Replication and Evaluation of Clustering by Fast Search and Find of Density Peaks

Master in Artificial Intelligence — Universitat Politècnica de Catalunya
Unsupervised Learning: Clustering Assignment

Joan Caballero Castro

I. Introduction

Being able to write scientific articles clearly and in detail so that other researchers can replicate experiments, follow the same procedures, and validate the obtained results is an essential characteristic that all scientific publications should possess, irrespective of the field, whether it be physics, mathematics, computer science, or any other. In this work, we replicate and evaluate the implementation, experiments, and results of the Density Peaks Clustering (DPC) algorithm from the paper *Clustering by fast search and find of density peaks* by Alex Rodriguez and Alessandro Laio (2014) [1]. In their paper, Rodriguez and Laio proposed a novel clustering algorithm based on a straightforward yet fundamental idea: *"cluster centers are characterized by a higher density than their neighbors and by a relatively large distance from points with higher densities."* Following this principle, they developed an algorithm noted for its simplicity, speed, and effectiveness across a variety of datasets.

In this work, we faithfully replicated the experiments presented in the original paper, compared our results to theirs, and discussed their claims, reproducibility issues, and other problems we encountered during the algorithm's implementation.

The DPC algorithm has significantly impacted clustering research due to its simplicity and strong performance across diverse datasets. Since its introduction in 2014, multiple improvements and variants of the algorithm have emerged. For instance, [2] improves the original DPC by computing local density using K-Nearest Neighbors (KNN) and allocating non-center points using fuzzy-weighted KNN. The DPCV and MDDPCV algorithms proposed by [3] replace the Euclidean distance metric with variance and Manhattan distances, respectively. [4] introduced a heat diffusion method for more effectively estimating the probability distribution of datasets, improving sensitivity to hyperparameters. And several other improvements [5], [6], [7], [8].

With this report, we aim to strengthen the robustness of the conclusions drawn from the original paper, validate the claims made regarding their experimental results, and provide additional contributions to the original DPC algorithm. Specifically, we compare DPC to other clustering algorithms using new datasets and additional evaluation metrics such as the Adjusted Rand Index (ARI), Normalized Mutual Information (NMI), Silhouette Coefficient, and Davies-Bouldin Index (DBI).

II. Algorithm Implementation

The original paper provides a comprehensive description of the Density Peaks Clustering (DPC) algorithm, including mathematical definitions and implementation instructions. We developed our own DPC implementation rigorously adhering to the original methodology while introducing necessary improvements for computational efficiency. We developed the algorithm in Python using optimized functions from libraries such as `numpy`, `scipy`, and `sklearn`.

The algorithm operates on two primary values computed for each data point i: the local density ρ_i and the minimum distance δ_i to any point with a higher density. As defined in the paper, the local density ρ_i is the number of points located within a cutoff distance d_c. To efficiently calculate ρ_i, we first compute the pairwise distance matrix of all data points using the optimized function `cdist()` from the `scipy` library, achieving a time complexity of $O(n^2)$ and a space complexity of $O(n^2)$, similar to standard approaches. We then compute the local density ρ_i for each point i by summing over all points j that are within the specified cutoff distance d_c. This sum is efficiently computed using the vectorized operation `np.sum()` from `numpy` over the distance matrix, which maintains a time complexity of $O(n^2)$ but significantly improves runtime efficiency compared to explicit looping methods.

Alternatively, when employing a Gaussian density estimator, the local density is calculated as follows:

$$\rho_i = \sum_j \exp\left(-\left(\frac{d_{ij}}{d_c}\right)^2\right), \tag{1}$$

where d_{ij} denotes the distance between points i and j.

To compute δ_i, defined as the minimum distance from point i to any other point j with higher density, we follow the formulation from the original paper:

$$\delta_i = \min_{j:\rho_j>\rho_i}(d_{ij}). \tag{2}$$

We optimized this computation by first sorting the density values ρ in descending order, thereby limiting the search of higher density points for point i to only the previously evaluated points in the range $j \in [0, i)$. This reduces the complexity from $O(n^2)$ in a naive approach to $O(n \log n)$ due to sorting, significantly improving computational efficiency.

The cutoff distance d_c can be estimated according to the approach suggested in the original paper, where d_c is selected such that the average number of neighbors (points within distance d_c) for each point corresponds to a specified percentage of the total number of points in the dataset. To implement this efficiently, we extract the upper triangular part of the pairwise distance matrix and calculate the desired percentile of these distances, maintaining a complexity of $O(n^2)$, comparable to standard percentile calculations. As a rule of thumbs by the authors, this percentage is around 1 to 2%.

After computing ρ_i and δ_i for each data point, cluster centers are identified by selecting the points corresponding to the highest values of the product:

$$\gamma_i = \rho_i \cdot \delta_i. \tag{3}$$

The top $n\_cluster$ points according to these γ_i values are designated as cluster centers.

To efficiently assign the remaining points to the identified clusters, we sort the points in decreasing order of local density. Beginning with the points with highest density, each point i finds the closest point with higher density within the previously sorted range $[0, i)$ and is assigned to the same cluster, effectively propagating cluster assignments with an optimized complexity of $O(n \log n)$, improving significantly compared to a naive approach of $O(n^2)$.

Additionally, the original paper discusses a special restrictive assignment mode designed for small image datasets, which assigns points to clusters only if their distance to the nearest assigned neighbor is less than d_c. This method has also been implemented for specific experiments detailed later.

Furthermore, we introduced an additional step to identify and compute halo points, which facilitates debugging and cluster boundary analysis. Halo points are determined by initially constructing two boolean matrices indicating, respectively, points within distance d_c and points belonging to different clusters. A logical AND operation between these matrices yields candidate halo points efficiently. From these candidates, we effectively identify the point with the highest density ρ_b and designate as halo points all candidates with density values below or equal to ρ_b.

Through these implementations and optimizations, we have achieved a comprehensive and efficient implementation of the Density Peaks Clustering algorithm, rigorously aligned with the methodology and instructions described in the original paper.

All our experiments were conducted on a Windows 11 machine equipped with an AMD Ryzen 7 5800H CPU using Python 3.12.8, NumPy 1.26.4, scikit-learn 1.6.1, and SciPy 1.15.1, ensuring the reproducibility of the results.

III. Replication and Comparative Analysis

Throughout the Density Peaks Clustering (DPC) paper, the authors conducted several experiments that showcased their algorithm's performance, presenting visualizations of the resulting cluster assignments and the corresponding Decision Graphs (DGs). Some experiments used artificial data generated by the authors, whereas others used datasets from other papers, such as [9], [10], and [11]. In this section, we faithfully reproduce every experiment described in the paper and compare the original DPC results with those obtained using our own implementation. In all experiments, relevant information for reproducing the dataset and the exact algorithm configuration was not provided. Therefore, we configured our experiments to closely match the results presented in the paper, and justify our experimental choices wherever applicable.

A. Experiment 1: Synthetic Point Distributions

The paper begins testing DPC's performance using artificial data points generated from five Probability Distribution Functions (PDFs) (see Figure 1). Two samples of 1000 and 4000 points were generated according to these PDFs, and the results for cluster assignments and DGs for both samples were presented. For experimental reproduction, the authors provided a file containing 2000 points sampled from these PDFs. Given that the authors only provided one image representing the PDFs and one file containing 2000 sampled points, we decided not to

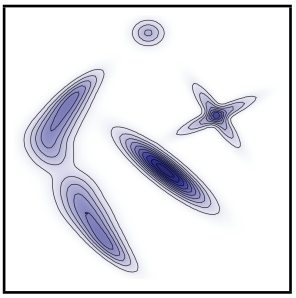
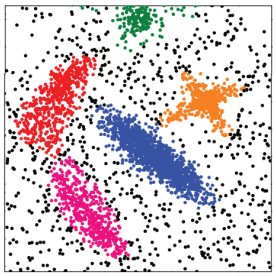


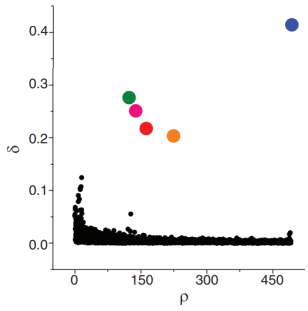
Fig. 1: The probability distribution from which point distributions were drawn. The regions with lowest intensity correspond to a background uniform probability of 20%.

create our own similar PDFs but instead reproduce their experiments using their provided sampled points. For the 4000-point scenario, we did not generate additional points; instead, we compared it with the 2000-point dataset while noting this difference in quantity for the results. For the 1000-point scenario, we randomly sampled 1000 points from the provided 2000-point dataset, acknowledging potential variations in the results due to a lower number of points and not using the exact same points as the original experiments. Additionally, the authors provided MATLAB [12] code for visualizing the cluster assignments, which was useful for us to maintain consistency in the plots. They used the Multi Dimensional Scaling (MDS) [13] algorithm for 2D visualization with the *metricstress* criterion, although they did not explicitly guarantee using the same configuration to generate their images in the paper. We also extracted their DPC hyperparameters from the MATLAB code, noting they employed a neighbor percentage of 5.6%, a cluster count of 5, and a Gaussian density estimator. However, upon testing this configuration, we found discrepancies. Specifically, the number of points assigned to clusters was greater in the original paper than in our replication. Consequently, we reduced the neighbor percentage to 2.5% to obtain results more aligned with those presented in the paper, indicating the MATLAB script provided was likely illustrative, and the authors may have used different hyperparameter values in their experiments. Figures 2 and 3 illustrate the comparisons between the original results and our reproduction.

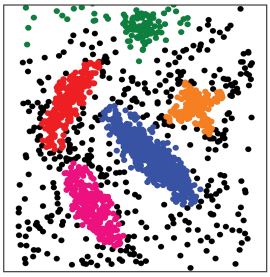
As shown, the point distributions in both the original and our replicated experiments closely resemble the PDFs in Figure 1. Both image pairs (2a vs. 2c, 3a vs. 3c) exhibit slight differences, attributable to minor algorithmic configurations, MDS visualization adjustments not fully detailed by the authors, the stochastic sampling of 1000 points, and using 2000 instead of 4000 points. A similar pattern appears in the DGs, where the blue-



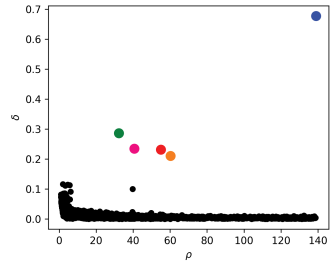
(a) Original point distribution for 4000 samples.



(b) Original decision graph for 4000 samples.

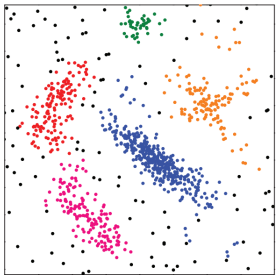


(c) Our point distribution for 2000 samples.

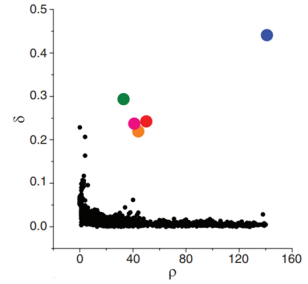


(d) Our decision graph for 2000 samples.

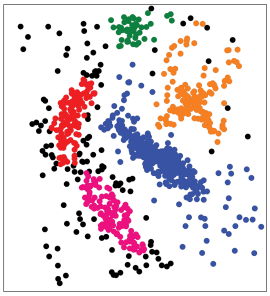
Fig. 2: Comparison of results for the first experiment using 4000 samples (in our case 2000).



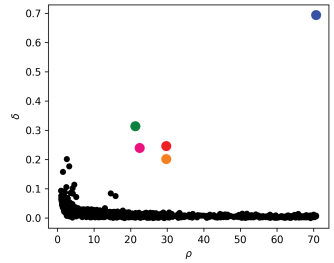
(a) Original point distribution for 1000 samples.



(b) Original decision graph for 1000 samples.



(c) Our point distribution for 1000 samples.



(d) Our decision graph for 1000 samples.

Fig. 3: Comparison of results for the first experiment using 1000 samples.

cluster center notably achieves higher ρ and δ values, while other cluster centers remain distinct from the non-cluster centers (black points), maintaining similar orders of magnitude between the original and replicated results. We can also observe peculiarities in the non-cluster center points, which obtain higher δ values in similar low ρ regions in both scenarios.

Because the authors did not release ground-truth labels, we could not report external metrics such as Adjusted Rand Index (ARI) and Normalized Mutual Information (NMI) for this experiment; likewise, we could not compare internal metrics such as Silhouette Coefficient and Davies-Bouldin Index (DBI) with the original results, as those values were not published. Another replicability issue is that an additional image presented by the authors, depicting the fraction of points assigned to incorrect clusters as a function of sample dimension, could not be replicated since the exact PDFs used by the authors to generate larger datasets was not provided (see Figure 4). Thus, we rely on the authors' claims without an empirical or mathematical verification.

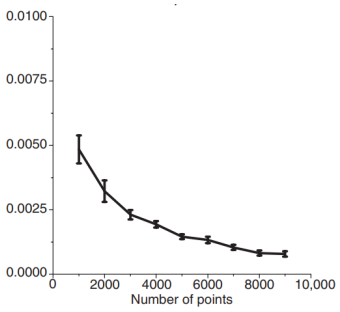


Fig. 4: The fraction of points assigned to the incorrect cluster as a function of the sample dimension. Error bars indicate the standard error of the mean.

B. Experiment 2: Test Cases from the Literature

The following experiment conducted in the paper involves replicating four synthetic point distributions from various papers that introduce new clustering algorithms. The authors selected one point distribution from each paper, evaluated their DPC's performance, and compared it to the results reported by the other papers. However, they did not provide detailed information on the DPC configurations used in each case, so we had to test multiple configurations until obtaining results similar to those in the original paper. We were able to obtain all four datasets while searching for the synthetic point distribution used in the first test case. We found a GitHub repository containing a text file for the first dataset. Fortunately, this file also included a link to a webpage containing datasets for the four synthetic point distributions used in this experiment [14]. These datasets originate from a paper authored by Pasi Fränti, one of the authors from the second test case, in which multiple clustering datasets from different papers were collected, and the performance of K-means was evaluated on six of these benchmark clustering datasets [15], making these datasets publicly available.

1) Aggregation Test Case: The first test case corresponds to a paper introducing a clustering technique based on aggregation [9]. This method takes m clusterings $C_1, \ldots, C_m$ as input and produces a single clustering C that maximally agrees with the input clusterings. For evaluation purposes, they used a dataset consisting of 788 points clearly grouped into seven predominantly elliptical and distinctly separated clusters (see Figure 5). Since the paper does not specify the exact DPC configuration used to obtain their results, we experimented with various configurations until reproducing comparable results. Our final setup used DPC with a Gaussian estimator and 2% neighbor percentage. The original paper mentions achieving comparable results to the initial study whereas other common methods fail. To validate these claims, we implemented the clustering aggregation technique along with a standard clustering approach, specifically K-means [16] from scikit-learn using default hyperparameters, which we used in the next experiments as a baseline. For clustering aggregation, we followed the paper's approach, aggregating clusters obtained from five different clustering methods: single link, complete link, and average link [17], Ward's clustering [18], and K-means. We used the default implementations from scikit-learn for these methods and aggregated their clusters as described in the paper. Figure 5 shows the ground-truth labels alongside the results obtained using our implementations of DPC, clustering aggregation, and standard K-means. As claimed by the authors, DPC produces similar results to clustering aggregation, while common methods like default K-means fail to achieve correct clustering (though adjusting hyperparameters could improve its results).

Although not explicitly presented in the paper, the authors provided for each test case a DG as a validity measure in their supplementary materials. We computed the DG for our DPC implementation and compared it with their provided DG (see Figure 6). As observed, both DGs are highly similar, clearly highlighting the red-cluster center as the most distinguishable with highest ρ and δ values, followed by the yellow and orange cluster centers, while the remaining cluster centers exhibit similar order and structure in both results. Since this validity measure is not applicable to other clustering algorithms,

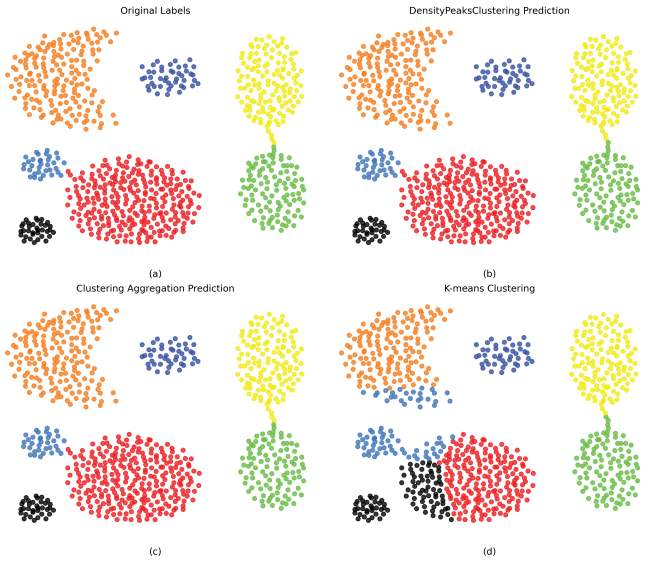
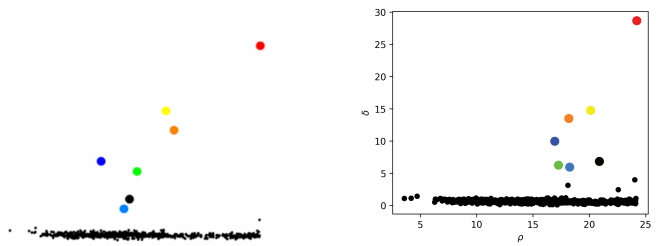


Fig. 5: Clustering results showing **(a)** ground truth labels, **(b)** our DPC implementation, **(c)** our Clustering Aggregation, and **(d)** scikit-learn K-means.

Table I presents alternative quantitative metrics for the three algorithms: ARI, NMI, silhouette score, and DBI. Clustering aggregation achieves perfect ARI and NMI scores, indicating that it clustered correctly all the points; while DPC attained nearly perfect values, both significantly outperforming K-means. For internal evaluation metrics, all three algorithms achieved similar silhouette scores around 0.5, indicating comparable cluster compactness and separation; while K-means exhibited a higher DBI score, indicating inferior clustering quality compared to the other two algorithms.



(a) Original decision graph. (b) Our decision graph.

Fig. 6: Comparison of decision graphs (Aggregation test).

TABLE I: Performance metrics comparison for the Aggregation test case.

Algorithm	ARI	NMI	Silhouette	DBI
DPC	0.9978	0.9957	0.4932	0.5036
Clustering Aggregation	1.0	1.0	0.4925	0.5036
K-means	0.7387	0.8397	0.4831	0.6993

2) DPCV Test Case: The second test case corresponds to a paper introducing a clustering technique based on removing clusters iteratively one by one until the desired number of clusters is reached [19]. For evaluation purposes, they used a dataset consisting of 5000 points with 15 predefined clusters highly overlapped between them (see Figure 7). The authors stated that DPC "successfully determines the cluster structure of the data set". To verify this claim, we clustered the dataset with our DPC implementation and compared our results with theirs. However, the public dataset was updated a year after the paper was published, and the locations and labels of some points may have changed, which might cause our obtained results to differ slightly with those from the paper. Moreover, the current ground truth of the labels is incorrect, which prevents us from knowing the true assignment of points to clusters and from calculating the external clustering metrics ARI and NMI to validate our results.

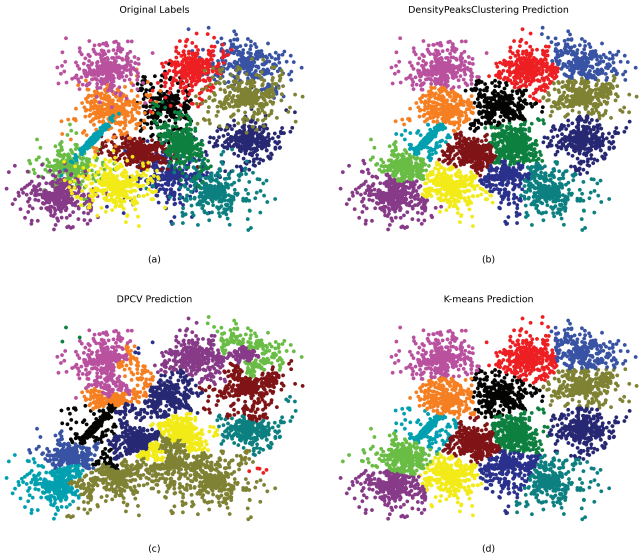


Fig. 7: Clustering results showing **(a) corrupted** ground truth labels, **(b)** our Density Peaks Clustering, **(c)** our DPCV implementation, and **(d)** scikit-learn K-means.

Therefore, in order to compare the results from the paper and our own with other reliable results, we implemented two additional algorithms to use in this dataset for their evaluation. Unfortunately, the paper [19] does not provide any numerical or graphical results for this dataset, so we could not use it for the comparison. Instead, we used an algorithm that improves upon DPC, proposed by [3]: Density Peaks Clustering based on Variance (DPCV). This algorithm replaces the Euclidean distance metric by using variances. We faithfully implemented DPCV based on our DPC algorithm and incorporated the improvements following the detailed

instructions in the DPCV paper. The other algorithm we used is the standard K-means, which we already mentioned was be used as a baseline across all test cases.

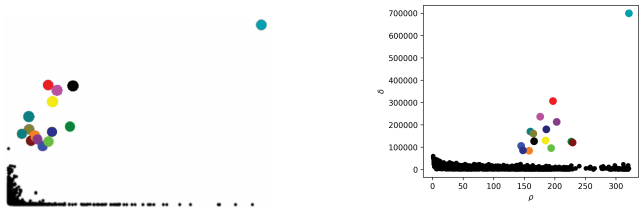
Figure 7 shows the label assignments for each of the three algorithms. As seen in Fig. 7(a), the ground truth labels are corrupted because some points are not correctly assigned to their respective clusters, which prevents us from computing valid external metrics; nevertheless, we have included the corrupted ARI and NMI to provide an approximate evaluation. DPC achieves a proper assignment that visually produces coherent clusters, correctly identifying all 15 clusters. This is reflected in the high silhouette and low DBI scores shown in Table II. DPC even corrects some of the point misassignments present in the corrupted ground truth. The DPC configuration used was `n_clusters=15`, `percent=2.5`, and `density_estimator='cutoff'`, found by performing a simple manual hyperparameter search. By contrast, DPCV struggles to produce a reliable clustering on this dataset, likely because its variance-based distance metric performs poorly when many clusters exhibit similar variances. This causes the algorithm to merge clusters with comparable variance, most notably the brown cluster which consolidates three distinct clusters, and to create small clusters of points at the borders of adjacent clusters, such as the red cluster (bottom-right) and dark-green cluster (top-left). We implemented DPCV using the optimal settings reported in [3] for this dataset: `n_clusters=15` and `n_neighbors=1`. Surprisingly, K-means also yields good clustering results, with performance comparable to DPC. We explored various hyperparameter configurations and found that the optimal setup was `n_clusters=15`, `random_state=42`, and `n_init=10`. As shown in Table II, the silhouette score for DPC and K-means is high while the DBI score is low, indicating that both algorithms produce compact clusters.

TABLE II: Clustering performance of DPC, DPCV, and K-means on the evaluated dataset.

Algorithm	ARI*	NMI*	Silhouette	DBI
DensityPeaks	0.7284	0.7975	0.4847	0.6541
DPCV	0.4875	0.6897	0.2091	0.9386
K-means	0.7278	0.7970	0.4923	0.6470

** Estimated values. The ARI and NMI scores may not reflect true clustering quality due to corrupted ground truth labels.*

As in this test case the external metrics did not provide reliable information, we show both the original and our DGs for DPC in Figure 8 to better visualize the cluster-center selection criterion. The turquoise cluster-center point corresponding to the bridge-shaped cluster is the most clearly identifiable, followed by the red cluster-center corresponding to the north-central region, and then the purple, black, yellow, and pink cluster-centers, which are closely grouped and exhibit very similar γ values. Additionally, many non-cluster-center points obtain high δ values despite having low ρ values, some even achieving higher δ values than actual cluster centers due to being farther away from them. These are most likely the isolated top-left purple and bottom-right green points, both located far from their respective cluster centers. Therefore, we can argue that in this dataset, only 5 out of the 15 clusters appear to have sufficiently high γ values to be clearly identified as cluster centers.



(a) Original decision graph. (b) Our decision graph.

Fig. 8: Comparison of decision graphs (DPCV test).

3) FLAME Test Case: The third test case corresponds to a paper introducing Fuzzy Clustering by Local Approximation of Membership (FLAME) [10]. They utilized a dataset comprising 240 points distributed across two distinct clusters: one forming a compact, dense grouping in the upper-central region, and another forming a broader, sparser, and curved structure resembling a U-shape in the lower part of the space (see Figure 9). Since the original paper did not specify the exact DPC configuration used to achieve their results, we experimented with various configurations until obtaining comparable outcomes. Our final configuration employed DPC with a Gaussian estimator and 3% neighbor percentage. The original paper mentions achieving *"results comparable to the original method."* To validate this claim, we implemented our own version of the FLAME algorithm based on the instructions from [10], and supplemented by third-party code. As the FLAME paper also omitted their exact hyperparameter configurations, we performed an extensive Bayesian optimization search targeting the ARI metric to closely align with the ground truth labels. The optimal configuration identified for FLAME was `metric='minkowski'`, `cluster_neighbors=20`, `iteration_neighbors=20`, `max_iter=2000`,

`eps=1.6e-05`, and `thd=-2.2`. Additionally, we again used scikit-learn's default K-means implementation as a baseline.

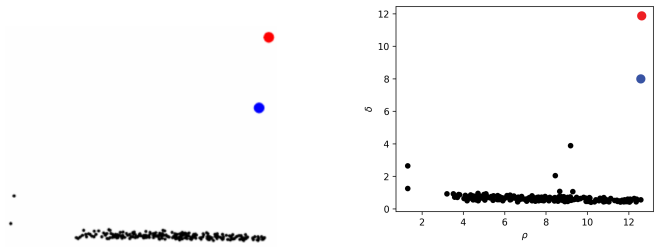
Fig. 9: Clustering results showing **(a)** ground truth labels, **(b)** our Density Peaks Clustering, **(c)** our FLAME implementation, and **(d)** scikit-learn K-means.

An interesting finding was that the original FLAME paper presented a perfect cluster mapping, while our implementation did not achieve this exact mapping, possibly due to minor differences in the code or the inability of Bayesian optimization to discover the exact hyperparameters despite the extensive search. Figure 9 demonstrates how our DPC results accurately respect the ground truth, achieving perfect ARI and NMI scores (see Table III). In contrast, FLAME and K-means underperformed by approximately 0.3 and 0.6 points, respectively. Regarding internal validation metrics, the silhouette score was similar across all three algorithms. However, FLAME obtained a better DBI score of 0.7242 compared to about 1.1 for DPC and K-means. This improvement was mainly due to FLAME identifying two outlier points (marked with black x symbols in the upper-left corner), enhancing the compactness and separation of the blue cluster. While K-means labeling resembled FLAME's labeling, it misclassified more points in specific areas, notably the left arm of the lower semi-circular cluster. Returning to the paper's claim, our results confirm that DPC indeed achieves comparable performance to the original FLAME method, accurately mapping points to their respective clusters.

The original DG was compared with our DG in Figure 10. Both plots highlight the red cluster as possessing the highest ρ and δ values, while the blue cluster trails slightly behind. The two outliers appear distinctly in the bottom-left area in both plots, characterized by extremely low ρ values due to their isolated locations and higher δ values from their relative proximity to the blue-cluster center. This further supports the close similarity between our replicated results and the provided results from the original DPC paper.

TABLE III: Performance metrics comparison for the FLAME test case.

Algorithm	ARI	NMI	Silhouette	DBI
DPC	1.0	1.0	0.3285	1.1605
FLAME	0.7109	0.7257	0.3308	0.7242
K-means	0.4649	0.4267	0.3706	1.1178



(a) Original decision graph. (b) Our decision graph.

Fig. 10: Comparison of decision graphs (FLAME test).

4) SPIRAL Test Case: The last and fourth test case corresponds to a paper introducing a robust path-based spectral clustering [11]. They utilized a dataset comprising 312 points grouped into three spiral-shaped clusters, with each spiral arm representing a distinct cluster (see Figure 11). As the DPC paper did not specify the exact configuration used to achieve their results, we experimented with various configurations until obtaining comparable outcomes. Our final configuration employed DPC with a Gaussian estimator and 3% neighbor percentage. The original paper mentions their algorithm *"correctly identifies the three clusters without generating a connectivity graph"*. Although we intended to implement our own version of the robust path-based clustering algorithm following the instructions provided in [11], the paper lacked key steps and detailed configurations necessary for an accurate replication. Due to these uncertainties, we instead chose Spectral Clustering from scikit-learn as it was the model that most closely resembles the presented path-based algorithm.

Unlike the method proposed in the original paper, which achieved perfect labeling, our Bayesian optimization-tuned Spectral Clustering implementation did not correctly label all points. Interestingly, our Spectral Clustering results closely matched the Spectral Clustering results reported by [11]'s authors. The optimal parameters for our Spectral Clustering were

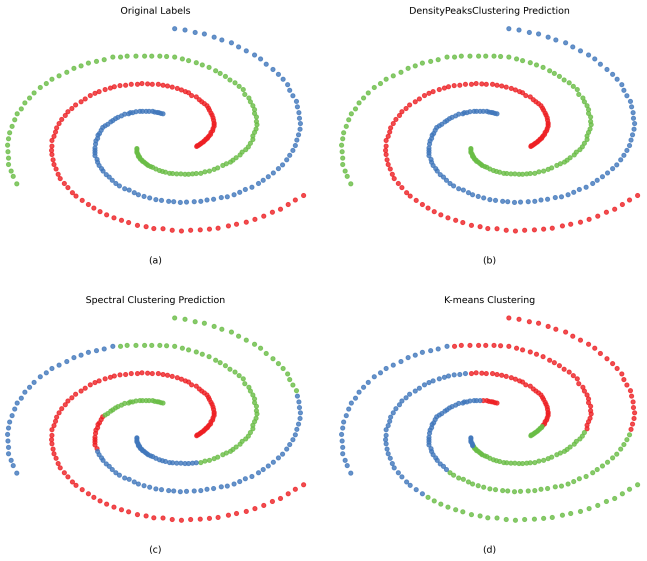


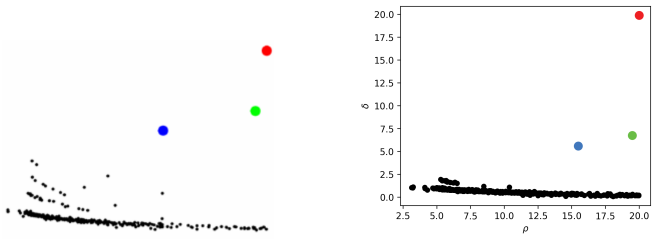
Fig. 11: Clustering results showing **(a)** ground truth labels, **(b)** our Density Peaks Clustering, **(c)** our Spectral-Clustering implementation, and **(d)** scikit-learn K-means.

`affinity=nearest_neighbors`, `gamma=0.34`, `n_neighbors=5`, `assign_labels=kmeans`, and `eigen_tol=1.59`. Figure 11 displays the original labels alongside results from our implementations of DPC, Spectral Clustering, and default K-means. DPC again perfectly labeled all the points, achieving perfect ARI and NMI scores (see Table IV). Spectral Clustering, however, struggled with label assignment, incorrectly labeling portions of two spiral arms. K-means clustered based on spatial regions (west, northeast, and south), identifying cluster centers in the middle of these regions, resulting in ARI and NMI scores close to zero. DPC exhibited the lowest silhouette score, indicating the least compact cluster structure, as expected given the inherently non-compact geometry of these datasets. Conversely, K-means obtained the lowest DBI, indicative of the tightest and most well-separated clusters, due to its partitioning by spatial regions rather than adherence to the dataset's spiral morphology. These observations confirm the authors' claim about DPC effectively identifying the correct cluster labeling.

TABLE IV: Performance metrics comparison for the SPIRAL test case.

Algorithm	ARI	NMI	Silhouette	DBI
DPC	1.0	1.0	0.0013	5.882
Spectral	0.4553	0.4978	0.0349	10.3284
K-means	-0.006	0.0004	0.3604	0.8858

The original DG was compared with our DG results, again demonstrating high similarity (see Figure 12). The red-cluster center achieved the highest ρ and δ values, closely followed by the green and blue-cluster centers. Notably, some points (corresponding to the spiral arm endpoints) displayed low ρ but elevated δ values in the lower-left section of both DGs, indicating their isolated locations and increased distance from cluster centers.



(a) Original decision graph. (b) Our decision graph.

Fig. 12: Comparison of decision graphs (SPIRAL test).

C. Experiment 3: Olivetti Face Database

The third experiment presented in the paper involves clustering images from the Olivetti faces dataset. We clustered 100 different images extracted from 10 individuals, with each person corresponding to one cluster containing 10 images. We selected the first 100 images from the total 400 images available in the dataset. For this experiment, we needed to implement an additional step: using the Complex-Wavelet Structural Similarity Index (CW-SSIM) [20] as the similarity metric for the images, as the paper specified that *"The similarity between two images was computed by following ([20])."* We implemented the CW-SSIM similarity metric closely following the methodology described in the original CW-SSIM paper and by referencing an available MATLAB CW-SSIM implementation online [21]. In this case, the authors were more explicit regarding the hyperparameters used, stating: *"The density is estimated by a Gaussian kernel with variance* $d_c = 0.07$*"*; therefore, we adopted this configuration in our experiments. Furthermore, the authors described a more restrictive criterion for assigning images to clusters in this experiment, which we also incorporated into our implementation:

> "For such a small set, the density estimator is unavoidably affected by large statistical errors; thus, we assign images to a cluster following a slightly more restrictive criterion than in the preceding examples. An image is assigned to the same cluster of its nearest image with higher density only if their distance is smaller than d_c. As a consequence, images farther than d_c from any other image of higher density remain unassigned."

However, the paper did not provide details regarding the hyperparameters required for CW-SSIM; as a result, we adopted the default parameters proposed in the original CW-SSIM paper.

Figure 13 presents the original DPC results for the Olivetti faces alongside our three DPC implementations. Figure 13B displays the results of our DPC using the CW-SSIM similarity metric without the restrictive configuration; Figure 13C shows our DPC results using the SSIM similarity metric from scikit-learn without the restrictive configuration; and Figure 13D presents our DPC results using CW-SSIM with the restrictive configuration.

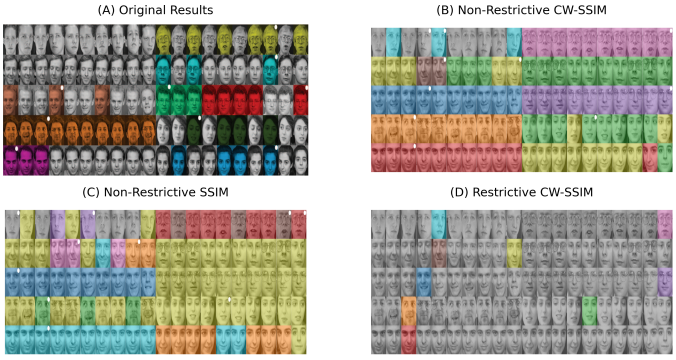
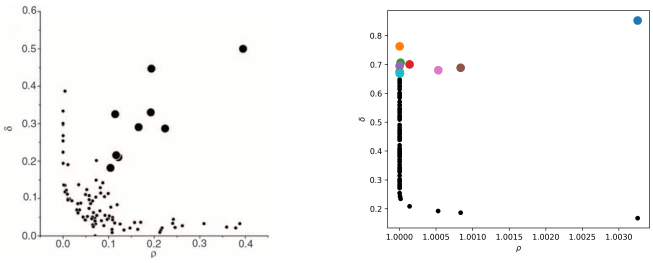


Fig. 13: Original and replicated DPC results for the Olivetti faces dataset.

In this experiment, we obtained results significantly different from those in the paper. Comparing the original results in Figure 13A to our replication in Figure 13D, we attribute this discrepancy primarily to differences in the custom CW-SSIM implementation. Specifically, our implementation of the restrictive criterion appears to be more stringent, resulting in fewer images being assigned to clusters. Nevertheless, both implementations successfully identified one face per individual and avoided assigning faces from different people into the same cluster.

When not applying the restrictive criterion, Figures 13B and 13C demonstrate that more images are assigned to clusters. Using CW-SSIM we achieve more homogeneous clustering, correctly identifying all faces from the pink, blue, and orange clusters, and almost correctly identifying the red cluster with only one misclassification. However, the green and yellow clusters contained faces from multiple individuals due to similarities among these faces. Additionally, the top-left light-blue image faces only has three images assigned due to the challenges posed by their varying face orientations. On the other hand, the SSIM metric performed notably worse, clustering most faces into the yellow cluster, though it did correctly assign all faces in the blue cluster and the light-blue cluster, albeit misclassifying two images from the orange cluster. Given these outcomes, we conclude that the authors' choice of a custom CW-SSIM similarity metric over a more conventional SSIM metric from scikit-learn significantly improved their clustering performance.

Figure 14 illustrates both the original and our DGs for this experiment. In our DG, we colored points to clearly identify cluster centers, a detail the original DG did not provide. The blue cluster, corresponding to the correct assignment of all ten faces of a single individual, is the most clearly identified cluster center. The next best-defined clusters are the brown cluster (containing only two faces in the second row) and the orange cluster (containing ten correctly assigned faces from another single individual). Although the orange cluster accurately identifies all faces of one individual without incorrect predictions, it is not as distinctly identified as the blue cluster. This distinction arises from the δ parameter, representing the distance to other cluster centers. Specifically, the face selected as the blue-cluster center (indicated by the white dot in Figure 13) is significantly different from other images, while the face selected as the orange-cluster center closely resembles images from the red and green clusters, which also have very similar γ values. Additionally, the structure of both decision graphs differs significantly. In our case, the decision graph assigns most points with low ρ, indicating that all faces exhibit low similarity scores, suggesting they are highly dissimilar from one another. This behavior is likely attributable to the properties of the CW-SSIM metric. In contrast, this issue did not arise in the original results, possibly due to a different implementation or specific tuning of the CW-SSIM algorithm, which was not documented in the paper.



(a) Original decision graph. (b) Our decision graph.

Fig. 14: Original and replicated decision graphs for the Olivetti faces experiment.

In this case, we cannot support the claim made by the paper regarding superior performance in this experiment, since we could not precisely replicate the configuration or results presented in the paper. We attempted to contact

the original authors for additional information about this and other experiments; but, unfortunately, we did not receive a response.

D. Experiment 4: Comparison with Other Algorithms

In this final experiment, we experimentally compared the DPC algorithm with selected scikit-learn algorithms, chosen due to their relevance and distinct methodological approaches. Additionally, we assessed these algorithms on new datasets not previously examined to evaluate DPC's performance across diverse and widely recognized datasets.

The algorithms selected for comparison are:

- **K-means**: A popular clustering algorithm that partitions data into k clusters by minimizing intra-cluster variance. It's effective when clusters are spherical and similarly sized. By comparing it with DPC, which does not impose strict assumptions on cluster shapes, we can highlight scenarios where centroid-based approaches may struggle.
- **DBSCAN**: Density-Based Spatial Clustering of Applications with Noise clusters data based on density. Both DPC and DBSCAN are density-based, yet they differ in their cluster-center determination. Comparing these highlights how density criteria and parameter sensitivity (such as DBSCAN's ϵ versus DPC's ρ and δ) influence clustering outcomes.
- **Spectral Clustering**: Utilizes eigen-decomposition of an affinity matrix for dimensionality reduction before clustering, effectively capturing complex, non-convex structures.
- **GMM**: Gaussian Mixture Models represents clusters as Gaussian distributions, providing soft probabilistic assignments. Comparing GMM's probabilistic assignments to DPC's hard assignments helps reveal strengths and weaknesses in modeling cluster overlap and shape.

These algorithms were evaluated on five distinct datasets:

- **Iris Dataset**: 150 samples, 4 features, 3 slightly overlapping clusters; a benchmark for basic cluster validation.
- **Wine Dataset**: 178 samples, 13 features, 3 moderately overlapping clusters; tests algorithm robustness on higher-dimensional data.
- **Digits Dataset**: 1,797 samples, 64 features, 10 clusters representing handwritten digits; tests scaling and cluster separability.
- **Synthetic Moons (make_moons)**: 2D dataset with two interleaving half-circles; challenging for centroid-based methods.
- **Synthetic Blobs (make_blobs)**: Clearly separated spherical clusters; serves as a baseline scenario ideal for centroid-based algorithms.

Evaluation metrics included two external metrics: Adjusted Rand Index (ARI), and Normalized Mutual Information (NMI); and two internal metrics: silhouette score, and Davies–Bouldin Index (DBI). Decision Graphs (DGs) for DPC were also used to visually determine cluster centers.

Hyperparameter tuning for each algorithm was performed for every dataset using Bayesian optimization over 20 iterations with ARI as the objective metric. The hyperparameter search spaces were:

- **DPC**: percent $\in [0.5, 5.0]$, density_estimator $\in \{$cutoff, gaussian$\}$.
- **K-means**: n_init $\in [5, 15]$, init $\in \{$k-means++, random$\}$.
- **DBSCAN**: $\varepsilon \in [0.1, 2.0]$, min_samples $\in [2, 10]$.
- **Spectral Clustering**: $\gamma \in [0.01, 10.0]$, assign_labels $\in \{$kmeans, discretize$\}$, affinity $\in \{$nearest_neighbors, rbf$\}$.
- **GMM**: covariance_type $\in \{$full, tied, diag, spherical$\}$, init_params $\in \{$kmeans, random$\}$, max_iter $\in [50, 200]$.

Figures 15, 17, 18, 19, and 20 present the ground truth and clustering results in two dimensions. For datasets with more than two features: Iris, Wine, and Digits; dimensionality was reduced using t-SNE [22]. Tables V, VI, VII, VIII, and IX show the clustering metrics for each dataset across the five algorithms. The Appendix provides the optimal hyperparameter configurations identified via Bayesian optimization for each dataset and algorithm.

We loaded the datasets from the scikit-learn library. For datasets with more than two features (Iris, Wine, and Digits), we applied feature standardization using Standard Scaler to ensure consistency in scale across features. Additionally, for the Digits dataset, which contains 64 features, Principal Component Analysis (PCA) [23] was applied to reduce dimensionality while preserving 95% of the variance prior to clustering. In contrast, the Moons and Blobs datasets are synthetically generated with only two features and therefore did not require any preprocessing or dimensionality reduction.

We now analyze the results for each dataset individually, comparing metrics across algorithms and exploring how the dataset characteristics influence the results.

On the Iris dataset, DPC consistently achieves the highest external metrics, clearly outperforming the other algorithms. With one distinct cluster and two overlapping clusters, DPC successfully identifies correct cluster

centers. Figure 15 demonstrates how closely DPC aligns with ground truth labels. The Iris DG in Figure 16 shows the purple-cluster center having the highest $\gamma = \rho \times \delta$ value, followed by the overlapping green and yellow cluster centers. K-means and Spectral Clustering perform moderately well, while DBSCAN and GMM struggle due to density variations and overlapping clusters, respectively.

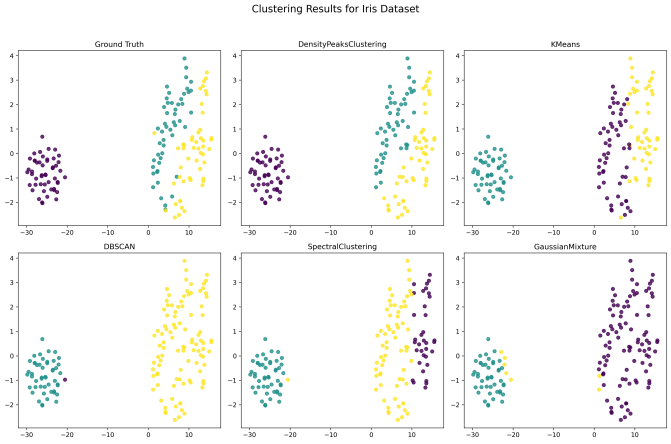


Fig. 15: Clustering performance of the five evaluated algorithms on the Iris dataset.

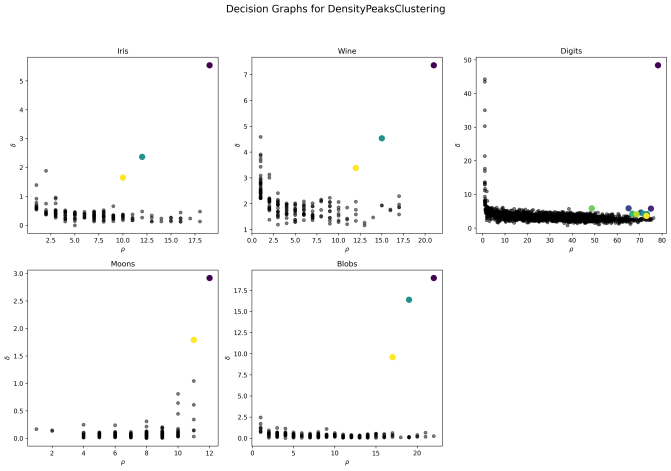


Fig. 16: DPC decision graphs for each benchmark dataset.

A deeper look at Table V reveals how each algorithm's built-in assumptions interact with the intrinsic geometry of the Iris data. DPC makes no assumptions on cluster shape or size and instead locates peaks in the density of the points. This enables it to properly cluster both the compact purple cluster, and both elongated and overlapping green and yellow clusters with high ARI (0.82) and NMI (0.80). By contrast, K-means enforces spherical clusters with equal variance, an assumption that is violated by the stretched shape of the green and yellow clusters. As a result, K-means only attains a moderate ARI (0.62) and NMI (0.66), confusing and misclassifying some points at both the upper and lower ends of the elongated clusters.

TABLE V: Clustering performance on the Iris dataset across different algorithms.

Model	ARI	NMI	Silhouette	DBI
DPC	0.82	0.80	0.37	1.14
K-means	0.62	0.66	0.46	0.83
DBSCAN	0.56	0.73	0.59	0.58
Spectral Clustering	0.65	0.68	0.46	0.82
GMM	0.52	0.66	0.48	0.89

DBSCAN, which clusters based on density connectivity rather than distance to centroids, is well-suited to non-spherical shapes and produces the best internal separation with highest Silhouette (0.59) and lowest DBI (0.58). However, because the optimal ϵ-neighborhood that cleanly isolates the purple cluster is too small to distinguish the lower-density border region between yellow and green clusters, DBSCAN merges and labels many boundary points incorrectly, yielding only an ARI of 0.56. Spectral Clustering, using a graph Laplacian to capture manifold structure, overcomes some shape constraints of purely distance-based methods (ARI=0.65, NMI=0.68) but remains sensitive to the choice of the similarity kernel, reflected in its poor internal scores (Silhouette=0.46, DBI=0.82). Finally, GMM fits each cluster to an elliptical Gaussian distribution, assuming that all clusters have similar spread and follow a normal distribution. It models the purple cluster well (Silhouette=0.48) but fails on skewed and overlapping clusters, leading to the lowest ARI (0.52).

Overall, these results highlight the importance of matching model assumptions such as sphericity, density uniformity, and Gaussianity, to the actual size, shape, and density characteristics of the data when selecting a clustering algorithm.

On the Wine dataset (Figure 17, Table VI), Spectral Clustering and K-means achieved the highest external metrics (ARI=0.95 and 0.90; NMI=0.93 and 0.88) as well as the best cluster cohesion and compactness (Silhouette $\approx$ 0.28, DBI $\approx$ 1.39). This performance is likely due to the three clusters being similar sized with spherical shape, perfectly suiting centroid-based methods. DPC followed with moderate external metrics (ARI=0.77, NMI=0.77) and near-identical Silhouette (0.28) and DBI (1.39) scores. However, it struggled with points near class boundaries with low density zones, which led to occasional misassignments that slightly reduced both external metrics. Figure 16 illustrates how

DPC clearly identified cluster centers and the presence of isolated boundary points with high δ that might have caused some misassignments.

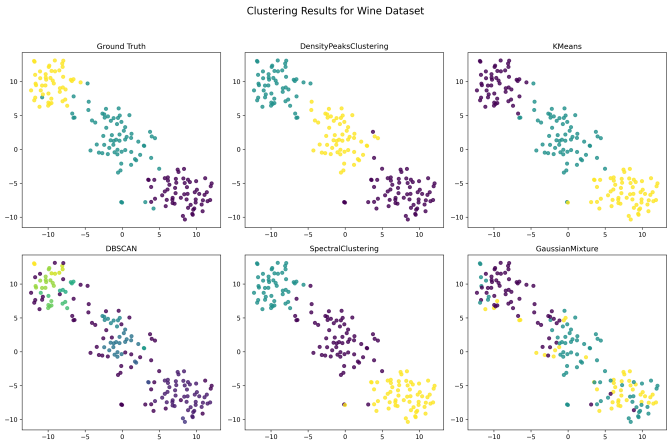


Fig. 17: Clustering performance of the five evaluated algorithms on the Wine dataset.

TABLE VI: Clustering performance on the Wine dataset across different algorithms.

Model	ARI	NMI	Silhouette	DBI
DPC	0.77	0.77	0.28	1.39
K-means	0.90	0.88	0.28	1.39
DBSCAN	0.65	0.70	0.19	1.22
Spectral Clustering	0.95	0.93	0.29	1.39
GMM	0.21	0.24	0.09	2.87

By contrast, while DBSCAN yields the lowest DBI (1.22), indicative of tight high-density regions, it suffers from reduced overall cohesion (Silhouette=0.19) and ARI/NMI around 0.65/0.70 due to its sensitivity to the ϵ-neighborhood settings in moderately dense areas. GMM performs poorly across all metrics (ARI=0.21, NMI=0.24; Silhouette=0.09; DBI=2.87), as its strict Gaussian and equal-covariance assumptions are violated by the elongated, skewed distributions present in the wine features.

On the Digits dataset (Figure 18, Table VII), Spectral Clustering achieves the highest external metrics (ARI=0.70, NMI=0.82), followed by DPC (ARI=0.62, NMI=0.74). K-means (ARI=0.48, NMI=0.63) and GMM (ARI=0.50, NMI=0.64) achieve only moderate clustering performance, as their assumptions about spherical shape and Gaussian-covariance are violated by the tightly packed and irregular digit clusters. DBSCAN fails entirely to identify the ten classes (ARI=0.00, NMI=0.00), with its global ϵ-neighborhood criteria collapsing most points into a single purple cluster. The density graph in Figure 16 shows only one clear peak corresponding to the purple-cluster center, explaining why DPC recovers the most distinct digit class but mislabels the others.

Looking at internal indices, DBSCAN achieves the best Silhouette (0.28) and lowest DBI (1.23) by isolating the densest local region, yet these clusters bear no relation to the true digit labels. K-means, Spectral Clustering, and GMM each achieve modest cohesion (Silhouette ≈ 0.15) and average compactness (DBI > 1.8), reflecting their inability to adapt fully to the complex manifold structure; while DPC exhibits the poorest internal separation (Silhouette=0.03, DBI=2.75).

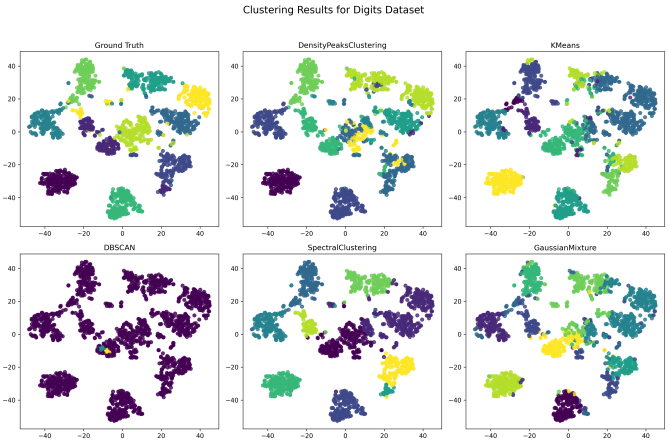


Fig. 18: Clustering performance of the five evaluated algorithms on the Digits dataset.

TABLE VII: Clustering performance on the Digits dataset across different algorithms.

Model	ARI	NMI	Silhouette	DBI
DPC	0.62	0.74	0.03	2.75
K-means	0.48	0.63	0.16	1.80
DBSCAN	0.00	0.00	0.28	1.23
Spectral Clustering	0.70	0.82	0.15	1.87
GMM	0.50	0.64	0.14	2.81

On the Moons dataset (Figure 19, Table VIII), DBSCAN and Spectral Clustering both achieve perfect external metrics (ARI=1.00, NMI=1.00) by accurately identifying the two crescent shapes without imposing linear and spherical boundaries. In contrast, K-means (ARI=0.22, NMI=0.17) and GMM (ARI=0.47, NMI=0.38) fail to capture the non-convex geometry, misclassifying both crescent ends in the central region. As a result, K-Means attains the highest Silhouette score (0.48) and lowest DBI (0.79) by partitioning the data into more compact and evenly distributed clusters, even though the clustering does not reflect the true structure. DPC struggled with points near the center, where the two clusters come into close proximity, leading to misclassifications in which parts of the purple arm were assigned to

the yellow cluster. These regions are closer to the yellow-cluster center than to the purple one, resulting in lower δ values. In addition to their lower point density yielding reduced ρ scores, these points also obtained very low γ values, which likely contributed to the misassignments (see DG in Figure 16).

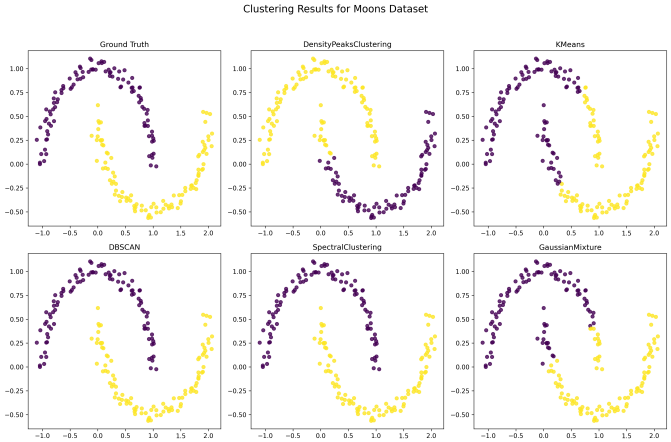


Fig. 19: Clustering performance of the five evaluated algorithms on the Moons dataset.

TABLE VIII: Clustering performance on the Moons dataset across different algorithms.

Model	ARI	NMI	Silhouette	DBI
DPC	0.72	0.68	0.40	0.95
K-means	0.22	0.17	0.48	0.79
DBSCAN	1.00	1.00	0.32	1.18
Spectral Clustering	1.00	1.00	0.32	1.18
GMM	0.47	0.38	0.46	0.83

Lastly, on the Blobs dataset (Figure 20, Table IX), all five algorithms achieve perfect external metrics (ARI=1.00, NMI=1.00) and identical internal metrics (Silhouette=0.85, DBI=0.22) due to the clear separation and the uniform size and density of the clusters. The DG in Figure 16 clearly shows three distinct density peaks corresponding to the three blob cluster centers, highlighting the simplicity and well-structured nature of this dataset.

TABLE IX: Clustering performance on the Blobs dataset across different algorithms.

Model	ARI	NMI	Silhouette	DBI
DPC	1.00	1.00	0.85	0.22
K-means	1.00	1.00	0.85	0.22
DBSCAN	1.00	1.00	0.85	0.22
Spectral Clustering	1.00	1.00	0.85	0.22
GMM	1.00	1.00	0.85	0.22

Overall, across these five benchmark datasets, from simple isotropic blobs to overlapping Gaussian clusters, non-convex moons, and high-dimensional digits,

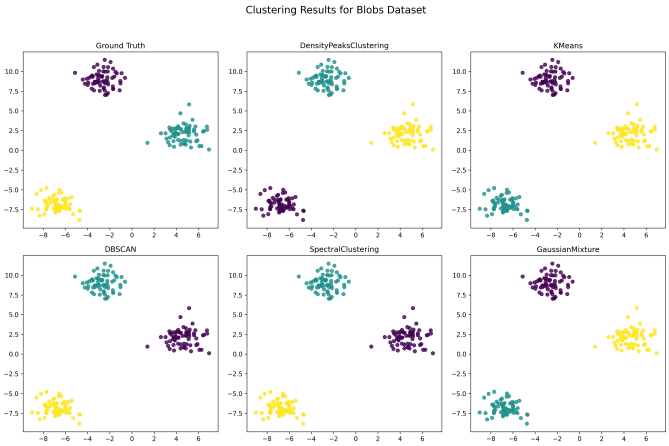


Fig. 20: Clustering performance of the five evaluated algorithms on the Blobs dataset.

Density Peaks Clustering (DPC) demonstrated a consistent, robust and model-agnostic performance. It perfectly clusters clear and equally sized blobs and excels on moderately overlapping structures such as Iris and Wine, while it remains competitive on non-linear (Moons) and high-dimensional (Digits) data; only conceding occasional misassignments in regions of ambiguous density. Compared to K-means and GMM, which struggle with non-convex and skewed shapes; and DBSCAN, which either over-merges or requires delicate ϵ-tuning; DPC produces an effective balance between shape flexibility and noise resilience, performing on par with Spectral Clustering where capturing the structure and shape of the data is especially important.

IV. Discussion

In this work, we have conducted numerous experiments with Density Peaks Clustering (DPC) and evaluated its performance in comparison with a variety of algorithms of differing nature on diverse datasets. We observed that DPC can produce high-quality clusterings across a range of different scenarios: from simple datasets with highly compact and clearly separated clusters, achieving perfect results in the Blobs case (see Figure 20); to datasets with increased number of clusters and overlap while maintaining excellent performance (Figures **??**); as well as more challenging benchmarks such as Olivetti Faces (Figure 13) and non-convex shapes like Spiral (Figure 11).

We were able to replicate with high fidelity many of the experiments from the original paper, configuring the hyperparameters of DPC and the comparison algorithms to match the authors' specifications and reported outcomes. In some cases, however, we could not reproduce certain experiments due to insufficient detail

regarding DPC configuration and preprocessing steps on datasets such as the Olivetti Faces or the DPCV test case. To strengthen the objectivity and robustness of our evaluation, we introduced additional metrics (ARI, NMI, Silhouette, and DBI) not provided by the original authors. We also included the Decision Graphs (DGs) proposed by the authors, which plot local density ρ against the minimum distance to higher-density points δ, offering a direct visualization of DPC's cluster-center selection criterion.

Future work should compare the performance of the original DPC algorithm with several of its improvements presented in the literature, such as DPCV and MDDPC [3], CFSFDP-HD [4], SNN-DPC [6], FKNN-DPC [2], among others. These enhancements should be selected to cover different areas of improvement—such as scalability and high-dimensional data handling—and evaluated to determine their impact on clustering effectiveness in both Decision Graph analyses and standard internal and external metrics. Additionally, future studies should apply these comparisons to new datasets with varying shapes, cluster counts, probability distributions, sample sizes, and degrees of overlap.

V. Conclusions

In this work, we replicated and evaluated the implementation and experiments of the Density Peaks Clustering (DPC) algorithm proposed in *Clustering by Fast Search and Find of Density Peaks* [1]. We faithfully re-implemented DPC following the detailed steps described in the original paper. However, we identified several omissions in the presentation of results and claims: the authors do not specify the exact hyperparameters used to generate their figures, nor do they report common clustering metrics such as Adjusted Rand Index (ARI), Normalized Mutual Information (NMI), Silhouette score, or Davies–Bouldin Index (DBI).

Despite these gaps, we successfully reproduced all of the paper's core experiments and provided complete details on dataset preprocessing, hyperparameter settings, and the additional metrics we computed. We exactly replicated the first experiment and the Aggregation, FLAME, and SPIRAL test cases from the second experiment. In contrast, we were unable to reproduce the DPCV test case with full fidelity due to having a modified dataset and missing key configuration instructions; nonetheless, our visual and quantitative results closely resemble those reported in the paper. Similarly, the Olivetti Faces experiment could not be matched precisely because of insufficient detail on the CW-SSIM similarity metric, preprocessing steps, and DPC configuration.

To further assess the generality of DPC, we conducted additional experiments not present in the original paper, comparing DPC with several established clustering algorithms on diverse datasets. These extended evaluations demonstrate DPC's robustness and competitive performance across a wide range of scenarios. We conclude that, while the original paper's results and claims are largely validated by our findings, greater transparency in hyperparameter selection and the inclusion of standardized evaluation metrics would greatly enhance reproducibility and comparability.

References

[1] A. Rodriguez and A. Laio, "Clustering by fast search and find of density peaks," *Science*, vol. 344, no. 6191, pp. 1492–1496, 2014. [Online]. Available: https://www.science.org/doi/abs/10.1126/science.1242072

[2] J. Xie, H. Gao, W. Xie, X. Liu, and P. W. Grant, "Robust clustering by detecting density peaks and assigning points based on fuzzy weighted k-nearest neighbors," *Information Sciences*, vol. 354, pp. 19–40, 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S002002551630158X

[3] S. Ding, W. Du, C. Li, X. Xu, L. Wang, and L. Ding, "Density peaks clustering algorithm based on improved similarity and allocation strategy," *International Journal of Machine Learning and Cybernetics*, vol. 14, no. 4, pp. 1527–1542, 2023. [Online]. Available: https://doi.org/10.1007/s13042-022-01711-7

[4] R. Mehmood, G. Zhang, R. Bie, H. Dawood, and H. Ahmad, "Clustering by fast search and find of density peaks via heat diffusion," *Neurocomputing*, vol. 208, pp. 210–217, 2016, sI: BridgingSemantic. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0925231216304829

[5] C. Li, G. Ding, D. Wang, Y. Li, and S. Wang, "Clustering by fast search and find of density peaks with data field," *Chinese Journal of Electronics*, vol. 25, pp. 397–402, 05 2016.

[6] R. Liu, H. Wang, and X. Yu, "Shared-nearest-neighbor-based clustering by fast search and find of density peaks," *Information Sciences*, vol. 450, pp. 200–226, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0020025518302093

[7] S. Ruan, R. Mehmood, A. Daud, H. Dawood, and J. S. Alowibdi, "An adaptive method for clustering by fast search-and-find of density peaks: Adaptive-dp." Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2017, p. 119–127. [Online]. Available: https://doi.org/10.1145/3041021.3054148

[8] J. Chengheng and L. Yongmei, "Parallel clustering by fast search and find of density peaks," in *2016 International Conference on Audio, Language and Image Processing (ICALIP)*, 2016, pp. 563–567.

[9] A. Gionis, H. Mannila, and P. Tsaparas, "Clustering aggregation," *ACM Trans. Knowl. Discov. Data*, vol. 1, no. 1, p. 4–es, Mar. 2007. [Online]. Available: https://doi-org.recursos.biblioteca.upc.edu/10.1145/1217299.1217303

[10] L. Fu and E. Medico, "FLAME, a novel fuzzy clustering method for the analysis of DNA microarray data," *BMC Bioinformatics*, vol. 8, no. 1, p. 3, 2007. [Online]. Available: https://doi.org/10.1186/1471-2105-8-3

[11] H. Chang and D.-Y. Yeung, "Robust path-based spectral clustering," *Pattern Recognition*, vol. 41, no. 1, pp. 191–203, 2008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320307002038

[12] T. M. Inc., "Matlab version: 9.13.0 (r2022b)," Natick, Massachusetts, United States, 2022. [Online]. Available: https://www.mathworks.com

[13] W. S. Torgerson, "Multidimensional scaling: I. theory and method," *Psychometrika*, vol. 17, pp. 401–419, 1952. [Online]. Available: https://api.semanticscholar.org/CorpusID:120849755

[14] P. Fränti, "Clustering benchmark datasets," http://cs.joensuu.fi/sipu/datasets/, accessed: March 17, 2025.

[15] P. Fränti and S. Sieranoja, "K-means properties on six clustering benchmark datasets," pp. 4743–4759, 2018. [Online]. Available: http://cs.uef.fi/sipu/datasets/

[16] S. Lloyd, "Least squares quantization in pcm," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 129–137, 1982.

[17] R. R. Sokal and C. D. Michener, "A statistical method for evaluating systematic relationships," *University of Kansas Science Bulletin*, vol. 38, pp. 1409–1438, 1958.

[18] J. H. J. Ward, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, Mar. 1963.

[19] P. Fränti and O. Virmajoki, "Iterative shrinking method for clustering problems," *Pattern Recognition*, vol. 39, no. 5, pp. 761–775, 2006. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0031320305003778

[20] M. P. Sampat, Z. Wang, S. Gupta, A. C. Bovik, and M. K. Markey, "Complex wavelet structural similarity: A new image similarity index," *IEEE Transactions on Image Processing*, vol. 18, no. 11, pp. 2385–2401, 2009.

[21] Mehul, "Complex-wavelet structural similarity index (cw-ssim)," https://www.mathworks.com/matlabcentral/fileexchange/43017-complex-wavelet-structural-similarity-index-cw-ssim, 2013, [Online; accessed March 18, 2025].

[22] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning Research*, vol. 9, pp. 2579–2605, 2008. [Online]. Available: http://www.jmlr.org/papers/v9/vandermaaten08a.html

[23] K. Pearson, "On lines and planes of closest fit to systems of points in space," *Philosophical Magazine*, vol. 2, no. 11, pp. 559–572, 1901.

APPENDIX

TABLE X: Optimal DPC hyperparameters identified via Bayesian optimization for each dataset in Experiment 4.

Dataset	Percent (%)	Density Estimator
Iris	4.0582	Cutoff
Wine	2.5062	Cutoff
Digits	0.5000	Gaussian
Moons	3.3391	Cutoff
Blobs	4.0844	Cutoff

TABLE XI: Optimal K-means hyperparameters identified via Bayesian optimization for each dataset in Experiment 4.

Dataset	Init. Method	Number of Initializations (n_{init})
Iris	K-means++	12
Wine	Random	9
Digits	Random	5
Moons	K-means++	5
Blobs	K-means++	12

TABLE XII: Optimal DBSCAN hyperparameters identified via Bayesian optimization for each dataset in Experiment 4.

Dataset	Epsilon (ε)	Min. Samples
Iris	1.0872	3
Wine	1.9560	2
Digits	1.9539	7
Moons	0.2013	2
Blobs	1.9539	7

TABLE XIII: Optimal Spectral Clustering hyperparameters identified via Bayesian optimization for each dataset in Experiment 4.

Dataset	Gamma (γ)	Assign Labels	Affinity
Iris	0.6174	K-means	Nearest Neighbors
Wine	0.0507	Discretize	RBF
Digits	0.6174	K-means	Nearest Neighbors
Moons	0.6174	K-means	Nearest Neighbors
Blobs	2.4526	K-means	RBF

TABLE XIV: Optimal GMM hyperparameters identified via Bayesian optimization for each dataset in Experiment 4.

Dataset	Covariance Type	Init. Parameters	Max Iterations
Iris	Full	K-means	135
Wine	Tied	Random	190
Digits	Spherical	Random	96
Moons	Diagonal	K-means	200
Blobs	Spherical	K-means	140