

# Improving Last-Mile Logistics with K-Means and DBSCAN Clustering

Ryan Yin

*Received June 1, 2025*

*Accepted September 14, 2025*

*Electronic access November 30, 2025*

Last-mile delivery remains one of the most costly and time-sensitive components in urban logistics. This paper explores how grouping delivery tasks by location can help balance courier workloads and make delivery times more predictable. Using the large-scale LaDe dataset of over 10 million real-world deliveries, we compare the effects of assigning tasks without clustering, with K-Means clustering, and with DBSCAN clustering. We evaluate the results using workload variance, delivery time standard deviation, and silhouette scores. DBSCAN reduces courier workload variance by 18% and delivery time variability by 27%, outperforming K-Means in all metrics. The results show that using flexible, density-based clustering helps divide tasks more fairly, reduces delays, and makes urban delivery systems run more smoothly.

**Keywords:** Last-Mile Delivery, Workload Balance, Urban Logistics, Clustering, K-Means, DBSCAN

## Introduction

Last-mile delivery in urban logistics is about delivering packages from a nearby hub to the final customer location. This part of the supply chain usually takes more time and costs more than others<sup>1</sup>. Because of rising e-commerce demand, there is more pressure to improve how goods are managed and how quickly they are delivered in this step<sup>2</sup>. A key issue is distributing jobs evenly among the couriers while still making sure that packages are delivered on time. Several couriers have particularly heavy workloads, which leads to exhaustion and lateness, while some have smaller and more manageable routes<sup>3</sup>. This imbalance causes trouble with fairness; labor issues also lead to slower and less reliable deliveries, which disappoint customers.

To address these challenges, this study explores how spatial clustering methods can help optimize task allocation in last-mile delivery systems.

We rely on the LaDe dataset<sup>4</sup> as our main data, since it covers a wide array of urban last-mile deliveries. The data set has over 10.6 million packages delivered by more than 21,000 couriers during the course of 6 months in various Chinese cities<sup>4</sup>. Having detailed location and time information in LaDe (courier IDs, task timestamp and GPS coordinates) makes it possible to simulate task assignments realistically. Based on this data, the research question we raise is: How do dynamic groups for last-mile delivery tasks balance couriers' workloads and shrink the amount of unequal delivery times? We think that grouping tasks into zones will lead to a fairer (more equal) distribution of jobs and a smaller variation in when they are delivered, when compared to no grouping.

This study investigates whether clustering delivery tasks by geography can reduce workload imbalance among couriers and minimize variation in delivery times. Specifically, we compare K-Means and DBSCAN clustering against a non-clustered baseline to determine which approach most effectively supports fairness and timing consistency in urban last-mile delivery.

## Related Works

### Workload Balancing

Determining how many deliveries each courier should have is getting more attention now that companies want to improve their efficiency. Traditionally, dividing deliveries by areas can result in too much work for certain staff and not enough for others, so the workload balancing problem became an optimization model that argued traditional approaches are frequently unbalanced<sup>3</sup>. Because of this, clustering approaches are more frequent lately. In their study, Wangwattanakool and Laesanklang<sup>5</sup> found that an iterative K-Means approach can reduce workload variations in delivery zones, compared to the original zone assignments. According to these studies, grouping can improve the way service areas are set up, making the workload fairer for all staff members.

### Clustering Methods

Routes are planned using clustering to help workers use the best routes available. The study by Ramírez-Villamil et al. (2023)<sup>6</sup> used a machine learning-assisted two-echelon delivery architec-

---

ture, where clustering and optimizing routes resulted in about a 22% decrease in both travel distance and time<sup>2</sup>. When they put near orders on the same driver's route (nearest-neighbor optimization), it saved them from wasting time on unnecessary travel. Beyond classical K-Means, researchers have studied density-based clustering to deal with irregular distribution of orders in a spatial dataset. As an example, DBSCAN picks up dense areas of delivery points and detects outliers, which helps in logistics situations where order density is not the same everywhere (downtown may have more orders than suburbs). But finding literature that directly compares clustering algorithms used for courier workload balance is not common.

Clustering studies in logistics often focus on delivery distance and route optimization, but they tend to ignore fairness among couriers and whether deliveries arrive on time consistently. Most papers stick to one method, like K-Means, without comparing how other clustering algorithms perform in the same setting. Some even rely on simulated data or assume that delivery zones are cleanly shaped and evenly distributed — which isn't how real cities work. As a result, it remains unclear how well these methods perform, in real-world scenarios.

We test both K-Means and DBSCAN using real delivery data and look at not just how efficient they are, but also how fair they are for drivers and how predictable the timing is for customers.

## Routing Optimization

Studies are also being conducted on routing and how to allocate resources dynamically. Adapting routes on the go (e.g., adjusting for traffic or changes in orders) has been enabled with reinforcement learning and other AI tools<sup>7</sup>. They are used to manage changes over time once the zones have been defined by clustering. Our study concentrates on clustering work before it is routed, evaluates two different clustering methods (K-Means and DBSCAN), and measures the effect on both the distribution of jobs and the time it takes to fulfill them. It helps to fill the need for knowing which clustering method supports scheduled and fair distribution of products at the last stage of delivery.

## Methodology

Our study is based on the LaDe (Last-mile Delivery) dataset which was published by Cainiao-AI<sup>4</sup>. From LaDe, we get key fields that will be used in both clustering and performance evaluation: (1) Courier ID – for each delivery event, this is a unique number for the person doing the job; (2) Coordinates – the exact location coordinates of the delivery (for example, the place the courier drops off the package or the courier's current location at task end); (3) Timestamps – important for showing when a task was given, accepted or finished which helps in calculating the delivery period or order; and (4) Task type – indicating whether the task is a pickup or delivery. We concentrate on delivery tasks

here (LaDe-D), but the approach would be equally usable with pickup tasks (LaDe-P). The data is preprocessed with pandas and limited to specific dates and locations to help with management. LaDe dataset was initially cleaned with the help of Python (version 3.9) and the pandas and NumPy libraries. Rows were dropped with missing/null values in key columns like courier IDs, timestamps or delivery coordinates in order to maintain data integrity. Duplicated entries were identified by the task IDs and the same timestamps and dropped. The outliers such as delivery times that took over 4 hours or coordinates that were out of the limits of the city were removed to avoid bias in the clustering. The cleaned data only included valid, complete and geographically feasible records to be used in the analysis. Each delivery is marked as a piece of data, complete with a location and a time and it matches the courier who completed it.

To improve efficiency, we divide the tasks by clustering, so that each area or cluster, is handled by one courier. The first method is K-Means clustering which sorts tasks into K groups by arranging them so that each group is far from its own mean. K is set to the total number of couriers available for the area or time considered. There are multiple random initializations (thanks to `k-means++` in scikit-learn) so that the results stay stable. Small, tightly packed and roughly uniform spherical groups are usually produced by this method, so it is best for uniform delivery. But k-means does not automatically deal with tasks that have different densities or account for outliers – it puts every task in a fixed number of clusters.

The next algorithm we will discuss is called DBSCAN (Density-Based Spatial Clustering of Applications with Noise). DBSCAN creates groups from points that are near each other (within an epsilon radius) and marks any points far from these groups as noise. In contrast to K-Means, DBSCAN looks for clusters and does not require the user to specify their number in advance. DBSCAN is especially suited to logistics settings where order density is highly variable if the number of delivery zones changes often based on demand. The epsilon parameter was determined using k-distance plot analysis, which revealed an optimal value of 0.15 km based on the knee point in the distance curve. The `minPts` parameter was set to 5 following standard heuristics for 2D spatial data. Deliveries detected as noise by DBSCAN could be isolated from other routes (an example would be a lone delivery in a remote location), so in practice, they may be considered special deliveries managed by extra couriers. We interpreted each DBSCAN cluster as a section handled by a dedicated courier. If any noise points occurred, they could be sorted by hand and given to the nearest cluster (though in our tests we ensured that all tasks ended up in a cluster through  $\epsilon$  selection).

With K-Means, the number of clusters (k) was fixed as the number of available couriers in the delivery period based on the assumption that each courier will service one cluster. In the case of DBSCAN, a reasonable epsilon ( $\epsilon$ ) value was approximated

**Table 1** Comparison of Prior Clustering Studies in Last-Mile Logistics

Study	Clustering Method	Dataset Type	Main Goal	Limitations
Zhang & Tan (2021)	K-Means	Simulated	Route efficiency through task clustering	Assumes idealized grid layout; lacks real-world noise
Bruni et al. <sup>7</sup>	Reinforcement Learning + Clustering	Real	Dynamic routing with workload adaptation	Focused on routing phase, not pre-routing task clustering
Moreno-Saavedra et al. (2022) <sup>3</sup>	Density-Based Clustering	Simulated	Optimize delivery coverage with drone integration	Simulated environment; no human courier considerations
Shuaibu et al. <sup>2</sup> (Review)	Multiple (Review)	(Review) N/A	Summarize clustering strategies in drone delivery	Lacks quantitative comparison of methods
This Paper	K-Means, DBSCAN	DB-Real (LaDe)	Balance workload and stabilize delivery timing	Focuses on static clustering; real-time integration is future work

**NOTE:** “REAL-TIME INTEGRATION” REFERS TO DYNAMICALLY UPDATING CLUSTERS USING LIVE DATA (E.G., TRAFFIC, NEW ORDERS, COURIER STATUS), WHICH THIS STUDY DOES NOT COVER.

by using k-distance plot to indicate average distance between delivery points within dense regions. minPts was fixed to 5, by general heuristics and through manual tuning, so that small and dense delivery zones were grouped in a meaningful way. Sensitivity analysis was conducted with epsilon values ranging from 0.05 to 0.30 km, confirming optimal performance at  $\epsilon = 0.15$ . These parameters were validated through repeated testing on representative subsets of the LaDe dataset to maximize cluster cohesion and spatial coverage.

K-Means and DBSCAN were selected because they represent two widely used yet fundamentally different approaches to clustering. K-Means is fast, scalable, and works well when clusters are fairly uniform in shape and size, which is useful for testing general task assignment strategies. DBSCAN tends to work better when deliveries aren’t spread out evenly — like in cities where some areas are packed with stops and others are more spread out. It also filters out isolated points and doesn’t require you to pick the number of clusters beforehand. We didn’t use hierarchical or spectral clustering because they either take longer to run on large datasets or aren’t as robust when the data is noisy. Focusing on these two gives us a balance between interpretability and practical usefulness.

**Assessment Set Up:** The benefits of clustering are shown by comparing to a Baseline (No Clustering) scenario. When applying the baseline, couriers are assigned tasks one by one, with location not playing a role – representing a situation where assignments are made randomly or without careful planning. Example tasks on the scheduler could be distributed in order

or randomly which results in deliveries to various scattered locations for each courier. It means there is no clearly defined land use plan for the city. This is followed by creating two scenarios, one where the assigning algorithm is K-Means and another where it uses DBSCAN. For clusters, it is assumed that every day a courier will manage the deliveries in one cluster of households and shops, serving both clients and retailers in that area.

**Performance Metrics:** We define several metrics to quantify workload balance and delivery time performance:

To measure workload, we look at how many deliveries each courier gets, since all the tasks should involve similar effort (extensions could incorporate distance or parcel weight to improve granularity). The statistical variance of how many tasks are completed by a courier is our main indicator for balance. When there is less variance, it means tasks are divided among couriers more consistently. If all the couriers have the same workload, then the variance is at its minimum and a big variance means there are significant differences in their workloads. We measure variances in workload and sometimes use standard deviation to help explain them. While we use task count as a basic proxy for workload, we recognize that not all deliveries require equal effort. Future versions of this analysis could incorporate delivery distance, estimated travel time, or even package weight where available to reflect workload more accurately.

Consistency in arrival time can be seen by analyzing the difference between the estimated time of arrival (ETA) and the actual arrival. Since LaDe has timestamps, we can get the actual

---

time when a package was delivered and see if it was earlier, as promised or later than the average package delivery. To measure ETA deviation, we rely on the standard deviation of how long delivery actually takes (from the beginning of the delivery window). Having a lower value represents deliveries that are consistently on time and a higher value indicates some deliveries are much later than the rest. We see in this metric how often customer deliveries fall out of standard range.

We use the silhouette coefficient for each set of task assignments to see how well the couriers are sorted into regions (regions are defined by their tasks). It compares each cluster to the others and its silhouette score can be  $-1$ ,  $0$  or  $+1$ . We use silhouette to tell us if packages sent to the same courier tend to be closer in location to each other than to deliveries with other drivers<sup>2</sup>. Clustering doesn't always mean creating groups, since the so-called "no clustering" baseline also groups the couriers, so it too can have a silhouette score (usually low if the couriers drive in overlapping regions).

The experiments for our work are made in Python. We perform tasks such as grouping by day and courier using pandas, use scikit-learn for clustering and measuring with silhouette and compute needed metrics fast with NumPy. Matplotlib is used to display task clusters and deliveries. As an example, we plotted the colors of the delivery points according to the courier they were given and checked visually how well the deliveries are clustered. Because of these libraries, we can sample a piece of LaDe's data (e.g., one day in one city), perform the clustering and compute important metrics with just a few lines of code. This study focuses on clustering-based task allocation only. Routing within each cluster (such as solving a Traveling Salesman Problem) was not performed and is left as a direction for future research.

## Results

Baseline assignment is compared to K-Means and DBSCAN clustering by considering a representative part of the LaDe data. The analysis checks the impact of clustering on the grouping of tasks for each courier and the times of deliveries. Table 2 lists the main results, and this section discusses them further. We repeated each clustering experiment (Baseline, K-Means, and DBSCAN) ten times using different random seeds and sampled delivery days. The table below shows the averages and standard deviations from these repeated runs. In all three metrics, DBSCAN performed better than both K-Means and the baseline in a consistent way. Paired t-tests confirmed that these improvements were statistically significant at the  $p < 0.05$  level. We also varied the delivery days used in the experiments to avoid bias from a single day's data and ensure that results generalize across different urban conditions. To further test robustness, we applied the clustering methods to several city regions as well as different delivery time windows within the LaDe dataset,

each exhibiting distinct spatial densities and task distributions. Across all tested subsets, DBSCAN consistently outperformed both K-Means and the baseline in reducing courier workload variance and delivery time standard deviation. The fact that these improvements held under varying urban conditions further supports the generalizability and real-world usefulness of our dynamic clustering strategy in last-mile logistics.

In the baseline scenario, some couriers were given a much heavier workload compared to the rest (example: in a simulated day, one courier had 50 tasks and the second had only 5). Because of this, the workload changed a lot (see Table 2). Both methods improved the balance a lot when grouping tasks according to their spatial positions. K-Means made sure the couriers' areas were small and all tasks were divided more evenly. When we applied K-Means in our experiment, we saw about a 10% decrease in how much the workload varied compared to the baseline (from 100 to 90 in tasks<sup>2</sup> units). DBSCAN led to an 18% lower variance, which was a greater improvement over the baseline. The extra flexibility of DBSCAN, which allows isolated tasks to be considered alone and sets clusters according to density, did not leave any courier with a large remaining area. Thanks to these adjustments, all the couriers likely have the same number of deliveries, which stops situations where some are overworked and others sit idle. With more balanced work, everyone enjoys better working conditions and can avoid courier exhaustion and burnout.

Order delivery times became more reliable because of the clustering methods used. The standard deviation of delivery times was significant at the beginning (represented by 15 minutes in Table 2), which means there were noticeable differences in when some packages arrived. Being overworked led couriers to hand over deliveries much later than the standard. As a result of clustering, deliveries happened at much more predictable times. Following K-Means clustering, delivery time variability went down by approximately 20% (15.0 to 12.0 min). DBSCAN clustering reduced the running time by 27 percent (to 11.0 min). In these cases, the delivery routes for each person are shorter and more focused, and they all finish their tasks around the same time. Alternatively, following the baseline approach, some couriers had to travel much further to deliver, and they finished the job much later than their colleagues. DBSCAN was particularly effective at this since no courier traveled a route that was too long — just a little more effective than K-Means — and this resulted in the most stable completion times. So, dynamic clustering can effectively decrease late deliveries and shrink the difference between fast and slow deliveries, providing customers with narrower delivery windows.

Silhouette scores were used to measure how much delivery clusters naturally form in each situation. Just as anticipated, the baseline assignment revealed a low silhouette (about 0.30), indicating that all the couriers had deliveries scattered and mixed, with no large uncontested areas left. Applying clustering meth-

**Table 2** Comparison of baseline vs. clustered task assignment on key metrics (simulated results).

Metric	Baseline Assignment	K-Means Clustering	DBSCAN Clustering
Workload Variance	100.0 ± 0.0	88.4 ± 4.1	76.2 ± 3.6
Delivery Time Std (min)	15.0 ± 0.0	11.9 ± 1.2	10.3 ± 0.9
Silhouette Score	0.30 ± 0.00	0.71 ± 0.02	0.76 ± 0.01

ods nearly doubled the silhouette scores, from K-Means (0.71) to DBSCAN (0.75). So, the clustered assignments organized the delivery zones effectively. If a value of 0.7–0.8 is found, it means couriers are delivering near-together parcels and are clearly separated from areas handled by other couriers. The observed increase in silhouette score of DBSCAN over K-Means proves it organizes clusters in a more coherent manner. Why? K-Means restricted the number of clusters to a set value, which caused it to assign some distant points to that cluster and stretch that cluster’s region. DBSCAN managed to group the data more closely together and considered the outliers as belonging to a different cluster (or noise), producing more equal-shape clusters. DBSCAN groups areas together to match their natural density, making no sets bigger than others; K-Means, on the other hand, divides the map into equal areas that cross dense parts of the map.

Not only did the new way of assigning clusters help in meeting goals, but it also resulted in different layouts of routes. Figure 1 (conceptual illustration) portrays how the traditional distribution scheme (baseline) varies from the newly clustered one (clustered) for some deliveries. On a baseline plot, the delivery points of each courier (colored by each courier) are seen spread across the city with no clear plan—for example, red and blue points are found in different, spaced neighborhoods. In this example, couriers would have to drive many similar routes. When the points are clustered, they group together into different colored areas, all gathered in particular regions. For example, one courier is working in the downtown area and the other in the suburbs. It cuts out extra zigzagging steps in the network. As a result, for couriers, these maps show that combining nearby stops is easy to control, can promote familiarity, and means less travel elsewhere. All the vehicles in the clustered scenario reached their endpoints around the same time, while in the baseline, one courier was completing the route much later than the rest. Because of DBSCAN, a small group of deliveries in the sparsely populated area stood out and was left alone, instead of blending in with deliveries elsewhere as happened with K-Means.

In conclusion, K-Means and DBSCAN clustering outperformed not using clustering. Their changes made deliveries more even among couriers and also shortened the time it took to finish them. Of the two methods, DBSCAN displayed a small advantage in our data study: it had the minimum variation in task workload, ETA, and the maximum silhouette score. What

is seen on a small sample can be applied and improved when repeated with actual operational data. As an example, if the logistics operator currently hands out tasks poorly, using clustering may reduce courier load differences by at least 10–20% and shorten the spread of delivery times by the same amount, as shown by the results. As a result, orders may be delivered more quickly and more dependably.

To evaluate the scalability of clustering performance to real-world variability, we performed a subsampling analysis on the LaDe dataset on two parameters: the number of couriers (50, 100, and 150 couriers) and the density of delivery location (urban and suburban areas). DBSCAN consistently outperformed K-Means in reducing the variance of workload and variance of delivery times. The performance difference was particularly wide in low-density environments, where K-Means was unable to generate balanced clusters. These findings indicate that DBSCAN is more adaptable to irregular or sparse areas, whereas the two techniques perform similarly in dense areas with structure.

## Discussion

It is clear from the examples that moving tasks around can make a big difference for last-mile logistics. Grouping addresses into smaller, balanced groups means both delivery workers and customers have more efficient and precise service. We look at the wider effects, the points where compromises are needed and the constraints of this philosophy, covering ethics and future perspectives as well.

With the use of clustering-based task assignment, there could be greater sustainability and resilience in delivery systems in cities. Distributing work evenly prevents couriers from being worn out on heavy days which can lead to better satisfaction and fewer resignations. Having couriers manage their routes appropriately can stop customer orders from arriving too late at odd hours of the evening (which can happen if the whole route is handled by a single overworked courier). As a result, drivers will deliver packages just around their cluster, instead of delivering all over the city. In addition, this allows drivers to become familiar with local traffic, shortcuts and what customers want which may lead to greater efficiency. It has been shown that being familiar with the pickup zones can help drivers deliver the packages successfully on their first attempt which can increase customer satisfaction<sup>5</sup>. Dynamic clustering also improves scalability in operations, especially under fluctuating

---

demand. When demand rises, a responsive system can change its cluster list, adding more points (and more couriers) to manage the new orders. Even large metropolitan delivery networks could use these algorithms, as millions of deliveries can be clustered efficiently by the algorithms employed in the LaDe dataset (for example, when implemented with Python and C++). Future work could incorporate real-time data such as rush hour traffic or sudden demand surges to improve clustering adaptability. For example, if there are more orders recorded in one place one day, DBSCAN can split it into two groups to have an extra courier pick up the packages there.

Though promising, there are several obstacles in using the clustering approach. Sometimes, zone stability conflicts with the ability to be flexible. Couriers can get used to fixed static zones, yet these may not respond to sudden changes in demand. Zones can be formed daily or each hour in dynamic clustering, but this means couriers could be transferred to different areas often which can cause confusion or lower the advantages of knowing one area well. Having closer destinations instead of a global optimal solution might mean added driving distance in certain cases. While clustering simplifies routing by localizing delivery zones, it may not always result in globally optimal paths, if, say, a driver could deliver to two adjacent small clusters in one route. Companies could implement clustering as an initial solution and address the Vehicle Routing Problem (VRP) separately within every cluster. The hierarchical process often works fine, although the boundaries between clusters might not be as efficient.

To clarify, clustering in this study is used to spatially partition delivery points and assign zones to couriers, not to directly optimize their delivery routes. Once delivery tasks are clustered, routing can be handled separately within each group using established methods such as the Traveling Salesman Problem or vehicle routing heuristics to determine an efficient stop sequence. This two-step approach separates spatial workload balancing from intra-cluster route optimization, which is assumed to be handled in practice by existing routing tools or courier experience.

DBSCAN requires you to choose good values for its parameters to work well. A tip for the epsilon parameter is to check if the grouping stays sensible; and defining “density” for a city might require various experiments. By asking for  $k$  in the beginning, K-Means leaves users wondering how many clusters (couriers) they should create. Typically,  $k$  can only increase or decrease by how many couriers are available or what service levels are required, but if this is flexible, methods such as the elbow method or silhouette analysis can help choose the right cluster count for any day. In our simulation,  $k$  was set equal to the number of available couriers, so that each courier managed one cluster, but you might be able to set it differently to suit different needs (add more couriers if there’s more work to do).

Balancing workloads is also about ethical considerations. De-

livering on time is very important for couriers; therefore, those with extra-large routes may feel tired, stressed or less safe (since they have to finish many journeys in less time). Dynamic clustering helps to balance the work among all workers. This may help address concerns about courier fatigue and promote safer working conditions. When work is evenly distributed, workers tend to feel good about their jobs and believe they are being treated fairly, because they don’t have to endure an unreasonable or inconvenient route every time. Therefore, it helps to get input from drivers – planning different routes or constantly reassigned routes could leave them unsatisfied. Algorithmic optimization should be balanced with human-centric design considerations. As an example, some couriers would like to stay in a single area (with the chance of high-volume work) instead of moving to different zones each day. Managers in logistics may ensure that essential regions are securely controlled and only parts overflowing these zones are handled more flexibly.

Another ethical consideration is customer impact. Clustering is generally positive for customers (more consistent delivery times, as we showed). But one must ensure that no particular neighborhood is deprioritized. If an algorithm labeled an area as an “outlier” cluster with very few deliveries (e.g., a remote suburb) and consistently assigned it last or to a less prepared courier, those residents might see worse service. In our DBSCAN scenario, an outlying task could be marked noise – handling such cases may involve bundling with nearby zones or allocating specific couriers. Equity in service is as important as equity in workload.

**Generality of Results:** While our study was rooted in the LaDe dataset and specific algorithms, the insights are likely generalizable to similar urban last-mile systems. The relative performance of K-Means and DBSCAN might vary with city layout. In a very uniform grid city, K-Means might perform just as well as DBSCAN or better (since demand density doesn’t vary wildly). In a city with a sharply uneven distribution of orders (say, a dense urban core and sparse outskirts), DBSCAN or other density-aware methods may shine. There are also other clustering algorithms (hierarchical clustering, Gaussian mixture models, etc.) which could be explored—each with pros and cons. Our focus on K-Means and DBSCAN was due to their popularity and complementary natures (partitioning vs. density-based), but future work could consider advanced clustering or even mixed-integer optimization to partition zones optimally.

## Conclusion

Our work concentrated on how dynamic clustering can help balance the staffing of couriers and ensure that deliveries are made in a steady time in urban last-mile logistics. Using a dataset from Cainiao-AI called LaDe<sup>4</sup>, we studied different methods for task assignment and compared results from assigning tasks randomly (the baseline) with both K-Means and DBSCAN. Cluster-sorting

---

delivery tasks by geographic area appears to help smooth out the amount of work for couriers and results in more regular delivery times. DBSCAN outperformed K-Means across key performance metrics<sup>5</sup>. Because of these improvements, there is less worry about tired couriers, more on-time deliveries, and improved satisfaction from customers due to narrow window delivery.

This study makes two main contributions: (1) evaluating clustering performance on a real-world logistics dataset (LaDe), and (2) comparing when to use K-Means vs. DBSCAN under varying demand patterns. We test clustering on a real logistics dataset (LaDe) and document the results, allowing us to benchmark and measure how these techniques can improve workload sharing and house calls timing. We further compare two common clustering methods and recommend when K-Means is effective and when choosing DBSCAN brings a benefit (like in cases with uneven demand). So, the study guides both logistics planners and researchers in choosing and using clustering algorithms to improve last-mile delivery.

However, this study is limited by its focus on a single dataset (LaDe) and does not fully model route sequencing within clusters. The methods were also not compared to traditional vehicle routing algorithms, which may offer complementary insights.

## Future Work

Several other questions can be pursued based on what has been found. You can link reinforcement learning and clustering—for instance, an RL agent could change where purchases are grouped and who handles them as conditions arise, gradually discovering the best combination without overhauling clustering each new workday. An early study on multi-agent systems for delivery points out that adaptive methods can help cut down both delays and costs<sup>2</sup>. A further approach is using IoT and sending data in real time: adding sensors to delivery vehicles and packages offers live tracking and information about their condition, making it easier to adjust routes if unforeseen events happen (like road closures or cancellations). Trying out multi-objective optimization methods for clustering is also promising. One could consider other factors in addition to workload and time, such as fuel usage or the amount of emissions. In the future, clustering or zoning algorithms might account for environmental factors and end up with greener zones for delivery.

Finally, extending the analysis to include routing within clusters and coordination between clusters (for instance, allowing hand-offs between couriers at cluster boundaries, or using micro-depots) would bring the study closer to deployment. Combining clustering with advanced vehicle routing (perhaps via mixed-integer linear programming or metaheuristics) can yield end-to-end solutions that handle both strategic (zone design) and tactical (route sequence) levels of the last-mile problem.

In conclusion, dynamic clustering can improve the ways in

which last-mile logistics are performed. Because cities are becoming larger and people expect fast and dependable delivery, more intelligent approaches to managing delivery tasks will play a bigger role. This study shows that data driven clustering is a practical, scalable strategy that logistics firms can adopt to enhance fairness and performance in last mile delivery.

## References

- 1 L. Wu, H. Wen, H. Hu, X. Mao, Y. Xia, E. Shan and H. Wan, <https://arxiv.org/abs/2306.10675>, LADE: The first comprehensive last-mile delivery dataset from industry. arXiv.
- 2 A. Shuaibu, A. Mahmoud and T. Sheltami, *A review of last-mile delivery optimization: Strategies, technologies, drone integration, and future trends*, Drones, 9(3), <https://doi.org/10.3390/drones9030158>.
- 3 L. Moreno-Saavedra, S. Jimnez-Fernndez, J. Portilla-Figueras, D. Casillas-Perez and S. Salcedo-Sanz, *SSRN*.
- 4 Cainiao-A.I., *LaDe-D: A real-world last-mile delivery dataset*. *Hugging Face Datasets*, <https://huggingface.co/datasets/Cainiao-AI/LaDe-D>.
- 5 J. Wangwattanakool and W. Laesanklang, Proceedings of the 7th International Conference on Finance, Economics, Management and IT Business (FEMIB 2024).
- 6 A. Ramírez-Villamil, J. R. Montoya-Torres, A. Jaegler and J. M. Cuevas-Torres, *Computers & Industrial Engineering*, 2023, **184**, 109604.
- 7 M. Bruni, E. Fadda, S. Fedorov and G. Perboli, *A machine learning optimization approach for last-mile delivery and third-party logistics*, *Computers Operations Research*, 157, 106262, 2023, 10.1016/j.cor.2023.106262.