Optimizing warehouse locations for E-commerce using geospatial clustering

Vaibhav Mukundan
Department of Computer Science
Western University
London, Canada
vmukund@uwo.ca

Dipesh Gyanchandani
Department of Computer Science
Western University
London, Canada
dgyancha@uwo.ca

Raj Tulluri
Department of Computer Science
Western University
London, Canada
rtulluri@uwo.ca

***Abstract*—In the dynamically evolving landscape of e-commerce, efficient warehouse location optimization has become a cornerstone for operational efficiency, directly impacting delivery speed, cost-effectiveness, and customer satisfaction. This study introduces a novel approach to warehouse location optimization, leveraging advanced spatial clustering and optimization techniques to address the logistical challenges faced by e-commerce businesses. Utilizing a comprehensive dataset encompassing geolocation data from customers, the study employs density-based spatial clustering to identify strategic hotspots of high demand. Subsequently, an optimization model, integrating travel distances and durations derived from the Google Maps API, is applied to these clusters to pinpoint ideal warehouse locations. This model uniquely incorporates weight parameters for each distance metric, optimized using Random optimization (RO) techniques, to reflect the varying significance of different logistical factors across diverse geographic and demographic landscapes. The findings of this research underscore the critical role of strategically placed warehouses in reducing delivery times and costs, thereby enhancing the overall efficiency and competitiveness of e-commerce businesses. Beyond e-commerce, the methodologies and insights presented here are poised to transform warehousing strategies across various sectors, offering a data-driven blueprint for optimizing distribution networks in the digital age.**

***Index Terms*—Clustering Algorithms, E-Commerce, Optimization Techniques, Logistics**

I. Introduction

The optimization of warehouse locations is a central concern in contemporary e-commerce logistics, where the efficient distribution of goods directly influences delivery times, operational costs, and customer satisfaction. This study addresses the critical problem of strategically placing warehouses to minimize the distance between distribution centers and customers, thereby enhancing the overall efficacy of the supply chain. As the e-commerce sector undergoes unprecedented expansion, the need for intelligent and data-driven warehouse placement methods becomes ever more crucial.

The importance of this study is highlighted by the critical role that optimum location of warehouses play in meeting customers' increasing expectations for timely and dependable deliveries. Efficient warehouse location strategies are not only vital for the competitiveness and profitability of e-commerce enterprises but also have broader implications for industries grappling with spatial optimization challenges. The contributions are not only for the immediate concerns of e-commerce logistics but also to offer insights applicable to a spectrum of sectors, extending the reach of spatial optimization strategies.

A local warehouse significantly enhances e-commerce efficiency by shortening the distance to customers, enabling rapid order fulfillment. This strategic location reduces last-mile delivery time and lowers transportation costs, while also simplifying long-distance shipping challenges. By being closer to customers, it facilitates targeted inventory management and recommendation strategies, tailored to regional needs. Thus, a local warehouse is crucial in optimizing inventory and ensuring swift, customer-centric service.

The initial approach involves deploying Density-Based Spatial Clustering of Applications with Noise (DBSCAN) and Hierarchical Density-Based Spatial Clustering of Applications with Noise (HDBSCAN) as baseline models to map out approximate warehouse locations. DBSCAN provides a solid basis for improving warehouse location by focusing on regions of high data density. HDBSCAN extends DBSCAN's capabilities by adding a hierarchical dimension to the clustering process. This suggests that it can recognize clusters with varying degrees of compactness and discriminate between areas with high and low data density. Together these methods, provide an initial spatial model, akin to a rough sketch, laying the groundwork for subsequent optimization efforts in the quest for strategically positioning the warehouse.

The utilization of DBSCAN and HDBSCAN in tandem offers a robust initial spatial model, lays the groundwork for the subsequent optimization phase, where we delve deeper into refining these locations to align with operational efficiency and logistical feasibility. The optimization phase involves feature engineering using the Google Maps API for calculating travel distances and durations, and Random Optimzation (RO) techniques for fine-tuning the warehouse positions based on a variety of logistical parameters. This multi-faceted approach, starting with DBSCAN and HDBSCAN and seamlessly transitioning to optimization algorithms, forms the core of the methodology in strategically positioning warehouses to serve the e-commerce market effectively.

II. Related Work

The evolving landscape of geospatial data analysis has been significantly shaped by advancements in clustering techniques, particularly in identifying high-density areas or 'hotspots' in various fields. This section reviews influential works that have contributed to this domain, highlighting their methodologies and contextualizing their relevance to the study's focus on optimizing warehouse locations in the e-commerce sector.

Zhang and Ukkusuri's paper (2021) [1] presents a pioneering approach to optimizing taxi group ride services through spatial-temporal clustering. Their research is centered around the optimization of taxi group ride services (TGRS) using extensive taxicab trip data in New York City. The core of their methodology involves a novel spatial-temporal clustering technique, which initially identifies potential group rides by exploring trip clusters. Subsequently, an agglomerative clustering method is applied to merge these trip clusters at both spatial and temporal scales, thereby determining efficient taxi stand locations and schedules. The study's ability to process large-scale data efficiently and its innovative use of clustering techniques to improve urban mobility solutions have been instrumental in inspiring the methodology. Particularly, the concept of identifying clusters based on spatial-temporal data parallels the approach to determining efficient warehouse locations in the e-commerce logistics chain.

Similarly, Li, Shi, and Zhang's research (2021) [2] delves into urban hotspot detection using taxi trajectory data. Their approach involves a two-phase clustering method, starting with an improved ST-HDBSCAN algorithm to form initial clusters based on spatial proximity. This is followed by a density filtering process within the road network space, refining the accuracy of the identified hotspots. Notably, their work demonstrates the practicality of clustering techniques in real-world settings, emphasizing the importance of accurately capturing high-activity areas.

The methodologies and findings of these two studies directly inform and inspire this research in the e-commerce logistics domain. In particular, the concept of spatial-temporal clustering from Zhang and Ukkusuri's [1] work offers a valuable perspective for analyzing customer order distributions in e-commerce. By identifying dense regions of customer orders, akin to their approach to identifying potential group rides, we aim to determine optimal warehouse locations that efficiently serve these high-demand areas. Similarly, the two-phase clustering process detailed by Li, Shi, and Zhang [2], particularly the use of road network data to refine clustering results, is instrumental to our study. In this context, This study leverages a similar concept to further refine the positioning of warehouses. By incorporating road network considerations, The aim is to optimize warehouse locations not only in terms of proximity to customer clusters but also regarding accessibility and efficiency in delivery logistics. This approach underscores the importance of integrating road network data into spatial clustering analyses, especially in operations where logistics and transportation play a crucial role.

While both these studies primarily focus on transportation and urban planning, the fundamental principles they employ are highly pertinent to the research. This paper extends the application of these geospatial clustering techniques to the domain of e-commerce logistics, a field that necessitates efficient and cost-effective warehouse placement strategies. The research differs from these earlier works in its application context but shares the core objective of utilizing clustering to enhance operational efficiency. We employ HDBSCAN, a density-based clustering algorithm, to analyze the distribution of customer orders, thereby identifying dense regions indicative of high logistical activity. These regions are treated as hotspots, guiding the determination of optimal warehouse locations. The contribution to the field of geospatial data analysis lies in this novel application of established clustering techniques to e-commerce logistics, a rapidly growing sector with significant operational challenges. By adapting methodologies from urban transportation studies, The aim is to provide a framework that not only addresses a specific problem in e-commerce logistics but also sets a precedent for the application of these techniques in similar distribution-based industries.

III. Geo-spatial Clustering Framework

The study addresses the challenge of optimizing warehouse locations for e-commerce logistics by introducing a method that combines spatial clustering techniques with optimization algorithms. This approach, based on thorough data analysis, seeks to derive practical solutions from complex geospatial data, ultimately aiming to improve operational efficiency and customer satisfaction in the e-commerce sector.

The experiment is divided into four stages.

- *Data sourcing*: The initial stage of the study involved procuring and preparing a geolocation dataset, sourced from Kaggle [3], which comprises both customer and seller locations. This dataset forms the backbone of the spatial analysis, essential for optimizing e-commerce warehouse locations.
- *Clustering*: In the subsequent phase of the methodology, this study ventures into spatial clustering, an approach pivotal for discerning patterns within geolocation data. This technique is essential to the study as it allows us to unravel the complex distribution of customer locations, a critical factor in determining the most efficient warehouse placements in the e-commerce landscape, and the commencement of the DBSCAN and HDBSCAN clustering algorithms. This density-based approach allows for a more nuanced understanding of customer location patterns.
- *Feature engineering*: An essential aspect of the methodology is the integration of road network data, obtained from the Google Maps API, into this analytical framework. This step is pivotal in augmenting the spatial clustering outcomes with real-world applicability.
- *Optimization*: The last and pivotal stage in methodology is to optimize warehouse locations by employing a suite

of advanced optimization algorithms, each contributing uniquely to the determination of the most strategically viable warehouse sites. The approach is to encompasses Particle Swarm Optimization (PSO), the Nelder-Mead algorithm, and Simulated Annealing, offering a multifaceted perspective on the optimization challenge. The comparative analysis, grounded in metrics such as time taken, number of iterations, and the cost reduction achieved, will guide the experiment in selecting the most effective algorithm for the chosen dataset.

A. Data sourcing & pre-processing

The Brazilian E-Commerce Public Dataset by Olist [3], containing order data from various Brazilian marketplaces between 2016 and 2018 was chosen for this experiment. The dataset is structured to provide insights across several dimensions of e-commerce transactions, from order details to customer locations and product reviews, including a geolocation dataset that assigns latitude and longitude to Brazilian zip codes.

To prepare the data for clustering analysis, a series of pre-processing steps were conducted. Records missing geolocation information were removed to ensure spatial analyses were based on accurate customer locations. Additionally, the dataset was deduplicated to retain only unique customer locations, thereby preventing the skewing of clustering outcomes due to repeated entries. This pre-processing is critical, as accurate spatial representation is fundamental for the effective application of DBSCAN and HDBSCAN clustering algorithms. These steps ensure that the analysis is performed on high-quality data.

Moreover, it should be noted that the latitude and longitude coordinates provided within the dataset adhere to the World Geodetic System 1984 (WGS 84) standard [4], which is the established global reference framework for geospatial measurement. The conformity of the dataset to the WGS 84 standard ensures that the geolocation data align with internationally recognized benchmarks, facilitating spatial analyses that are both accurate and consistent. This standardization is crucial for the integrity of the clustering approach.

B. Exploration of Clustering approaches

In the subsequent section, we delve into the exploration of clustering approaches, specifically focusing on DBSCAN [5] and HDBSCAN [6]. These algorithms are renowned for their robustness in identifying clusters of arbitrary shapes and sizes in spatial datasets. DBSCAN and HDBSCAN were selected for their capacity to handle the spatial complexity and density variation inherent in geographically dispersed e-commerce data. This section elucidates the operational principles of these algorithms, describes the methodology, and presents the results of the clustering analysis. This examination aims to discern the optimal clustering approach that aligns with the study's objectives.

DBSCAN is well-suited for data sets where the clusters are not necessarily spherical and where there may be noise or outliers. The core concept of DBSCAN revolves around the classification of data points into three types: core points, border points, and noise, as shown in algorithm 1. A core point has at least a minimum number of other points (MinPts) within a given radius (ϵ). A border point is within the ϵ radius of a core point but has fewer than MinPts within its ϵ neighborhood. Noise is any data point that is neither a core point nor a border point.

Algorithm 1 DBSCAN simplified algorithm

```
for each point P in dataset D do
    if |N_ε(P)| < MinPts then
        mark P as NOISE
    else
        mark P as CORE POINT
    end if
end for
for each CORE POINT P do
    Create a new cluster C
    for each point Q in N_ε(P) do
        if Q is a CORE POINT then
            add Q to cluster C
        end if
    end for
end for
for each BORDER POINT P do
    Assign P to a cluster of a CORE POINT in N_ε(P)
end for=0
```

The neighborhood function in DBSCAN, which returns all points within a certain radius ϵ of a given point p, can be expressed as the equation (1):

$$N_\epsilon(p) = \{q \in D \mid \text{dist}(p, q) \leq \epsilon\} \tag{1}$$

In this formula, $N_\epsilon(p)$ is the set of points q that are within the ϵ-neighborhood of p, where D is the dataset and dist(p,q) is the distance between two points p and q, which must be less than or equal to ϵ for q to be included in the neighborhood.

In this experiment, the DBSCAN algorithm was applied to the entire spectrum of customer locations obtained from the Olist dataset, with the aim of identifying dense regions indicative of potential warehouse placement. The algorithm was parameterized with an epsilon (eps) value of 6 and a minimum points (min_pts) threshold of 5. These parameters were chosen to discern the spatial clusters reflecting groups of customers in close proximity. The selection of these parameters, epsilon (eps) and minimum points (min_pts), was informed by a grid search technique. This systematic approach involved iterating over a range of values for eps and min_pts, evaluating the clustering outcome against predefined metrics.

The application of DBSCAN to the customer location data yielded distinct spatial clusters, particularly in densely populated urban areas, as observed in figure 1. The clusters

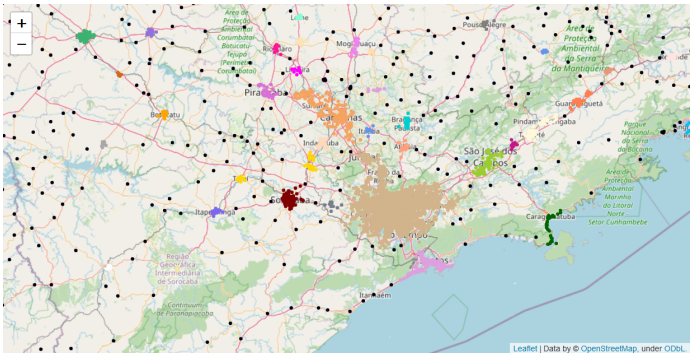

Fig. 1. DBSCAN clustering high density customer locations in urban areas

identified, represented by the colored points in the figure, suggest potential sites for warehouse placement, while the black points highlight regions that are less ideal due to sparsity or dispersion of customers and hence labelled as noise. The results obtained provide a valuable baseline for understanding the geographical distribution of customer orders.

After the exploration of DBSCAN, the next step is to examine HDBSCAN, a clustering algorithm that extends DBSCAN by converting it into a hierarchical clustering algorithm and then using a tree-based method to extract flat clusters based on the stability of clusters. HDBSCAN works by first transforming the space according to the density, similar to DBSCAN, but then builds a hierarchy of clusters and uses a novel technique to extract the clusters based on the stability of clusters over scale. This results in a set of clusters that are more stable over space and are less likely to be affected by noise.

HDBSCAN begins by computing the core distance for each point in the dataset. The core distance is the smallest value of (epsilon) such that the point is a core point with respect to and a set minimum number of points, MinPts. This core distance is then used to construct the mutual reachability distance, a measure that ensures that points within dense clusters are closer together, while points that are not in dense clusters are kept further apart. Following this, HDBSCAN builds a minimum spanning tree (MST) of the dataset using the mutual reachability distances. This tree is then condensed by pruning branches that do not satisfy the minimum cluster size criterion. Clusters are derived from this condensed tree by selecting the most stable clusters over the hierarchy. A cluster's stability is measured by its persistence over the data's density, which indicates the likelihood of a cluster to persist as an entity within the dataset.

In the algorithm 2, the key steps involve the calculation of core distances and the construction of the MST and the resulting dendrogram. The process of condensing the dendrogram and extracting clusters is based on the stability of the clusters, which is a measure of how persistently a set of points remains densely connected within the dendrogram. The core distance for a given point p is defined as the distance ϵ from p to its

Algorithm 2 HDBSCAN Algorithm

Require: Dataset D, Minimum cluster size $MinClustSize$

1: Compute the core distance for each point in D
2: Compute the mutual reachability distance for each pair of points in D
3: Construct the mutual reachability graph G using the mutual reachability distances
4: Compute the minimum spanning tree MST of G
5: Convert MST into a dendrogram D
6: Condense the dendrogram D by setting the minimum cluster size to $MinClustSize$
7: Extract clusters from the condensed dendrogram
8: **return** Clusters =0

MinPts-nearest neighbor, as shown in equation (2).

$$\text{core-dist}_{\epsilon,MinPts}(p) = \min\{\epsilon \mid |N_\epsilon(p)| \geq MinPts\} \quad (2)$$

The mutual reachability distance between two points p and q is defined as the maximum of their core distances and the actual distance between them, ensuring that points within the same cluster are mutually reachable. This is shown in equation (3).

$$\text{mutual-reach-dist}_{\epsilon,MinPts}(p,q) = \max \begin{cases} \text{core-dist}_{\epsilon,MinPts}(p), \\ \text{core-dist}_{\epsilon,MinPts}(q), \\ d(p,q) \end{cases} \quad (3)$$

HDBSCAN was deployed on the customer location data to discern clusters that indicate dense regions suitable for warehouse placement. The algorithm was configured with a minimum cluster size of 5, determined through a grid search to optimize the clustering outcome. Unlike DBSCAN, HDBSCAN does not require the specification of an epsilon parameter, thus simplifying the clustering process and allowing for the detection of clusters with varying densities.

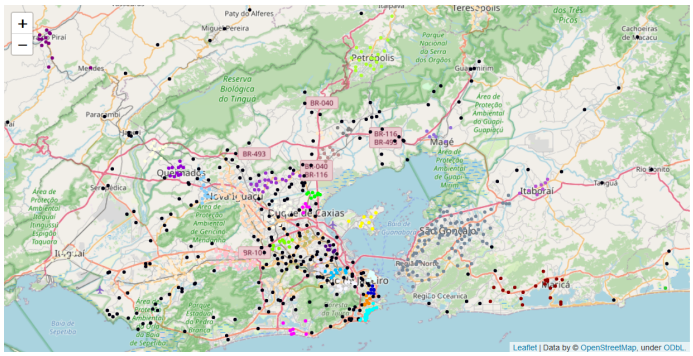

Fig. 2. HDBSCAN clustering dense regions within urban cities

The colored points in the figure 2 represent the clusters found by HDBSCAN. The visualization of clusters versus noise on the national map indicates a successful identification of dense regions where a significant number of customers are located. The granularity of clusters within urban centers, as

shown in the figure of specific regions, suggests that HDBSCAN is adept at identifying multiple potential warehouse locations within a city.

The Haversine distance is a key metric used to calculate the shortest distance between two points on the surface of a sphere given their longitudes and latitudes. This metric is particularly useful in geographical data analysis as it accounts for the curvature of the Earth. In the context of this paper, the Haversine distance formula was utilized as a metric within both the DBSCAN and HDBSCAN algorithms to accurately measure the distance between customer locations for clustering purposes.

$$\begin{aligned} a &= \sin^2\left(\frac{\Delta\phi}{2}\right) + \cos(\phi_1)\cdot\cos(\phi_2)\cdot\sin^2\left(\frac{\Delta\lambda}{2}\right) \\ c &= 2\cdot\arctan 2\left(\sqrt{a}, \sqrt{1-a}\right) \\ d &= R\cdot c \end{aligned} \tag{4}$$

- ϕ_1, ϕ_2 are the latitudes of points 1 and 2, respectively
- Δ_ϕ is the difference between the latitudes
- $\Delta\lambda$ is the difference between the longitudes
- R is the radius of the Earth (mean radius = 6,371 km)
- d is the distance between the two points

The equation (4) gives the calculation of haversine distance between two points, given the (latitude,longitude) pair for each point. Haversine distance was systematically applied in both clustering algorithms to ensure that the spatial analysis reflects the actual geographic distances, providing a more meaningful cluster formation that is essential for determining optimal warehouse locations.

C. Feature engineering

In the context of this research, three significant features derived from the geographical data of customers and an initial warehouse location, were engineered. These features are intended to capture the logistical nuances and constraints inherent in e-commerce distribution and are pivotal for the subsequent optimization of warehouse locations.

The first feature is the Haversine distance, as referenced before in equation (4), represents the shortest distance over the earth's surface between the customer's location and the warehouse, providing an 'as-the-crow-flies' approximation. The second and third features are derived using the Google Maps API, which is a robust platform offering detailed information about roads, pathways, and travel modes. The road distance feature obtained from the API provides the actual travelling distance by road. This feature is essential to estimate the logistical costs and time associated with road transport. Similarly, the road duration feature represents the time required to travel from the warehouse to the customer's location by road. This time is influenced by a myriad of factors, including road type, traffic conditions, speed limits, and other regulatory measures. Google Maps API estimates this duration by aggregating vast amounts of data, thereby providing an average duration that accounts for typical travel conditions.

The integration of the Google Maps API into feature engineering process involved sending requests with specific latitude and longitude coordinates and receiving structured data in return, which includes detailed attributes like distance and duration.

By incorporating travel distances and duration into the analysis and to go beyond mere geographic proximity to consider the practical aspects of accessibility and connectivity. These metrics are fundamental in refining clustering results, enabling us to assess not just the spatial closeness of customer locations, but also the logistical feasibility of reaching these points from potential warehouse sites.

D. Cost function optimization

The construction of a cost function serves as a cornerstone in the methodology to optimize warehouse locations. The cost function, denoted by J_i, is designed to evaluate the suitability of a potential warehouse location within a given cluster i by incorporating the features engineered in the previous step, i.e. haversine distance, road distance, and road duration between customer location and approximate warehouse location.

$$\begin{aligned} \mathrm{J}_i(w_i, \alpha_i, \beta_i, \gamma_i) &= \tfrac{1}{M_i}\textstyle\sum_{j=1}^{M_i}\left(\alpha_i d_h(w_i, x_j) + \beta_i d_r(w_i, x_j) + \gamma_i t_r(w_i, x_j)\right) \\ &+ \lambda(\alpha_i^2 + \beta_i^2 + \gamma_i^2) \end{aligned} \tag{5}$$

- J_i represents the cost function of cluster i
- M_i represents the total number of customers in cluster i
- w_i is the location of warehouse in cluster i
- x_j is the location of j^{th} customer in cluster i
- d_h is the Haversine distance between warehouse and customer
- α_i is the weight parameter of Haversine distance in cluster i
- d_r is the road distance between the warehouse and the customer calculated from Google Maps API
- β_i is the weight parameter of road distance in cluster i
- t_r is the road duration between the warehouse and the customer calculated from Google Maps API
- γ_i is the weight parameter of road duration in cluster i
- λ is the learning parameter for regularization

The cost function J_i (shown in equation (5)) is essentially a composite measure of the weighted sum of the haversine distance d_h, the road distance d_r, and the road duration t_r, normalized by the number of customers M_i in the cluster. The weights $\alpha_i, \beta_i, \gamma_i$ correspond to the significance attributed to each of these measures and are subject to optimization. The inclusion of a regularization term with a parameter λ addresses the potential issue of over-reliance on any single distance metric. It ensures a balanced contribution from each feature, enhancing the generalizability of the warehouse location decision. The regularization term adds a penalty for large weight magnitudes, thereby preventing overfitting and promoting a more nuanced decision surface. To maintain the integrity of the weights and avoid the trivial solution where one or more weights could be zero, thus nullifying their respective distance measures, a logarithmic penalty is incorporated.

$$\mathbf{w} = [\alpha_i, \beta_i, \gamma_i]^T \tag{6}$$

$$l_i = -\sum \log(\mathbf{w} + \epsilon) \tag{7}$$

In the equation (7), l_i represents the log penalty for cluster i, and ϵ is a small constant added to prevent the logarithm from diverging as the weight parameters approach zero. This ensures that all distance measures exert a non-negligible influence on the cost function.

This form of the cost function, which incorporates both the individual cluster costs and the logarithmic penalties for the weight parameters, is a comprehensive expression that holistically captures the optimization objective. This aggregate cost function is designed to minimize not only the distance-related costs associated with warehouse placement but also to maintain a balanced influence of each distance measure through regularization. The formulation in equation (8) represents the final cost function used for the optimization process.

$$total\_cost = J_i + l_i \tag{8}$$

The final cost function inherently possesses a non-convex nature. This characteristic stems from the inclusion of the logarithmic penalty term and the quadratic regularization component. Non-convex functions are known for their complex landscapes, featuring multiple local minima and possibly a global minimum, which are not easily navigable using traditional gradient-based optimization methods. In non-convex optimization, the primary challenge is to avoid getting trapped in local minima that do not represent the optimal solution. Gradient-based methods are particularly susceptible to this issue in non-convex settings and tend to converge to the nearest local minimum, which might be significantly sub-optimal compared to the global minimum. To circumvent these challenges, various randomized optimization methods such as the Nelder-Mead simplex algorithm, Particle Swarm Optimization (PSO), and Simulated Annealing (SA) are employed. These methods do not exclusively rely on gradient information and are designed to explore the solution space more broadly. This broader exploration increases the likelihood of finding the global minimum, or at least a more favorable local minimum, in the complex landscape of a non-convex function.

1) Nelder-Mead Simplex: The Nelder-Mead Simplex Algorithm [7], described in Algorithm 3, is a heuristic optimization technique created for non-linear and multi-dimensional optimization. It starts by constructing an initial simplex, a geometrical figure consisting of N+1 points in an N-dimensional space, which represents different potential solutions. In this case the N-dimensions are warehouse location (latitude and longitude pair) and the weights, α, β and γ The function values at these vertices are calculated to evaluate their effectiveness. The algorithm, through a series of strategic operations, dynamically adjusts the simplex to navigate the function's landscape towards an optimal or near-optimal solution.

The operations within the algorithm, as outlined in equations (9), are pivotal for its iterative process. Each operation - Reflection, Expansion, Contraction, and Shrinkage - modifies the simplex in a specific manner:

Algorithm 3 Nelder-Mead Algorithm

Require: Initial set of points forming a simplex, function to minimize
Ensure: Optimized parameters
1: Initialize simplex with $N + 1$ points in N-dimensional space
2: Evaluate function at each point of the simplex
3: **while** termination criteria not met **do**
4: Order simplex points by function values
5: Calculate centroid of all points except the worst
6: Reflect the worst point across the centroid
7: **if** reflected point has better function value **then**
8: Expand the simplex in this direction
9: **else**
10: Contract the simplex around the best point
11: **end if**
12: **if** contraction does not yield improvement **then**
13: Shrink the simplex towards the best point
14: **end if**
15: **end while**
16: **return** Best point from simplex =0

- Reflection moves the worst point across the centroid, potentially finding a better position.
- Expansion further probes the direction of improvement if reflection yields a promising result.
- Contraction is applied when neither reflection nor expansion improves the result, pulling the simplex towards the best point.
- Shrinkage is the last resort, reducing the simplex size towards the best point when other operations fail to find improvement.

In the series of equations (9), x_w, x_c, and x_b represent the worst point, centroid point, and best point of the simplex, respectively. The constants η_1, η_2, η_3, and η_4 are parameters that govern the transformations of the simplex during the optimization process. These parameters are inherent to the algorithm and are typically fixed at standard values (reflected $= 1$, expanded $= 2$, contracted $= 0.5$, and shrinkage $= 0.5$). This approach ensures that the algorithm not only searches broadly but also fine-tunes its exploration, honing in on the most promising regions of the solution space.

$$\begin{aligned} &\text{Reflection:} \quad x_r = x_c + \eta_1(x_c - x_w) \\ &\text{Expansion:} \quad x_e = x_c + \eta_2(x_r - x_c) \\ &\text{Contraction:} \quad x_{co} = x_c + \eta_3(x_w - x_c) \\ &\text{Shrinkage:} \quad x_s = x_b + \eta_4(x_i - x_b), \forall i \end{aligned} \tag{9}$$

In the application of Nelder-Mead Simplex algorithm, the first step is to initialize the simplex with the approximated warehouse locations from clustering algorithms and randomized α, β and γ values. The algorithm's stopping criteria is set to 6 iterations due to computational complexity and resource

constraints introduced due to usage of external Google Maps API.

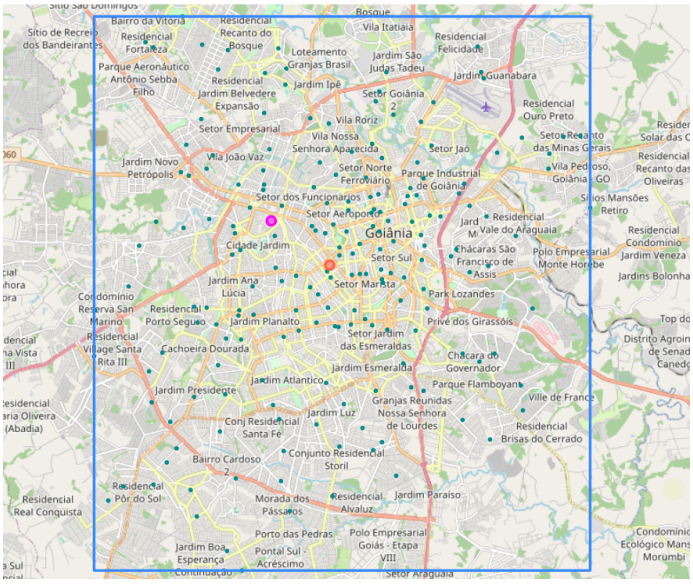

Fig. 3. Warehouse location optimization using Nelder-Mead algorithm

The results of the Nelder-Mead optimization method is depicted by the shift from the initial cluster centroid (red dot) to the optimized warehouse location (magenta dot), as shown in figure 3. The process resulted in a substantial reduction in cost, from an initial figure of 109.85 to a final cost of 21.25, as shown by the relocation from the red dot to the magenta dot. This demonstrates the algorithm's effectiveness in minimizing the objective function, which likely includes logistical factors such as transportation distances and costs. The value of alpha at 2.12 suggests a strong emphasis on the haversine distance in the cost function, beta at 0.89 assigns lighter importance to the road distance, and the relatively high value of gamma at 6.17 indicates that the road duration. This configuration of weights led to a location that minimized the composite cost effectively, as evidenced by the substantial reduction in cost from the initial to the final value.

2) Particle-Swarm Optimization: Particle Swarm Optimization [8] is a meta-heuristic algorithm that simulates the social behavior of swarms. It is designed to solve continuous and discrete optimization problems by iteratively improving a candidate solution with regard to a given measure of quality. PSO optimizes an objective function $\mathbf{f} : R^n \rightarrow R$ through the collective and intelligent behavior of particles within the search space. Each particle adjusts its trajectory based on its own historical best position (p_best) and the global best (g_best) found by any particle in the swarm. PSO is initialized with a group of random particles (solutions). These particles are then iteratively updated. This is shown in the algorithm 4.

In the above algorithm 4, the velocity ($v_i^{(t)}$) and position ($x_i^{(t)}$) of each particle represent its movement and placement in the solution space, respectively. The personal best ($p_{best,i}$) and global best (g_{best}) positions are the best solutions found by

Algorithm 4 Particle Swarm Optimization Algorithm

Require: Objective function $f : \mathbb{R}^n \rightarrow \mathbb{R}$, Number of particles P, Max iterations T
Ensure: Best found position g_{best} minimizing f

for $i = 1$ to P **do**
  Initialize particle position x_i randomly
  Initialize particle velocity v_i randomly
  Set personal best position $p_{\text{best},i} = x_i$
end for
Set global best position g_{best} to the best of $p_{\text{best},i}$ among all particles
for $t = 1$ to T **do**
  for $i = 1$ to P **do**
    Update velocity: $v_i = w \cdot v_i + c_1 \cdot \text{rand}() \cdot (p_{\text{best},i} - x_i) + c_2 \cdot \text{rand}() \cdot (g_{\text{best}} - x_i)$
    Update position: $x_i = x_i + v_i$
    if $f(x_i) < f(p_{\text{best},i})$ **then**
      Update personal best: $p_{\text{best},i} = x_i$
    end if
    if $f(p_{\text{best},i}) < f(g_{\text{best}})$ **then**
      Update global best: $g_{\text{best}} = p_{\text{best},i}$
    end if
  end for
end for
return g_{best} =0

the individual particle and the entire swarm. The inertia weight (w), cognitive coefficient (c_1), and social coefficient (c_2) regulate the particle's momentum and its attraction towards personal and global bests. Random factors r_1 and r_2 ensure diversity in the search process. These elements collectively guide the swarm towards optimal solutions, balancing exploration and exploitation.

The PSO was initialized with 10 particles, a number chosen to balance the exploration of the solution space with computational feasibility. Each particle represented a potential warehouse location and was characterized by randomized values of α, β, and γ parameters. The initial geographical positions of these particles were determined within a bounding box, formulated as a rectangle encompassing all customer locations in the cluster. This bounding box was defined by the extreme latitudinal and longitudinal points among the customer locations, ensuring the practicability of potential warehouse sites. Considering the computational complexity, particularly the overhead of querying the Google Maps API for calculating distances and travel times for each particle in every iteration, the algorithm's stopping criterion was set to 5 iterations. This decision was a pragmatic approach to balance the depth of search within the solution space against the computational and API usage constraints.

The results depicted in the figure 4, with the red dot representing the initial approximate warehouse location derived from clustering techniques and the magenta dot indicating the optimized location determined by Particle Swarm Optimization (PSO), reflect a significant optimization according to the

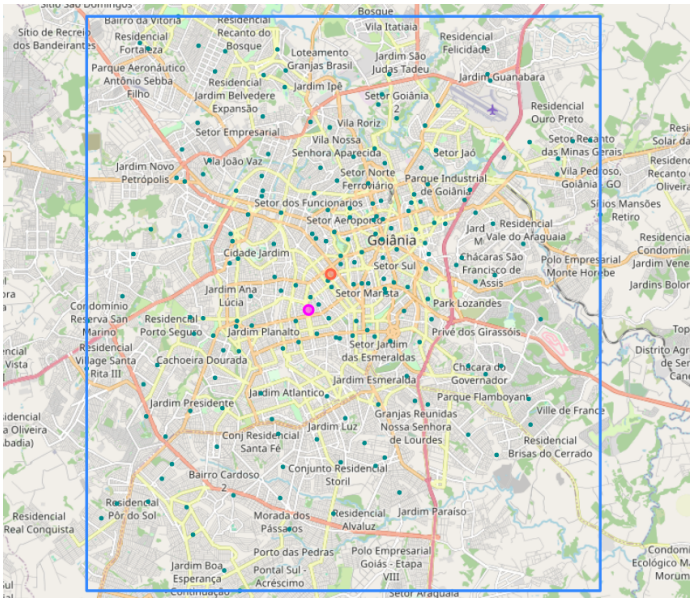

Fig. 4. Warehouse location optimization using PSO algorithm

Algorithm 5 Simulated Annealing Algorithm

Require: Objective function $f : \mathbb{R}^n \to \mathbb{R}$, initial temperature T_0, cooling rate α, minimum temperature T_{min}
Ensure: Optimized solution s_{best}

```
Initialize solution s randomly
Set s_best = s
Set temperature T = T_0
while T > T_min do
    Generate a new solution s' in the neighborhood of s
    Calculate ΔE = f(s') − f(s)
    if ΔE < 0 or exp(−ΔE/T) > random(0, 1) then
        Set s = s'
        if f(s') < f(s_best) then
            Set s_best = s'
        end if
    end if
    Update temperature: T = η · T
end while
return s_best =0
```

defined cost function. The initial cost, assessed at 94.12, has been substantially reduced to 15.63, which signifies a marked improvement in the objective measure of quality for the warehouse location. The values of alpha (7.52), beta (8), and gamma (2.67) represent the weights of the various components within the cost function. The high values of alpha and beta indicate that the proximity and road network efficiency have been heavily prioritized in the cost function, which suggests that the optimized location is significantly more favorable in terms of these two parameters compared to the initial estimate. The relatively lower value of gamma, while still important, suggests that the road duration factor, although optimized, was not as heavily weighted in the cost function.

3) Simulated Annealing: Simulated Annealing [9] is a probabilistic optimization technique that draws inspiration from the metallurgical process of annealing. In the domain of optimization algorithms, SA stands out with its unique approach of refining a single solution through a process of iterative improvement and probabilistic transitions. It is particularly useful for optimization problems where the search space is large and the potential for getting trapped in local optima is significant. The algorithmic foundation of SA is the concept of temperature-dependent transition probability, which allows for occasional uphill moves, thereby granting the algorithm the capability to escape local optima. This probability decreases as the ”temperature” parameter is lowered, ensuring that the exploration of the search space is vigorous in the early stages and progressively becomes more focused on exploitation as the temperature decreases. This process continues until the system has cooled to a predefined final temperature or other stopping criteria are met. The algorithm 5 explains the application of SA for optimization.

The Simulated Annealing (SA) algorithm begins with an initial solution s and an initial temperature T_0. The best solution s_{best} is initially set to this starting point. The algorithm iteratively explores the solution space by generating new solutions $s^{'}$ near the current solution s. The change in the objective function's value, E, guides the acceptance of new solutions; a new solution is accepted if it improves the objective function, or with a probability determined by the Metropolis criterion, which allows accepting worse solutions to escape local minima. This criterion depends on the temperature T and the difference in the objective function values E. The temperature is progressively reduced according to a cooling rate η until it reaches a minimum threshold T_{min}. The algorithm's ability to probabilistically accept worse solutions decreases as the temperature lowers, thus reducing its exploration range and focusing more on the exploitation of the best-found solutions. The output of the algorithm is the best solution s_{best} found during the search.

In the implementation of SA, the initial temperature parameter, T_0, was set to 250. This relatively high starting temperature allowed for a broad initial search and the potential acceptance of sub-optimal solutions. The cooling rate, η, was set to 0.99 (1% reduction in temperature), representing a gradual decrease in temperature to ensure a thorough exploration of the solution space as the algorithm progressed. The number of iterations was deliberately limited to 10, a decision dictated by the computational cost of the experiment.

The results of SA, as depicted in figure 5, showcases an optimization process where the initial warehouse location—indicated by the red dot—has been reassessed to find a more cost-effective position, represented by the magenta dot. The initial cost associated with the warehouse location was significantly reduced from 162.54 to 88.04 through the application of SA. The optimized values for alpha (0.03), beta (0.17), and gamma (0.46) suggest the relative importance of each factor within the cost function. The low value of alpha indicates that the haversine distance had a minimal impact on

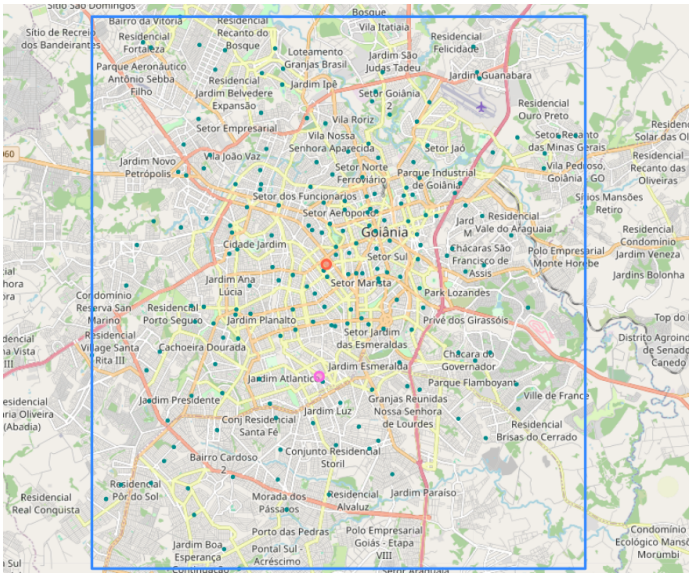

Fig. 5. Warehouse location optimization using SA algorithm

the cost, while the moderate value of beta points to a greater but still limited influence of the road distance. Conversely, the high value of gamma underscores the road duration as the most influential factor in the optimization process.

IV. Results & Discussions

The results section of this study, meticulously analyzes and interprets the outcomes derived from the application of various clustering and optimization techniques. To validate and ascertain the reliability of the optimization outcomes, the study incorporates the statistical technique of bootstrapping. This resampling method is employed to estimate the distribution of the optimized parameters and assess the stability and consistency of the solutions provided by the heuristic optimization algorithms. Through bootstrapping, the study generates a robust distribution of optimal solutions from multiple data samples. This approach ensures that the conclusions drawn are not only statistically significant but also resilient to the variability intrinsic to heuristic-based optimization.

A. Performance comparison of clustering techniques

Based on the results presented in Table 1, the analysis of DBSCAN and HDBSCAN through key performance metrics—Silhouette Score, Davies-Bouldin Index, and Calinski-Harabasz Index—reveals distinctive outcomes for each algorithm. The Silhouette Score, which quantifies the clustering clarity, suggests that DBSCAN achieves a marginally higher level of cluster definition and separation than HDBSCAN. In terms of cluster compactness and separation, as measured by the Davies-Bouldin Index, DBSCAN again demonstrates superior performance with a lower score compared to HDBSCAN. Furthermore, the Calinski-Harabasz Index indicates a more substantial disparity between the algorithms, with DBSCAN recording a value that markedly exceeds that of HDBSCAN, pointing to a significantly better ratio of between-cluster to within-cluster variance. This comprehensive assessment, substantiates the selection of DBSCAN as the more apt clustering technique for this specific dataset, particularly when the objective is to delineate well-separated and tightly clustered groups.

TABLE I
Performance Metrics for DBSCAN and HDBSCAN

Algorithm	Silhouette Score	Davies-Bouldin Index	Calinski-Harabasz Index
DBSCAN	0.618	0.173	154,919.69
HDBSCAN	0.597	0.431	72,777.34

The map-based visualization of customer data clustering using DBSCAN and HDBSCAN as shown in Figure 6 reveals significant differences in how each algorithm handles spatial data, especially in the context of an uncertain dataset. DBSCAN shows a tendency to create a notable number of outliers, indicating possible challenges in clustering less dense or ambiguous areas. This is particularly visible in areas with sparse customer distribution, where DBSCAN fails to form distinct clusters. On the other hand, HDBSCAN demonstrates a more inclusive approach to clustering. It successfully incorporates a broader range of data densities, from sparse to dense regions, as evidenced by the reduced number of outliers and the formation of more cohesive clusters in the visualization. This qualitative distinction in the spatial representation of the clusters provides an essential perspective on the clustering behavior of both algorithms.

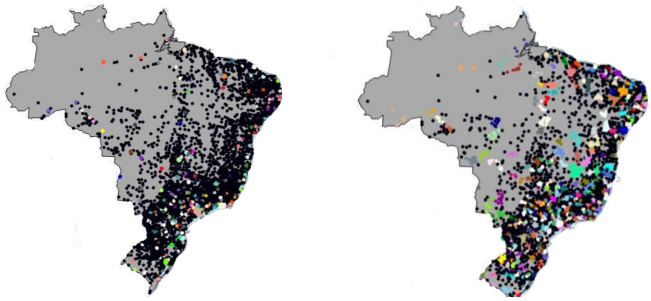

Fig. 6. Visual Comparison of DBSCAN and HDBSCAN

Choosing between DBSCAN and HDBSCAN, therefore, hinges on the specific requirements of the analysis and the nature of the dataset. If the primary goal is to focus on densely populated customer regions with clear delineations, DBSCAN's approach might be more suitable. However, for a dataset characterized by variability and uncertainty, where capturing a wide spectrum of customer densities is crucial, HDBSCAN emerges as the preferable option. Its ability to adapt to different data densities and minimize outliers makes it a more robust and versatile tool for warehouse location estimation, particularly in scenarios where customer distribution patterns are diverse and unpredictable. Thus, HDBSCAN stands out as the more appropriate choice, despite HDBSCAN

exhibiting marginally lower score metrics as evidenced by the table, the discrepancy is not substantial enough to deem it an inferior solution.

B. Evaluation of optimization techniques

Each algorithm was subjected to a series of iterations to assess its performance dynamics. The algorithms were tested across a spectrum of iteration counts, allowing for granular analysis of both the computational time and the associated cost at each step. This graph served as a basis for comparative analysis, offering a visual and quantitative foundation to evaluate the efficiency and the quality of solutions of the optimization functions.

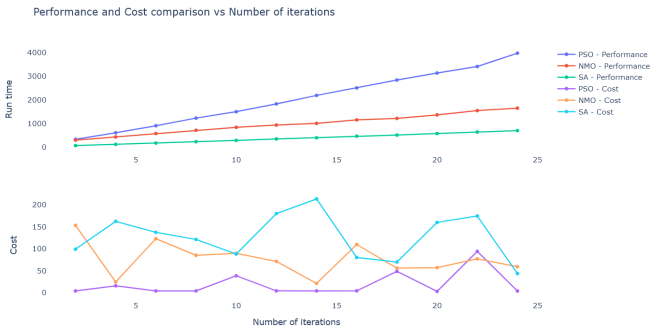

Fig. 7. Visual Comparison of Optimization Techniques

Analyzing the run-time graph, as shown in figure 7, we see distinct behaviors across the optimization algorithms. PSO demonstrates a linear increase in run-time, suggesting greater computational expense with additional iterations, which could impact its applicability in scenarios with limited computational resources. NMO presents a more consistent and moderate rise in runtime, indicating a relative balance between computational time and iterative increase. The performance time graph for SA demonstrates consistently low run times with only a minimal escalation observed as the number of iterations increases.

Turning to the cost graph, PSO indicates an initial optimization benefit that stabilizes, suggesting it could be a suitable option for achieving a quick, satisfactory solution when computational time is of the essence. NMO displays fluctuations in cost, yet it maintains a generally stable range, which might be advantageous for applications requiring consistent performance over a predictable timeframe. On the other hand, SA, despite its potential for lower costs at higher iterations, exhibits significant fluctuations. This volatility in cost performance with SA may raise concerns about its dependability for achieving consistently low-cost solutions across successive iterations.

Considering these observations, while SA has the potential to achieve lower costs, the apparent instability suggests that NMO might be a more reliable choice when consistency and stability of cost are critical. Consequently, NMO could be recommended for situations where the balance between solution quality, cost stability, and computational time is key. If minimizing cost is the main goal and computational time is not a limiting factor, then employing SA with a well-adjusted cooling schedule can be a viable option, particularly in scenarios where intermittent increases in cost are acceptable.

C. Bootstrapping to finalize the final system

Bootstrapping [10] is a powerful statistical resampling technique used to estimate the distribution of a statistic by repeatedly sampling with replacement from the original data. Essentially, it enables the assessment of the reliability and variance of a predictive model by simulating the process of sampling from a population when the actual population parameters are unknown. By generating numerous pseudo-samples, bootstrapping can provide insights into the stability and accuracy of the model's estimates, such as the mean, variance, or confidence intervals. This technique is particularly useful when the dataset is limited in size or when assumptions of traditional parametric methods are not met, offering a robust alternative for inferential analysis. In the context of optimization and clustering, bootstrapping can be employed to validate the consistency of the results and to gauge the model's performance, particularly its robustness to changes in data, such as those that may occur due to an increase in the size or diversity of the dataset over time.

The bootstrapping results, visualized at a 95% confidence interval, provide a quantified measure of the robustness of the optimization functions—Nelder-Mead and Simulated Annealing—utilized in the study. The histograms, derived from 100 bootstrap samples, depict the distribution of the cost associated with the optimization outcomes.

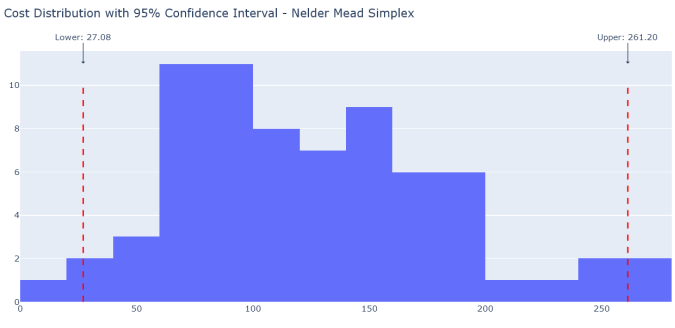

Fig. 8. Simulated annealing confidence interval at 95% on cost distribution

For the NMO, as shown in figure 8, the confidence interval is relatively wide, stretching from approximately 27.08 to 261.20. This indicates a significant variability in the cost outcomes, suggesting that the Nelder-Mead method's results can vary widely across different samples.

In contrast, the SA optimization, as depicted in figure 9, exhibits a tighter confidence interval ranging from approximately 36.63 to 232.07. This narrower interval indicates a more consistent performance of the SA method under the bootstrapped samples, implying potentially higher reliability in the cost outcomes it produces.

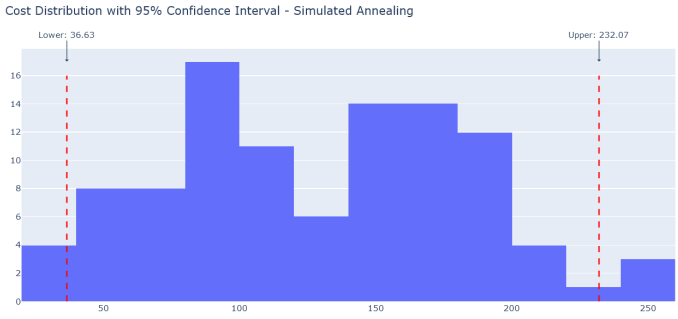

Fig. 9. Simulated annealing confidence interval at 95% on cost distribution

Based on the bootstrapping results and the corresponding 95% confidence intervals, the SA technique demonstrates a more consistent optimization performance with a narrower confidence interval. The tighter range indicates less variability in the cost outcomes across the bootstrap samples, suggesting that SA is likely to produce more reliable and predictable results when applied to different samples or under varying conditions.

V. Limitations & Future works

Understanding the limitations of the study, it is critical to acknowledge that the dataset's concentration on Brazil limits the broader relevance of the results to a specific geographic context. Because the study primarily deals in customer locations, the absence of seller locations is both a constraint and an opportunity for future research. Subsequent studies might look at integrating seller location data to offer a more complete view of e-commerce supply chain dynamics. Given the static nature of the dataset and the constraints on dynamic warehousing feasibility, the findings are inherently confined to a snapshot in time. Future studies might look at including temporal aspects, which would allow for a more dynamic understanding of spatial patterns and logistical optimization over time. Despite these limitations, this research provides an important step toward understanding the complexities of e-commerce logistics.

Looking ahead, the study beckons toward promising avenues for future research. Given the existing dataset's constraint of about 14,000 unique customer locations and its exclusive focus on Brazil, increasing the dataset and its geographical coverage might greatly improve the generalizability of findings. Exploring datasets that span numerous nations may reveal varied geographical patterns, providing a more thorough knowledge of global e-commerce logistics.

While dynamic warehousing may be challenging in the current context, there lies an opportunity to delve into further enhancing local warehousing strategies. Future studies might look into how local warehouses, using the existing dataset, could exploit consumer order patterns to improve inventory management. Using predictive analytics to predict localized demand changes might result in more planned and responsive inventory operations.

VI. Conclusion

The overarching objective of this study was to ascertain the most effective methodology for optimizing warehouse locations to minimize logistical costs, characterized by factors such as Haversine distance, road distance, and road duration. This complex problem requires a strategic balance between proximity to customers and efficient use of transportation networks. To address this, the study initially employed clustering algorithms to identify preliminary warehouse locations based on customer distribution, with DBSCAN and HDBSCAN providing foundational insights into potentially dense regions of customer locations.

The subsequent application of optimization techniques, specifically PSO, NMO, and SA, refined these initial estimates. Each optimization method brought its strengths and variability to the problem, as reflected in the cost outcomes. SA, with its narrower confidence interval derived from bootstrapping, demonstrated a higher degree of consistency and predictability in its results, indicating its suitability for problems demanding robustness against variability in operational scenarios. Furthermore, the bootstrapping approach underscored the variability inherent in the optimization methods, providing a deeper understanding of each technique's reliability. While NMO showed potential in certain instances, its wider confidence interval suggests that it might be less reliable across various scenarios compared to Simulated Annealing.

In conclusion, the study's findings suggest that Simulated Annealing stands out as a particularly effective method for the optimization of warehouse locations, balancing the logistical considerations efficiently. The consistency of its performance, as highlighted by the bootstrapping analysis, aligns with the critical need for predictability in logistical planning. However, it is important to note that the choice of an optimization technique may still be context-dependent, and considerations such as computational resources, problem scale, and specific constraints should guide the final selection for real-world applications. The methodologies explored in this study contribute valuable insights into the complex decision-making processes governing logistical optimization and warehouse placement, with implications for both operational efficiency and customer service quality.

In concluding this investigation into warehouse location optimization for e-commerce logistics, it is imperative to acknowledge the robustness introduced by bootstrapping the optimization techniques applied. This step not only reinforces the validity of the current findings but also instills confidence in the scalability of the models, ensuring that they remain competent as data volumes expand. The methodologies employed, from the careful selection of clustering algorithms to the strategic application of optimization techniques, underscore a comprehensive framework that is not limited to the confines of the specific logistical issue.

The analytical rigor and adaptability of the approach we have adopted bear the potential for wide applicability across various industry challenges. The same principles and tech-

niques can be adeptly applied to other domains where clustering and optimization play a pivotal role, from supply chain management to route optimization, and beyond. This universality is particularly valuable in the fast-paced and data-rich environments that characterize modern industries. By leveraging such data-driven, scalable, and robust methods, businesses can not only enhance operational efficiency but also adapt to the evolving landscapes and the ever-increasing complexities of their respective markets. Thus, while the focus was on optimizing warehouse locations, the implications of the research extend to a myriad of optimization problems, presenting a versatile toolkit for industry-wide applications.

ACKNOWLEDGMENT

This project was undertaken as part of the Advanced AI course at Western University for the Master's in Computer Science program. We would like to express our gratitude to the faculty and instructors at Western University for their guidance and support throughout the course. Special thanks to Yalda Mohsenzadeh and Boyu Wang for their valuable insights and assistance during the project. We are also thankful for the academic resources and facilities provided by Western University that contributed to the successful completion of this research.

REFERENCES

[1] W. Zhang and S. V. Ukkusuri, "Share-a-Cab: Scalable Clustering Taxi Group Ride Stand From Huge Geolocation Data," in IEEE Access, vol. 9, pp. 9771-9776, 2021, doi: 10.1109/ACCESS.2021.3050299.

[2] F. Li, W. Shi and H. Zhang, "A Two-Phase Clustering Approach for Urban Hotspot Detection With Spatiotemporal and Network Constraints," in IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 14, pp. 3695-3705, 2021, doi: 10.1109/JSTARS.2021.3068308.

[3] Kaggle, "Brazilian E-Commerce Public Dataset by Olist," 2023. [Online]. Available: https://www.kaggle.com/datasets/olistbr/brazilian-ecommerce. [Accessed: 26-06-2020].

[4] National Geospatial-Intelligence Agency, "World Geodetic System 1984 (WGS 84)," National Geospatial-Intelligence Agency, 2023. [Online]. Available: https://earth-info.nga.mil/GandG/wgs84/. [Accessed: 19-12-2023].

[5] Ester, M., Kriegel, H.-P., Sander, J., Xu, X. (1996). A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining (KDD-96). Portland, OR: AAAI Press.

[6] Campello, R. J. G. B., Moulavi, D., Sander, J., Zimek, A. (2013). Density-Based Clustering Based on Hierarchical Density Estimates. In Proceedings of the 17th Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2013), Lecture Notes in Computer Science, Vol. 7819, pp. 160-172.

[7] J. A. Nelder and R. Mead, "A Simplex Method for Function Minimization," in The Computer Journal, vol. 7, no. 4, pp. 308–313, 1965.

[8] J. Kennedy and R. Eberhart, "Particle swarm optimization," in Proceedings of ICNN'95 - International Conference on Neural Networks, vol. 4, pp. 1942-1948, Perth, WA, Australia, 1995.

[9] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by Simulated Annealing," in Science, vol. 220, no. 4598, pp. 671-680, 1983.

[10] B. Efron, "Bootstrap Methods: Another Look at the Jackknife," in Annals of Statistics, vol. 7, no. 1, pp. 1-26, 1979.