^{\text{TEST}}_{\text{SAA}}(d))^2}, \tag{4-5}$$

where $\mathcal{D}$ is the set of 1,000 evaluation districts for this instance, and $\Phi(d)$ is the cost-estimation provided by the considered method on a district d.

Table 4.5 compares the RMSE of the different cost-estimation approaches. Each line corresponds to the results for the instances with a certain number of BUs (n) and district-size target (t), averaging the RMSE over the five different metropolitan areas and five possible depot-position configurations. The columns provide the characteristics of the instances, the average cost of the districts evaluated through SAA (indicating the magnitude of the target signal), and finally the RMSE of the different cost estimation approaches. Then, Table 4.6 focuses specifically on the medium-scale case of $n = 90$ and $t = 12$, with additional detailed the results for each metropolitan area and depot configuration.

As seen in these tables, with an overall RMSE of 2.96, the quality of the estimate obtained with the proposed `GNN` is vastly superior to that of all the other methods: `BHHD` with a RMSE of 4.86, `FIG` with a RMSE of 4.60, and `SNN`

PUC-Rio - Certificação Digital Nº 2012383/CA

n	t	$\hat{\Phi}^{\text{TEST}}_{\text{SAA}}$	RMSE of Estimation Method			
			GNN	BHHD	SNN	FIG
60	3	43.18 ± 1.47	**1.76**	2.33	2.10	2.17
	6	48.11 ± 1.97	**1.99**	3.57	3.24	3.41
	12	51.82 ± 2.42	**2.65**	4.49	4.14	4.27
	20	56.28 ± 2.95	**3.20**	5.55	5.04	5.25
	30	61.12 ± 3.22	**3.21**	5.09	4.69	4.85
90	3	52.40 ± 1.59	**1.93**	2.73	2.49	2.52
	6	56.92 ± 2.07	**2.40**	4.11	3.77	3.88
	12	60.93 ± 2.51	**3.23**	5.33	4.80	5.03
	20	66.64 ± 3.19	**3.89**	6.39	5.69	5.96
	30	72.48 ± 3.54	**3.75**	6.52	5.85	6.12
120	3	61.55 ± 1.81	**2.13**	3.32	3.15	3.20
	6	63.08 ± 2.06	**2.49**	4.14	3.90	4.00
	12	67.41 ± 2.57	**3.38**	5.47	5.03	5.19
	20	74.31 ± 3.30	**4.13**	6.59	6.09	6.25
	30	80.74 ± 3.77	**4.33**	7.21	6.65	6.89
Average		61.13 ± 2.56	**2.96**	4.86	4.44	4.60

Table 4.5: Accuracy of the different estimation approaches

PUC-Rio - Certificação Digital Nº 2012383/CA

Depot	**Met. Area**	$\hat{\Phi}^{\text{TEST}}_{\text{SAA}}$	RMSE of Estimation Method			
			GNN	BHHD	SNN	FIG
{C}	Bristol	66.81 ± 3.580	**4.84**	8.73	6.84	8.34
	Leeds	48.31 ± 2.877	**3.44**	5.92	5.40	5.48
	London	28.54 ± 1.862	**2.15**	3.46	3.28	3.34
	Manchester	42.77 ± 2.663	**3.07**	4.79	4.72	4.73
	West-Midlands	40.21 ± 2.445	**2.78**	4.27	4.09	4.08
{NE, NW, SE, SW}	Bristol	105.30 ± 3.419	**4.91**	9.28	7.33	8.43
	Leeds	71.40 ± 2.559	**3.41**	5.83	5.36	5.28
	London	36.99 ± 1.696	**2.11**	3.13	3.05	3.05
	Manchester	57.36 ± 2.413	**2.97**	4.45	4.41	4.39
	West-Midlands	53.10 ± 2.268	**2.71**	3.83	3.79	3.78

Table 4.6: Impact of the depot location and metropolitan area, for $\boldsymbol{n = 90}$ and $\boldsymbol{t = 12}$

with a RMSE of 4.44.

As seen in Table 4.5, the error committed increases with the number of BUs in the metropolitan area (n) as well as the target number of BUs (t) in each district. This is due to two factors. Firstly, the target signal (long-term operational cost of the districts) grows with t, and also with n to a lesser extent (instances with larger n include less-populated BUs that are located farther away from city centers). Therefore, as the target values grow larger, more estimation error is generally committed. Secondly, the number of possible districts grows exponentially in n and t, such that the universe of possible inputs grows, and it becomes harder to learn the target.

Considering the results of Table 4.6, we again notice that `GNN` estimations are far more accurate than the other approaches, for all metropolitan areas and depot configurations (close or away from the city center). Once again, we observe that the magnitude of the RMSE depends on the area. Indeed, metropolitan areas such as Bristol have a lower population density, and therefore longer tours within the districts, leading to generally higher operational-cost values. In this situation, it is natural for the error to be grow with the signal value reported in column $\hat{\Phi}^{\text{TEST}}_{\text{SAA}}$.

PUC-Rio - Certificação Digital Nº 2012383/CA

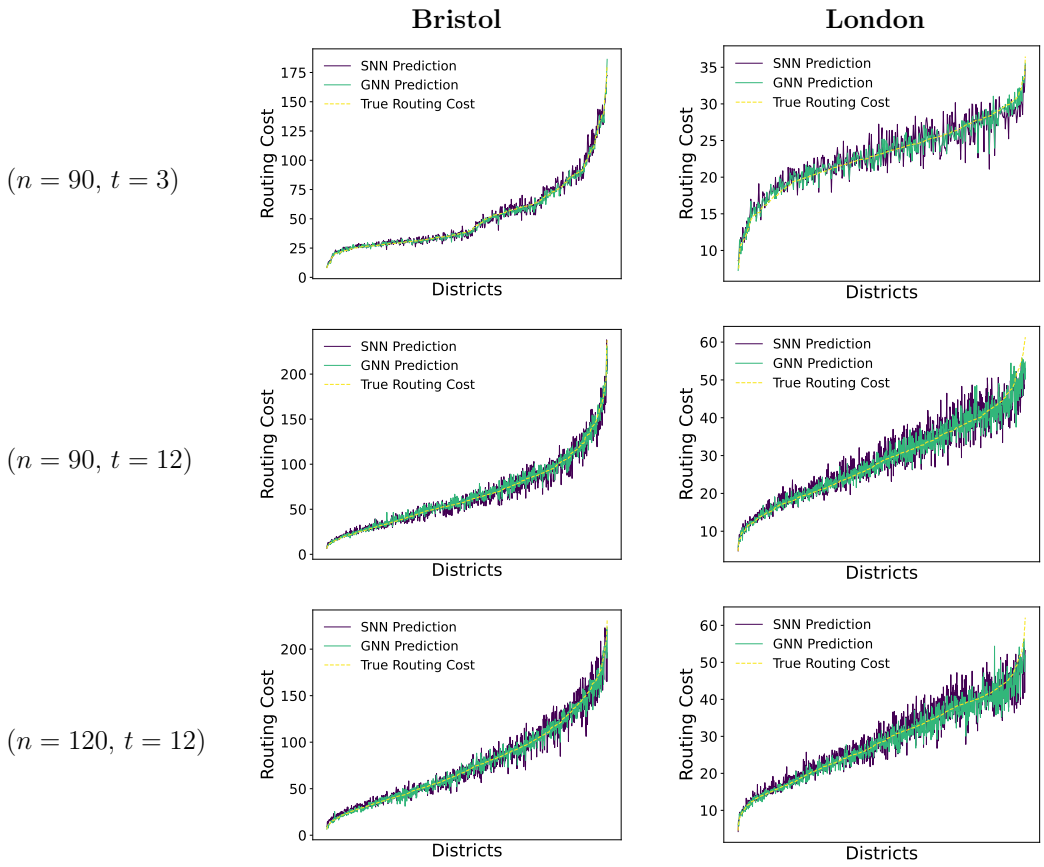


Figure 4.1: Visualization of the ground-truth and estimated district costs on a subset of districts

Finally, Figure 4.1 provides graphical representations of the ground-truth costs computed by SAA. The dotted curve corresponds to the true routing cost (ground truth), the blue continuous blue corresponds to `SNN` prediction, and the green continuous curve corresponds to the `GNN`. To facilitate the visualization, we display the set of test districts, ordered by increased ground-truth cost. The figure displays these graphs for two different metropolitan areas (Bristol and London), considering centered depots and three possible configurations for the n and t factors.

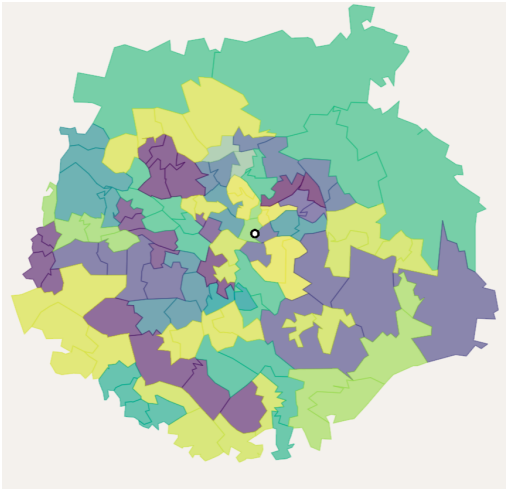
Comparing the figures with $t = 3$ and the figures with $t = 12$, it is possible to see that with a lower value for t the errors are much smaller, i. e. the lines of `GNN` and `SNN` are much closer to the true cost. A higher value for n also worsens the predictions, but its effect is less penalizing than changing the district sizes.

4.5 Results – Impact of District Cost Estimation Methods on Strategic Districting Decisions

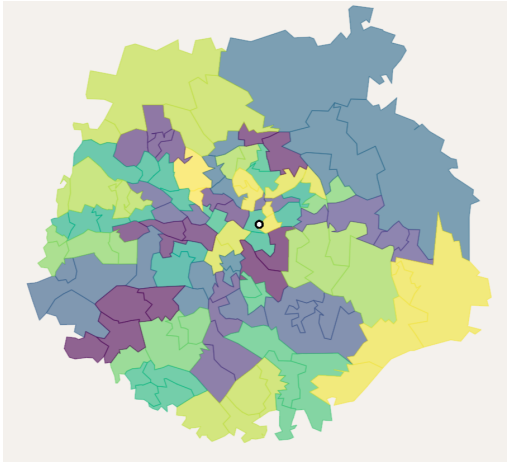
In this section, we want to evaluate the quality of the districting plan when designed using the different cost estimator methods. `BHHD-ILS`, `FIG-ILS`, `SNN-ILS`, and `GNN-ILS` were executed for the same amount of time for every instance. As in Section 4.3, instances with size 60 were executed for 180 seconds, 90 for 600 and 120 for 1200.

Table 4.7 shows the gaps from each method when compared to `GNN-ILS` solutions. Solution costs were calculated using SAA only considering the test scenarios. The results for each of the non-central depot positions were pretty similar, so we decided to group them. Each data point contains the average value for instance size n, target district size t, and depot centrality. For instance, when $n = 90$, $t = 12$ and depot is central, `BHHD-ILS` solutions have a 12.78% higher cost than the ones obtained by `GNN-ILS`, `FIG-ILS` has a 15.50% higher cost, and `SNN-ILS` 11.02%. The `GNN-ILS` was able to produce better solutions in every scenario. It is important to notice the impact that some instance characteristics have on the result. The most relevant difference appears when looking for different target district size. When the districts are small, the gains of `GNN-ILS` were also smaller. Varying from 2.26% to 22.08% when compared to `BHHD-ILS` with an instance size of 120. Also, the centrality of the depot plays an important role, when it is central the `GNN-ILS` is able to have a larger gap. When compared to `FIG-ILS` it varies from 13.12% of difference when it is central to 9.87% when it is not. As `BHHD-ILS`, `FIG-ILS`, and `SNN-ILS` achieved pretty similar results, the following analysis will only compare `GNN-ILS` to `SNN-ILS` since its results are slightly better.

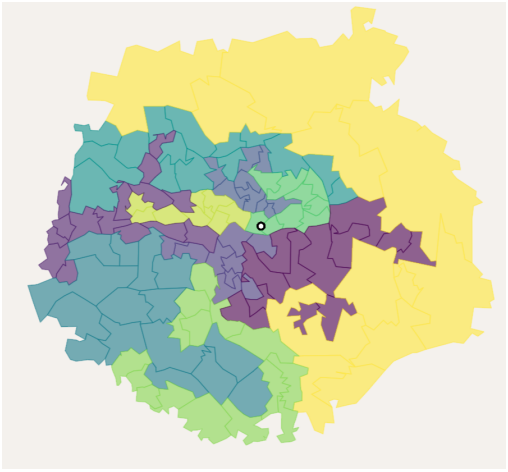
PUC-Rio - Certificação Digital Nº 2012383/CA



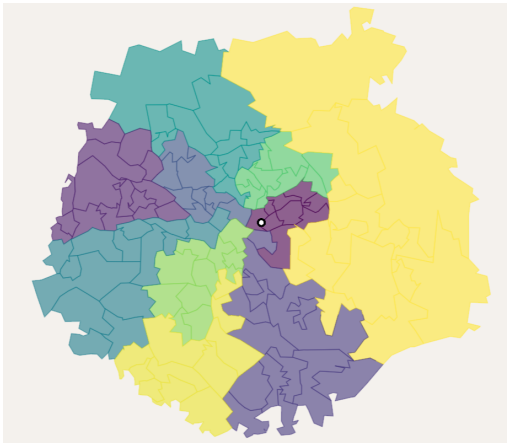
4.2(a): Leeds_C_120_3 `SNN-ILS` solution



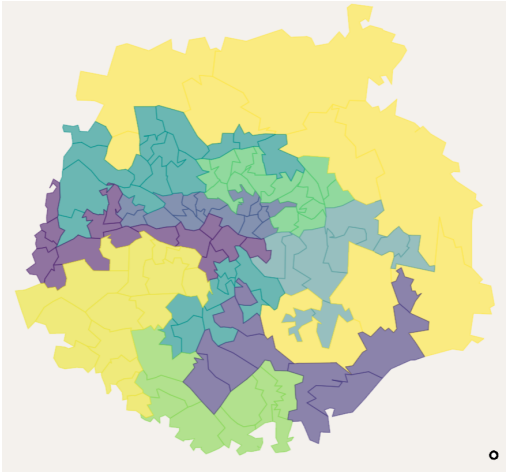
4.2(b): Leeds_C_120_3 `GNN-ILS` solution



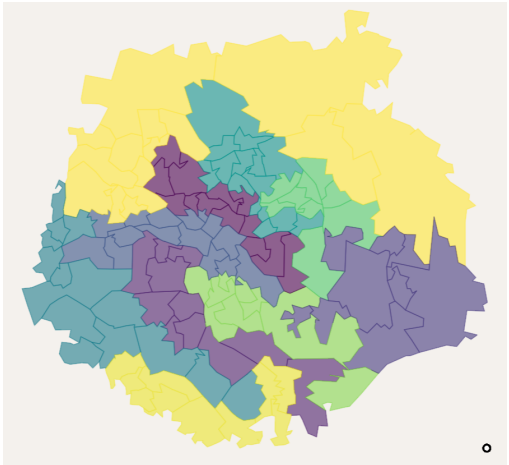
4.2(c): Leeds_C_120_12 `SNN-ILS` solution



4.2(d): Leeds_C_120_12 `GNN-ILS` solution



4.2(e): Leeds_SE_120_12 `SNN-ILS` solution



4.2(f): Leeds_SE_120_12 `GNN-ILS` solution

Figure 4.2: ILS solutions comparison

PUC-Rio - Certificação Digital Nº 2012383/CA

n	t	Central Depot			Not Central Depot		
		BHHD	**FIG**	**SNN**	**BHHD**	**FIG**	**SNN**
60	3	3.19	1.87	2.87	1.47	1.51	1.28
	6	9.13	11.41	8.97	6.63	6.76	6.89
	12	12.72	13.92	13.09	11.46	10.83	10.50
	20	14.48	14.88	12.49	11.91	12.41	12.45
	30	16.86	18.91	21.14	10.27	11.31	10.13
90	3	3.04	3.36	3.28	1.31	1.29	1.34
	6	8.83	8.65	9.17	6.69	5.84	6.04
	12	12.78	15.50	11.02	12.27	13.03	12.36
	20	19.91	20.06	16.78	16.17	16.62	14.55
	30	17.00	21.47	16.42	16.88	15.80	18.16
120	3	2.26	2.71	2.34	1.44	1.43	1.35
	6	7.41	6.46	7.55	5.39	5.56	5.21
	12	12.72	13.99	13.00	11.39	11.08	10.02
	20	22.08	20.95	22.99	15.41	16.48	13.87
	30	21.16	22.83	20.07	17.61	18.13	16.88
Average		12.21	13.13	12.08	9.76	9.87	9.40

Table 4.7: Relative difference from `BHHD-ILS`, `FIG-ILS` and `SNN-ILS` solutions while compared to `GNN-ILS`

Let's look at Figure 4.2 to have some intuition about the effects of instance characteristics. It shows six examples of generated solutions for different scenarios. Each basic unit is a colored area, districts are the basic units grouped by color and the depot position is represented by the white point. Comparing 4.2(a) and 4.2(b) we can see that `GNN` version was not able generate districts that looks more compact than `SNN-ILS`. Actually, as each district has fewer BUs, is harder to generate compact ones. Also, in this kind of scenario, it is necessary to travel to and from the depot, more times, being this round trip cost is a more relevant factor while evaluating the whole solution cost. Since `SNN` has a feature that considers the depot distance from each district, it can explicitly optimize it. Both factors contribute to the `GNN` not being able to do much better than the `SNN` in this of scenario. Figures 4.2(c) and 4.2(d) have a scenario where the `GNN` is able to do much better. Unlike the previous case, each district has a relevant amount of BUs, so `GNN` could build districts that look compact. Also, there are fewer round-trip costs from the depot to add up. Figures 4.2(e) and 4.2(f) have a scenario where the depot is in SE, it is possible

PUC-Rio - Certificação Digital Nº 2012383/CA

to see that the depot attracts the districts. Since it is farther from most BUs than when it is central, the depot travel costs again start to be a more relevant factor for the total cost. Even that, `GNN-ILS` was able to produce districts that look more compact.

Table 4.8 shows the gaps and the depot relevance from each different target district size and depot centrality. Depot relevance is the amount of the total costs that is represented by the round trip travel from the depot to each of the districts. Instances with a lower target district size have a larger number of small districts, that way, traveling inside a district tends to be cheaper and there are more travels to and from the depot. So the depot relevance is generally larger in this kind of instance. On the other hand, instances with a non-central depot tend to travel further distances to get to and from it. It explains why the depot relevance is lower when the depot is central. Is interesting to notice that in general the gains of `GNN-ILS` are higher when the depot is less relevant.

t	**Central Depot**		**Not Central Depot**	
	GAP	**Depot Relevance**	**GAP**	**Depot Relevance**
3	2.83	22.63	1.32	53.90
6	8.56	14.24	6.05	42.42
12	12.37	8.76	10.96	30.97
20	17.42	6.43	13.62	23.99
30	19.21	4.56	15.06	20.09

Table 4.8: Gap (%) from `SNN-ILS` to `GNN-ILS` and depot relevance (%) in total cost

PUC-Rio - Certificação Digital Nº 2012383/CA

4.6 Results – Compactness and Operational Efficiency of Districts

Table 4.9 shows the average compactness for solutions produced, a higher value meaning a more round-shaped district. It is possible to see that the ones by `GNN-ILS` have higher compactness in most of the scenarios, except when the depot is not central and the target district size is 3, 6, and 12. There is a clear trend while looking for the target district size, when it is higher is possible to design more compact districts. Also when the depot is central the districts are more compact.

Since it is commonly assumed that compactness is a good proxy for district costs we wanted to check if this holds in our experiment using the compactness formula detailed in Section 4.1. Table 4.10 shows the intra-district travel gains

t	Central Depot		Not Central Depot	
	GNN	**SNN**	**GNN**	**SNN**
3	**0.381**	0.373	0.367	**0.368**
6	**0.385**	0.342	0.322	**0.338**
12	**0.393**	0.309	0.293	**0.313**
20	**0.397**	0.313	**0.337**	0.326
30	**0.408**	0.349	**0.375**	0.355

Table 4.9: `GNN-ILS` vs `SNN-ILS` solution average compactness

t	Central Depot		Not Central Depot	
	Intra-District Cost Gains (%)	**Compactness Gains (%)**	**Intra-District Cost Gains (%)**	**Compactness Gains (%)**
3	1.72	1.89	1.51	-0.12
6	5.83	11.29	4.96	-4.86
12	9.67	21.31	7.90	-6.73
20	16.03	21.00	11.53	3.25
30	16.01	14.64	13.08	5.32

Table 4.10: Compactness gains compared to intra-district travel gains

from `GNN-ILS` solutions to `SNN-ILS`, i.e. the traveling costs ignoring the depot round trip, and their compactness relative difference. A negative value means that the `SNN-ILS` solution is better than `GNN-ILS` in a given criterion. There is a correlation between intra-district cost gains and compactness while looking at cases where the depot is central, at least until the target district size of 20. When looking at instances with a non-central depot things become different, even when `SNN-ILS` solutions are more compact still the `GNN-ILS` solutions have a lower intra-district cost. And this difference gets higher when looking from t equals from 3 to 6 or from 6 to 12. So this formula does not seem to explain very well the intra-district traveling cost with out-centered depots. Either way, it is interesting to notice that `GNN-ILS` has learned when is good to have a more round-shaped district and when it is not.

Similar behavior with out-centered depots can be observed when analyzing other routing problems, like the Capacitated Vehicle Routing Problem (`CVRP`). Figure 4.3 has the optimal solution for the `CVRP` instance X-n120-k6[1] [58]. It exhibits a *fish-scale* structure like in the solution of `GNN-ILS` represented in Figure 4.2(f).

[1]Solution can be found in: `http://vrp.galgos.inf.puc-rio.br/index.php/en/`

PUC-Rio - Certificação Digital Nº 2012383/CA

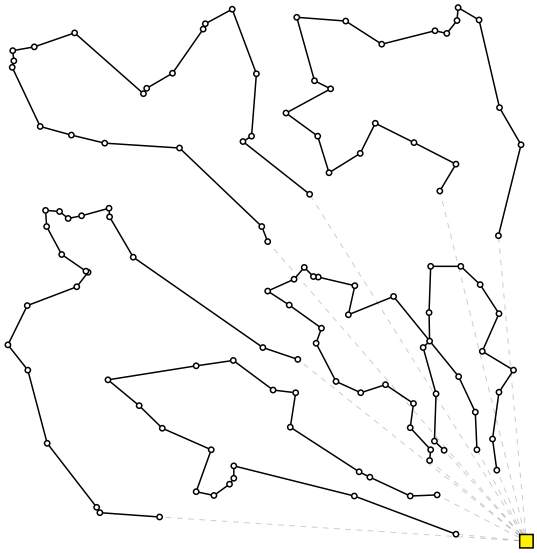


Figure 4.3: Optimal solution for Instance X-n120-k6 of CVRP

PUC-Rio - Certificação Digital Nº 2012383/CA

5
Conclusions

Districting is the process of partitioning a service region into larger clusters called districts. This practice is ubiquitous in large-scale transportation and has many industrial applications. Besides its NP-hard nature, districting decisions must hold in the long-term and optimizing them require to take into consideration routing costs of daily operations that are subject to volatile demands. The evaluation of the routing costs typically involve the solving of large-scale routing problems, which is time-consuming and sensitive to the spatial distribution of the requests. In this paper, we proposed to solve districting problems with an iterated local search that leverages a graph neural networks for estimating TSP routing costs inside the objective function. The network is trained in a supervised manner thanks to TSP costs that are pre-computed with Lin–Kernighan algorithm for many districting configurations.

Experiments have been carried out on five important metropolitan areas in the United Kingdoms. The `GNN` was able to successfully predict the long-plan horizon routing costs for a set of areas with an average error smaller than other traditional approaches for this problem, 8.43 of MSE compared to 32.13 for `BD`, 23.63 for `SNN`, and 25.94 for `FIG`. Even though running the `GNN` prediction is slower than solving the other formulas, `GNN-ILS` was able to produce better districts within the same amount of time, leading to gains around 10%, up to more than 20% in some scenarios, when compared to more traditional approaches. It also proved robust enough to get good results in a set of different scenarios. Instances with a higher depot relevance are harder for the `GNN-ILS` to improve, due to the fact that other formulations have an explicit factor to consider the depot round trip traveling cost. Finally, it is interesting to notice that even though no information about compactness was given for the GNN, it learned when to produce more round shaped districts.

Although being used for estimating routing costs related to TSPs, it is worthy to note that this approach could be considered for estimating the cost of other sub-problems, such as vehicle routing problems. This adaptation to other realistic situation is part of our future works. Currently, the learning part is intended to minimize a prediction error and do not account for how the predictions will be used in the subsequent optimization problem. This

PUC-Rio - Certificação Digital Nº 2012383/CA

kind of approach is commonly referred to as *predict-then-optimize*. Another line of research would be to consider an approach closer to *smart predict-and-optimize* [59], leveraging the optimization problem structure for designing better prediction models.

PUC-Rio - Certificação Digital Nº 2012383/CA

Bibliography

[1] BRUNO, G.; CAVOLA, M.; DIGLIO, A.; LAPORTE, G. ; PICCOLO, C.. **Reorganizing postal collection operations in urban areas as a result of declining mail volumes–A case study in Bologna**. Journal of the Operational Research Society, 72(7):1591–1606, 2021.

[2] BENZARTI, E.; SAHIN, E. ; DALLERY, Y.. **Operations management applied to home care services: Analysis of the districting problem**. Decision Support Systems, 55(2):587–598, 2013.

[3] GARCÍA-AYALA, G.; GONZÁLEZ-VELARDE, J. L.; RÍOS-MERCADO, R. Z. ; FERNÁNDEZ, E.. **A novel model for arc territory design: Promoting Eulerian districts**. International Transactions in Operational Research, 23(3):433–458, 2016.

[4] ZHONG, H.; HALL, R. W. ; DESSOUKY, M.. **Territory planning and vehicle dispatching with driver learning**. Transportation Science, 41(1):74–89, 2007.

[5] KOVACS, A.; GOLDEN, B.; HARTL, R. ; PARRAGH, S.. **Vehicle routing problems in which consistency considerations are important: A survey**. Networks, 64(3):192–213, 2014.

[6] DREXL, M.; SCHNEIDER, M.. **A survey of variants and extensions of the location-routing problem**. European Journal of Operational Research, 241(2):283–308, 2015.

[7] KALCSICS, J.; RÍOS-MERCADO, R. Z.. **Districting Problems**, p. 705–743. Springer International Publishing, Cham, 2019.

[8] FRANCESCHETTI, A.; JABALI, O. ; LAPORTE, G.. **Continuous approximation models in freight distribution management**. TOP, 25(3):413–433, 2017.

[9] KWON, O.; GOLDEN, B. ; WASIL, E.. **Estimating the length of the optimal TSP tour: An empirical study using regression and neural networks**. Computers & Operations Research, 22(10):1039–1046, 1995.

PUC-Rio - Certificação Digital Nº 2012383/CA

[10] LECUN, Y.; BENGIO, Y. ; HINTON, G.. **Deep learning**. nature, 521(7553):436–444, 2015.

[11] KIPF, T. N.; WELLING, M.. **Semi-supervised classification with graph convolutional networks**. arXiv preprint arXiv:1609.02907, 2016.

[12] LIN, S.; KERNIGHAN, B. W.. **An effective heuristic algorithm for the traveling-salesman problem**. Operations Research, 21(2):498–516, apr 1973.

[13] VERWEIJ, B.; AHMED, S.; KLEYWEGT, A. J.; NEMHAUSER, G. ; SHAPIRO, A.. **The sample average approximation method applied to stochastic routing problems: A computational study**. Computational Optimization and Applications, 24(2–3):289–333, feb 2003.

[14] HORN, D. L.; HAMPTON, C. R. ; VANDENBERG, A. J.. **Practical application of district compactness**. Political Geography, 12(2):103–120, 1993.

[15] BOZKAYA, B.; ERKUT, E. ; LAPORTE, G.. **A tabu search heuristic and adaptive memory procedure for political districting**. European Journal of Operational Research, 144(1):12–26, 2003.

[16] WEBSTER, G. R.. **Reflections on current criteria to evaluate redistricting plans**. Political Geography, 32(1):3–14, 2013.

[17] ZOLTNERS, A. A.; SINHA, P.. **Sales territory design: Thirty years of modeling and implementation**. Marketing Science, 24(3):313–331, 2005.

[18] LEI, H.; LAPORTE, G.; LIU, Y. ; ZHANG, T.. **Dynamic design of sales territories**. Computers & Operations Research, 56:84–92, 2015.

[19] NOVAES, A.; DE CURSI, J. ; GRACIOLLI, O.. **A continuous approach to the design of physical distribution systems**. Computers & Operations Research, 27(9):877–893, 2000.

[20] GALVÃO, L. C.; NOVAES, A. G.; SOUZA DE CURSI, J. E. ; SOUZA, J. C.. **A multiplicatively-weighted Voronoi diagram approach to logistics districting**. Computers and Operations Research, 33(1):93–114, 2006.

[21] CARLSSON, J.. **Dividing a territory among several vehicles**. INFORMS Journal on Computing, 24(4):565–577, 2012.

PUC-Rio - Certificação Digital Nº 2012383/CA

[22] ANILY, S.; FEDERGRUEN, A.. **One-Warehouse Multiple Retailer Systems with Vehicle Routing Costs**. Management Science, 36(1):92–114, 1990.

[23] KLEYWEGT, A.; SHAPIRO, A. ; HOMEM-DE-MELLO, T.. **The sample average approximation method for stochastic discrete optimization**. SIAM Journal on Optimization, 12(2):479–502, 2002.

[24] BEARDWOOD, J.; HALTON, J. H. ; HAMMERSLEY, J. M.. **The shortest path through many points**. Mathematical Proceedings of the Cambridge Philosophical Society, 55(9):299–327, 1959.

[25] DAGANZO, C. F.. **The distance traveled to visit n points with a maximum of c stops per vehicle: An analytic model and an application**. Transportation science, 18(4):331–350, 1984.

[26] CHIEN, T. W.. **Operational estimators for the length of a traveling salesman tour**. Computers and Operations Research, 19(6):469–478, 1992.

[27] ROSENFIELD, D.; ENGELSTEIN, I. ; FEIGENBAUM, D.. **An application of sizing service territories**. European Journal of Operational Research, 63(2):164–172, 1992.

[28] NOVAES, A.; GRACIOLLI, O.. **Designing multi-vehicle delivery tours in a grid-cell format**. European Journal of Operational Research, 119(3):613–634, 1999.

[29] LEI, H.; LAPORTE, G. ; GUO, B.. **Districting for routing with stochastic customers**. EURO Journal on Transportation and Logistics, 1(1):67–85, 2012.

[30] FIGLIOZZI, M. A.. **Analysis of the efficiency of urban commercial vehicle tours: Data collection, methodology, and policy implications**. Transportation Research Part B: Methodological, 41(9):1014–1032, 2007. Behavioural insights into the Modelling of Freight Transportation and Distribution Systems.

[31] NICOLA, D.; VETSCHERA, R. ; DRAGOMIR, A.. **Total distance approximations for routing solutions**. Computers & Operations Research, 102:67–74, 2019.

[32] AKKERMAN, F.; MES, M.. **Distance approximation to support customer selection in vehicle routing problems**. Annals of Operations Research, 2022.

PUC-Rio - Certificação Digital Nº 2012383/CA

[33] KRIZHEVSKY, A.; SUTSKEVER, I. ; HINTON, G. E.. **Imagenet classification with deep convolutional neural networks**. Advances in neural information processing systems, 25:1097–1105, 2012.

[34] VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, L. ; POLOSUKHIN, I.. **Attention is all you need**. arXiv preprint arXiv:1706.03762, 2017.

[35] BELLO, I.; PHAM, H.; LE, Q. V.; NOROUZI, M. ; BENGIO, S.. **Neural combinatorial optimization with reinforcement learning**. arXiv preprint arXiv:1611.09940, 2016.

[36] DEUDON, M.; COURNUT, P.; LACOSTE, A.; ADULYASAK, Y. ; ROUSSEAU, L.-M.. **Learning heuristics for the tsp by policy gradient**. In: van Hoeve, W.-J., editor, INTERNATIONAL CONFERENCE ON THE INTEGRATION OF CONSTRAINT PROGRAMMING, ARTIFICIAL INTELLIGENCE, AND OPERATIONS RESEARCH, p. 170–181, Cham, 2018. Springer, Springer International Publishing.

[37] BAHDANAU, D.; CHO, K. ; BENGIO, Y.. **Neural machine translation by jointly learning to align and translate**. arXiv preprint arXiv:1409.0473, 2014.

[38] KOOL, W.; VAN HOOF, H. ; WELLING, M.. **Attention, learn to solve routing problems!** arXiv preprint arXiv:1803.08475, 2018.

[39] SCARSELLI, F.; GORI, M.; TSOI, A. C.; HAGENBUCHNER, M. ; MONFARDINI, G.. **The graph neural network model**. IEEE transactions on neural networks, 20(1):61–80, 2008.

[40] XIE, Z.; LV, W.; HUANG, S.; LU, Z.; DU, B. ; HUANG, R.. **Sequential graph neural network for urban road traffic speed prediction**. IEEE Access, 8:63349–63358, 2019.

[41] RUSEK, K.; SUÁREZ-VARELA, J.; MESTRES, A.; BARLET-ROS, P. ; CABELLOS-APARICIO, A.. **Unveiling the potential of graph neural networks for network modeling and optimization in sdn**. In: PROCEEDINGS OF THE 2019 ACM SYMPOSIUM ON SDN RESEARCH, SOSR '19, p. 140–151, New York, NY, USA, 2019. Association for Computing Machinery.

[42] DERROW-PINION, A.; SHE, J.; WONG, D.; LANGE, O.; HESTER, T.; PEREZ, L.; NUNKESSER, M.; LEE, S.; GUO, X.; WILTSHIRE, B. ; OTHERS.

PUC-Rio - Certificação Digital Nº 2012383/CA

Eta prediction with graph neural networks in google maps. In: PROCEEDINGS OF THE 30TH ACM INTERNATIONAL CONFERENCE ON INFORMATION & KNOWLEDGE MANAGEMENT, p. 3767–3776, New York, NY, USA, 2021. Association for Computing Machinery.

[43] KAFAEI, P.; CAPPART, Q.; RENAUD, M.-A.; CHAPADOS, N. ; ROUSSEAU, L.-M.. **Graph neural networks and deep reinforcement learning for simultaneous beam orientation and trajectory optimization of cyberknife**. Physics in Medicine & Biology, 66(21):215002, 2021.

[44] JOSHI, C. K.; LAURENT, T. ; BRESSON, X.. **An efficient graph convolutional network technique for the travelling salesman problem**. arXiv preprint arXiv:1906.01227, 2019.

[45] CAPPART, Q.; CHÉTELAT, D.; KHALIL, E.; LODI, A.; MORRIS, C. ; VELIČKOVIĆ, P.. **Combinatorial optimization and reasoning with graph neural networks**. arXiv preprint arXiv:2102.09544, 2021.

[46] DAI, H.; DAI, B. ; SONG, L.. **Discriminative embeddings of latent variable models for structured data**. In: Balcan, M. F.; Weinberger, K. Q., editors, PROCEEDINGS OF THE 33RD INTERNATIONAL CONFERENCE ON MACHINE LEARNING, volumen 48 de **Proceedings of Machine Learning Research**, p. 2702–2711, New York, New York, USA, 20–22 Jun 2016. PMLR.

[47] VELIČKOVIĆ, P.; CUCURULL, G.; CASANOVA, A.; ROMERO, A.; LIÒ, P. ; BENGIO, Y.. **Graph attention networks**. arXiv preprint arXiv:1710.10903, 2018.

[48] HAMILTON, W. L.; YING, R. ; LESKOVEC, J.. **Inductive representation learning on large graphs**. In: PROCEEDINGS OF THE 31ST INTERNATIONAL CONFERENCE ON NEURAL INFORMATION PROCESSING SYSTEMS, NIPS'17, p. 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc.

[49] KHALIL, E.; DAI, H.; ZHANG, Y.; DILKINA, B. ; SONG, L.. **Learning combinatorial optimization algorithms over graphs**. Advances in Neural Information Processing Systems, 30, 2017.

[50] GLOROT, X.; BORDES, A. ; BENGIO, Y.. **Deep sparse rectifier neural networks**. In: Gordon, G.; Dunson, D. ; Dudík, M., editors, PROCEEDINGS OF THE FOURTEENTH INTERNATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE AND STATISTICS, volumen 15 de **Proceedings of**

PUC-Rio - Certificação Digital Nº 2012383/CA

Machine Learning Research, p. 315–323, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR.

[51] KINGMA, D. P.; BA, J.. **Adam: A method for stochastic optimization**. arXiv preprint arXiv:1412.6980, 2014.

[52] LOURENÇO, H. R.; MARTIN, O. C. ; STÜTZLE, T.. **Iterated Local Search: Framework and Applications**, p. 363–397. Springer US, Boston, MA, 2010.

[53] MIYAZAWA, F. K.; MOURA, P. F. S.; OTA, M. J. ; WAKABAYASHI, Y.. **Cut and flow formulations for the balanced connected k-partition problem**. In: Baïou, M.; Gendron, B.; Günlük, O. ; Mahjoub, A. R., editors, COMBINATORIAL OPTIMIZATION, p. 128–139, Cham, 2020. Springer International Publishing.

[54] PARK, N.. **Middle super output area population estimates - mid-2018: Sape21dt3a edition**. `https://www.ons.gov.uk/peoplepopulationandcommunity/populationandmigration/populationestimates/datasets/middlesuperoutputareamidyearpopulationestimates`, 2018. Accessed in November 27, 2021.

[55] ERNEST, C. R.. **Measuring compactness as a requirement of legislative apportionment, 5 midwest j**. Pol. Sci, 70, 1961.

[56] YOUNG, H. P.. **Measuring the compactness of legislative districts**. Legislative Studies Quarterly, 13(1):105, Feb 1988.

[57] HELSGAUN, K.. **An effective implementation of the Lin-Kernighan traveling salesman heuristic**. European Journal of Operational Research, 126(1):106–130, 2000.

[58] UCHOA, E.; PECIN, D.; PESSOA, A.; POGGI, M.; VIDAL, T. ; SUBRAMANIAN, A.. **New benchmark instances for the capacitated vehicle routing problem**. European Journal of Operational Research, 257(3):845–858, 2017.

[59] ELMACHTOUB, A. N.; GRIGAS, P.. **Smart "predict, then optimize"**. Management Science, 68(1):9–26, 2022.

PUC-Rio - Certificação Digital Nº 2012383/CA