Kernel Density Based Spatial Clustering of Applications with Noise

Rohan N. Kalpavruksha[1], Roshan N. Kalpavruksha,[1], Teryn Cha,[2] Sung-Hyuk Cha,[1]
[1]Computer Science Department, Pace University, New York, NY, USA
[2]Computer Sciences, Essex County College, Newark, NJ, USA
rk25464n@pace.edu, rk68465n@pace.edu, yan@essex.edu, scha@pace.edu

Abstract

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a widely used clustering algorithm renowned for its ability to identify clusters of arbitrary shapes and detect noise. However, its reliance on fixed parameters, such as the minimum number of points (*MinPts*) and the epsilon radius (ϵ), makes it sensitive to variations in sample density. This paper reinterprets DBSCAN as a specific case of kernel density estimation (KDE)-based clustering, where the kernel shape corresponds to a hyper-rectangular pillar or cylindrical kernel, depending on the distance metric. Building on this foundation, we introduce a flexible framework incorporating various kernel functions, including uniform, conical, Epanechnikov, cosine, exponential, and Gaussian kernels, to estimate the density distribution of data points. The threshold values are selected to identify high-density regions by retaining the top 90% of points, while excluding low-density points as noise, thereby enhancing clustering precision. Clusters are adaptively formed by leveraging points within the kernel range, thereby increasing the algorithm's robustness to noise and its adaptability to irregular density patterns. Empirical results demonstrate that the proposed approach outperforms traditional DBSCAN, as evidenced by lower Davies-Bouldin indices and higher silhouette scores. This study highlights the potential of density-driven clustering for practical applications, including social media sentiment analysis, customer segmentation in e-commerce, and medical data analysis, particularly in scenarios involving noise-prone or unevenly distributed datasets.

I. Introduction

Density-Based Spatial Clustering of Applications with Noise (DBSCAN) is a widely used clustering algorithm renowned for its ability to identify clusters of arbitrary shapes and effectively distinguish noise from meaningful data points (Ester et al. 1996). DBSCAN groups densely packed points based on two key parameters: epsilon (ϵ, the neighborhood radius) and min_samples (the minimum number of points required to form a cluster). While DBSCAN has achieved significant success across various applications, it exhibits limitations when handling datasets with heterogeneous densities or complex structures (Sander et al. 1998).

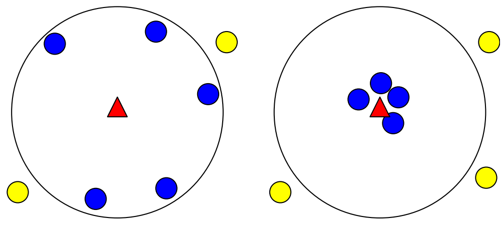


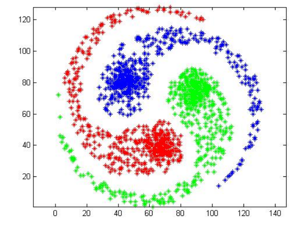
Figure 1: Motivation: DBSCAN vs. KDBSCAN.

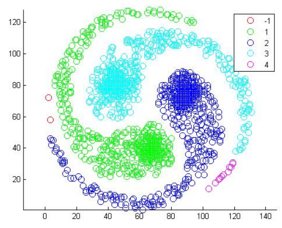
Generalized DBSCAN (GDBSCAN) extends the original DBSCAN framework by introducing flexibility in defining core points, allowing the use of additional criteria beyond simple point counts within a radius (Sander et al. 1998). While GDBSCAN increases adaptability, it still relies on fixed density thresholds, which may be inadequate for datasets with highly variable density distributions.

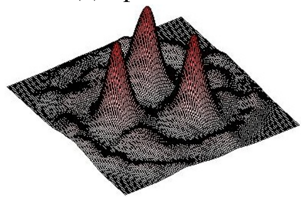
To address these challenges, we propose Kernel Density-Based Spatial Clustering of Applications with Noise (KDBSCAN), a novel clustering approach that integrates Kernel Density Estimation (KDE) into the DBSCAN framework. KDE, introduced by Parzen in (Parzen 1962), is a non-parametric method for estimating the probability density function (PDF), traditionally used in applications such as classification (Duda, Hart, and Stork 2001) and regression (Altman 1992). In this work, KDE is applied to clustering. Unlike DBSCAN and its derivatives, KDBSCAN eliminates the reliance on fixed density thresholds. Instead, it leverages the flexibility of kernel functions to dynamically adapt to the underlying data distribution, facilitating effective clustering across datasets with varying densities.

KDBSCAN offers several advantages over traditional DBSCAN. First, high-density regions are identified using KDE-based density estimation. Next, low-density outliers are excluded as noise, enhancing robustness and precision. Furthermore, clusters are formed adaptively based on kernel density values within a specified range, ensuring superior performance on datasets with irregular density patterns.

Figure 1 highlights the motivation behind KDBSCAN. In Case 1, DBSCAN identifies the point as a core point due to the number of neighbors within the radius ϵ, whereas in Case 2, it does not. By contrast, using a Gaussian kernel for KDE


(a) Spatial data


(b) DBSCAN plot


(c) 3D visualization

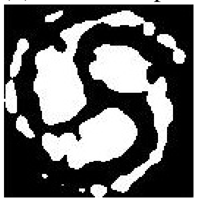
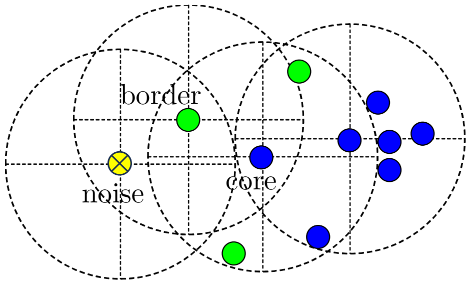
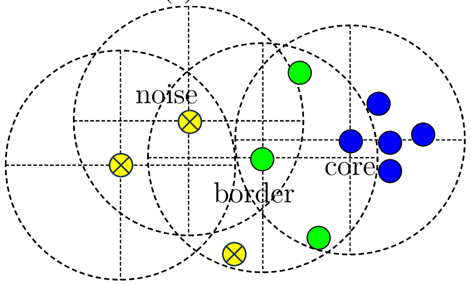

(d) Contour plot

Figure 2: Density types of point.



(a) DBSCAN



(b) KDBSCAN

Figure 3: Density types of point.

reverses the classifications: in Case 2, the density around the point is sufficiently high to be considered a core point, while in Case 1, it is not. This illustrates the adaptability and precision of KDE-based clustering in recognizing meaningful patterns.

One of the significant advantages of kernel DBSCAN over standard DBSCAN lies in its enhanced visualization capabilities. While DBSCAN facilitates the plotting of data points, as illustrated in Figure 2(b), kernel DBSCAN enables the visualization of contours in a three-dimensional plot, as shown in Figure 2(c). This approach allows for the representation of various contours corresponding to different threshold values, providing a more nuanced understanding of the data distribution. An example of a contour image is presented in Figure 2(d).

Similar work was exploited in (Pla-Sacristán et al. 2019) which focuses on adaptive bandwidth selection and combines kernel-based density with VDBSCAN, a variant that uses density variance to determine ϵ-neighborhoods. In contrast, our method introduces a fixed kernel-based density function directly into the core point definition, allowing for a smoother, tunable density threshold via kernel functions, with a focus on generalizability across datasets and flexibility in kernel choice. Additionally, our formulation decouples the ϵ parameter from distance-based thresholds, instead relying on kernel integrals for neighborhood influence, leading to more robust cluster detection in varying densities and overlapping regions.

The rest of the paper is organized as follows: First, Section II provides the preliminary definitions and concepts in DBSCAN. Next, kernel density estimation and its integration into clustering are introduced in Section III. Section IV evaluates various kernel DBSCAN methods on the NYPD Shooting Incident dataset, with a comparison against DBSCAN. Finally, Section V concludes this work.

II. Preliminary: DBSCAN

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) relies on five key concepts: *eps* (ϵ), *MinPts*, core points, border points, and noise points. These concepts are foundational for understanding the algorithm.

The parameter *eps* (ϵ) defines the maximum radius of the neighborhood around a point. This user-defined value is critical, as it determines the proximity required for points to be considered neighbors. The density of a point is calculated based on the number of points within its ϵ-radius:

$$\mathrm{dens}(x, \epsilon) = \sum_{y \in X} \mathfrak{T}(d(x, y) \leq \epsilon), \tag{1}$$

$$\mathfrak{T}(p) = \begin{cases} 1, & \text{if } p = \text{true}, \\ 0, & \text{otherwise.} \end{cases} \tag{2}$$

The default distance metric $d(x, y)$ is the Euclidean distance (L_2 norm).

The second parameter, *MinPts* (τ), specifies the minimum number of points required within the ϵ-radius for a point to be classified as a core point. Based on these parameters, every point in the dataset is categorized as either a core, border, or noise point:

- A *core point* has at least *MinPts* neighbors within its ϵ-radius.
- A *border point* is not a core point but lies within the ϵ-radius of a core point.
- A *noise point* is neither a core point nor a border point.

This classification can be expressed as:

$$\text{dens\_type}(x) = \begin{cases} \text{core} & \text{if dens}(x) \geq \tau \\ \text{border} & \text{if dens}(x) < \tau \\ & \wedge \exists y \in X(d(x,y) \leq \epsilon \\ & \wedge \text{dens\_type}(x) = \text{core}) \\ \text{noise} & \text{if dens}(x) < \tau \\ & \wedge \forall y \in X(d(x,y) \leq \epsilon \\ & \wedge \text{dens\_type}(x) \neq \text{core}) \end{cases} \quad (3)$$

For example, in Figure 3(a), assume *MinPts* $= 5$. The central point is a core point because it has at least five neighbors within the ϵ-radius. The first and second points from the left are not core points, as they have fewer than five neighbors. Among these, the leftmost point is a noise point because none of its neighbors is a core point, while the second point is a border point because it is within the ϵ-radius of a core point.

All data points are ultimately categorized as core, border, or noise. Let D_c and D_b represent the sets of all core points and border points, respectively. Clusters are formed as follows: First, each core point is assigned to a cluster. Next, border points are assigned to the nearest core point's cluster. Noise points are ignored and typically assigned to cluster ID 0. The pseudo-code for DBSCAN is shown in Algorithm 1.

Algorithm 1 *DBSCAN*

```
DBSCAN($D_c, D_b, \epsilon$) inputs are global.
  for $x \in D_c$ .................................... 1
    cluster($x$) = 0        # initialize unvisited ........ 2
  clusterID = 0 .................................. 3
  for $x \in D_c$ .................................... 4
    if cluster($x$) = 0 ............................ 5
      clusterID = clusterID + 1 .................... 6
      cluster($x$) = clusterID ...................... 7
      expandcluster($x$) .......................... 8
  for $x \in D_b$ .................................... 9
    cluster($x$) = cluster($\arg\min_{y \in D_c} d(x,y)$) .......... 10
  return cluster assignments for all points .......... 11

expandcluster($x$)    for $y \in D_c$ ..................... 1
    if cluster($y$) $\neq 0 \wedge d(x,y) \leq \epsilon$ ................ 2
      cluster($y$) = cluster($x$) ...................... 3
      expandcluster($y$) .......................... 4
  return
```

By default, DBSCAN assigns a border point to the cluster of the first core point it encounters during processing, as described in (Ester et al. 1996). However, in Algorithm 1, border points are instead assigned to the cluster of the nearest core point for improved consistency (line 10). This modification aligns with the algorithm's role as a generic framework, enabling the integration of various kernels as weights, as detailed in the subsequent section.

III. Kernel DBSCAN

This section introduces Kernel Density-Based Spatial Clustering of Applications with Noise (Kernel DBSCAN), an extension of the traditional DBSCAN that incorporates Kernel Density Estimation (KDE) to define point density. Unlike DBSCAN, which relies on the simple count of neighbors within a fixed radius, Kernel DBSCAN uses kernel functions to estimate the density of data points in the neighborhood.

Kernel Density Estimation (KDE), first introduced by Parzen (Parzen 1962), is a non-parametric method for estimating the probability density function (PDF) of a dataset. Numerous kernel functions have been explored in the literature; for a comprehensive list, see (Silverman 1986; Botev, Grotowski, and Kroese 2010).

Kernel DBSCAN follows the same clustering procedure as traditional DBSCAN, with a modification in the density definition. Specifically, the density of a point, $\text{dens}(x, \epsilon)$, is now determined using a kernel function K, as shown below:

$$\text{dens}(x, \epsilon) = \sum_{y \in X} K\left(\frac{d(x,y)}{\epsilon}\right) \quad (4)$$

where $d(x, y)$ represents the distance between points x and y, and ϵ is the radius parameter.

In this study, we explore six commonly used kernel functions in subsequent experiments: the uniform, triangular, Epanechnikov, cosine, exponential, and Gaussian kernels. Their shapes in one and two dimensions are illustrated in Figure 4. Let u denote the normalized distance between two points, defined as

$$u = \frac{d(x,y)}{\epsilon} \quad (5)$$

The six kernels are defined as follows:

$$K_u(u) = \begin{cases} \frac{1}{2}, & \text{if } |u| \leq 1 \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

$$K_t(u) = \begin{cases} 1 - |u|, & \text{if } |u| \leq 1 \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

$$K_e(u) = \begin{cases} \frac{3}{4}(1 - u^2) & \text{if } |u| \leq 1 \\ 0, & \text{otherwise} \end{cases} \quad (8)$$

$$K_c(u) = \begin{cases} \frac{\pi}{4}\cos\left(\frac{\pi u}{2}\right), & \text{if } |u| \leq 1 \\ 0, & \text{otherwise} \end{cases} \quad (9)$$

$$K_x(u) = = \frac{1}{2} e^{-|u|} \quad (10)$$

$$K_g(u) = = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2} \quad (11)$$

The uniform kernel, also known as the rectangular kernel, as defined in equation (6), produces results identical to those of traditional DBSCAN. When using the L_2 Euclidean distance, this kernel forms a cylindrical shape. If the L_1 Manhattan distance is used, the kernel forms a rectangular pillar shape, as depicted in Figure 4. This kernel focuses purely on the count of points within a given radius. The uniform or triangular kernel, defined in equation (7), assigns greater weight to closer neighbors. When the L_2 Euclidean distance is used, this kernel forms a conical shape, whereas with the L_1 Manhattan distance, it forms a pyramidal shape, as shown in Figure 4. The Epanechnikov kernel (Epanechnikov 1969), defined in equation (8), has a shape similar

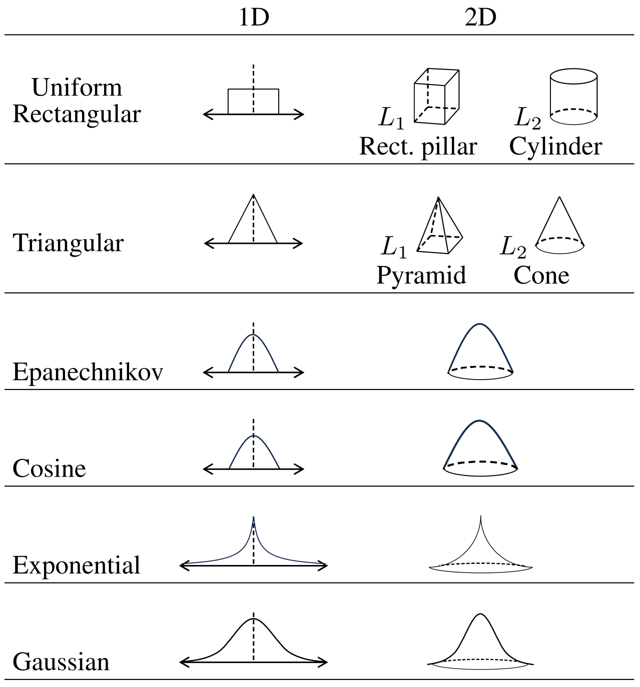


Figure 4: Pervasive Kernels.

to the cosine kernel, both of which are nonzero only for $|u| < 1$. The exponential (10) and Gaussian (11) kernels consider all points within their respective neighborhoods, making them particularly useful for estimating density over larger distances.

For instance, in Figure 3 (b), when the kernels defined in equations (7) $\sim$ (11) are applied, the central point is no longer classified as a core point, but rather as a border point, as all of its neighbors are far from the point. The second point from the left, which would typically be classified as a border point in traditional DBSCAN, is instead identified as a noise point because none of its neighbors is classified as a core point. This distinction highlights a key difference between conventional DBSCAN, which uses a uniform kernel, and kernel-based DBSCAN methods.

IV. Real world Spatial Data

In this section, we apply various kernel-based DBSCAN methods to a real-world spatial dataset from (City of New York 2021) to illustrate their superiority over standard DBSCAN. The dataset consists of geographic data potentially relevant to crime analysis, with a particular emphasis on clustering techniques. Initial exploratory data analysis is performed to visualize the spatial distribution of the data, which sets the stage for the subsequent application of DBSCAN and kernel density-based clustering methods.

The dataset includes latitude and longitude coordinates of shooting incidents in New York City between 2006 and 2021. The outcomes of different kernel DBSCAN methods are visualized by plotting these coordinates, as shown in Figure 5.

While traditional DBSCAN efficiently groups data points

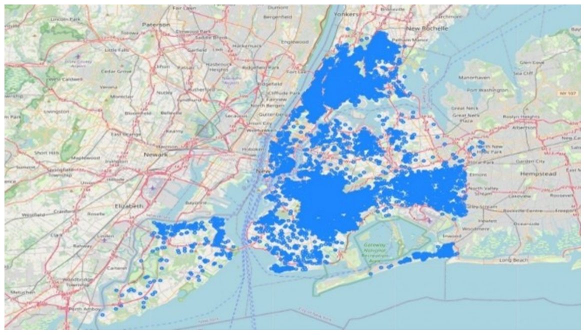


Figure 5: Spatial Representation of NYPD Shooting Incident Data (2006–2021) (City of New York 2021).

based on density thresholds, it may struggle with datasets that exhibit uneven distributions or varying densities. Kernel density-based clustering extends DBSCAN by leveraging customizable kernels to adaptively estimate point density. This extension improves cluster separation and delineation, particularly in complex or noisy datasets, resulting in more accurate and nuanced clustering outcomes.

Kernel Density Estimation (KDE) is applied to estimate the probability density function (PDF) of the data. Six kernel functions, as defined in Equations (6)$\sim$(11), were utilized to perform the estimation. The bandwidth parameter (ϵ) was varied at values of 0.005, 0.01, and 0.02 to smooth the density estimates. Figure 6 provides a detailed visual representation of the data distribution, revealing regions of high and low density.

Evaluation of Clustering Methods

To quantitatively assess the performance of the clustering methods, several metrics were considered. Among these, the Davies-Bouldin index (DB) (Davies and Bouldin 1979) and the Silhouette method (s) (Rousseeuw 1987) were selected due to their widespread use in evaluating DBSCAN clustering results.

Let a clustering method generate k clusters: $C_1, \ldots, C_k$. The pairwise cluster goodness coefficient between clusters C_x and C_y, denoted $R_{x,y}$, is defined as:

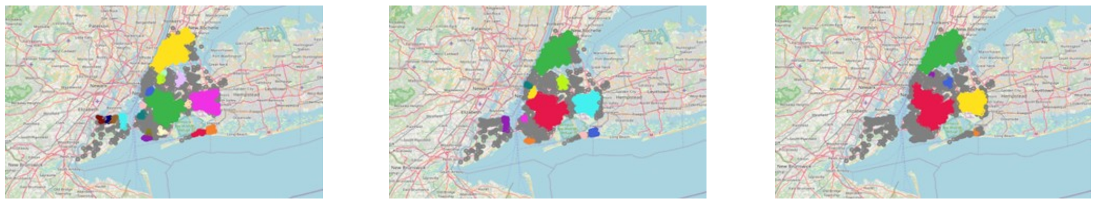
$$R(C_x, C_y) = \frac{\text{separation}(C_x, C_y)}{\text{cohesion}(C_x) + \text{cohesion}(C_y)}, \qquad (12)$$

where separation(C_x, C_y) refers to the distance between clusters C_x and C_y, and cohesion(C_x) and cohesion(C_y) are the internal cohesion measures of the clusters. $R(C_x, C_y)$ is the ratio of the separation between two clusters to the sum of their cohesions. A higher $R_{x,y}$ value indicates better clustering, reflecting greater separation and lower cohesion between clusters.

The Davies-Bouldin index (Davies and Bouldin 1979) is given by:

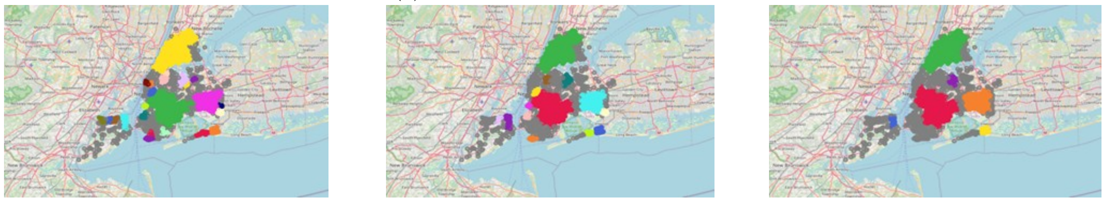
$$\text{DB}(C_1, \cdots, C_k) = \frac{\sum_{i=1}^{k} \max_{1 \le i \ne j \le k} R^{-1}(C_i, C_j)}{k} \qquad (13)$$

$\epsilon = 0.005$ $\epsilon = 0.01$ $\epsilon = 0.02$



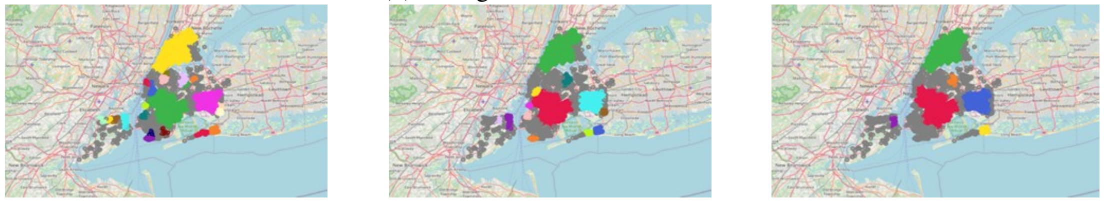
$s = 0.265$ DB = 0.505 $s = 0.448$ DB = 0.433 $s = 0.397$ DB = 0.467

(a) Uniform kernel DBSCAN



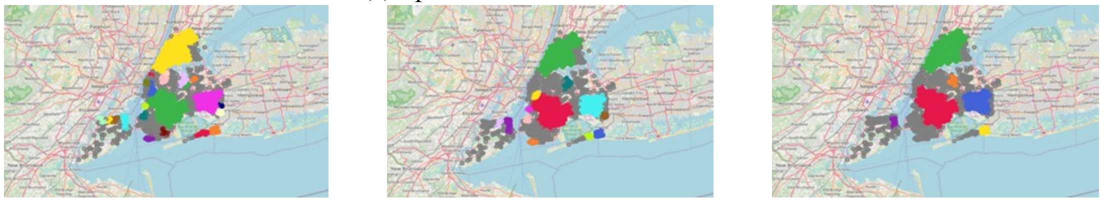
$s = 0.225$ DB = 0.568 $s = 0.376$ DB = 0.463 $s = 0.531$ DB = 0.411

(b) Triangular kernel DBSCAN



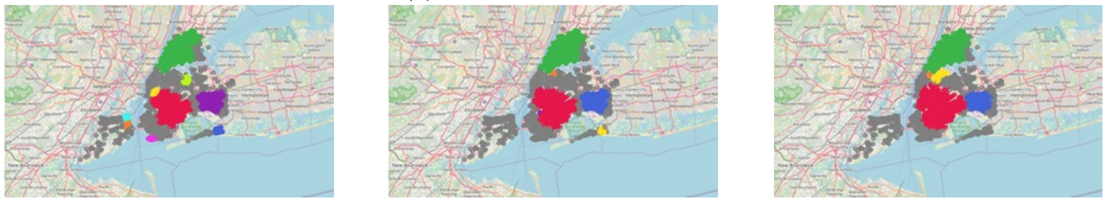
$s = 0.261$ DB = 0.571 $s = 0.422$ DB = 0.460 $s = 0.530$ DB = 0.411

(c) Epanechnikov kernel DBSCAN



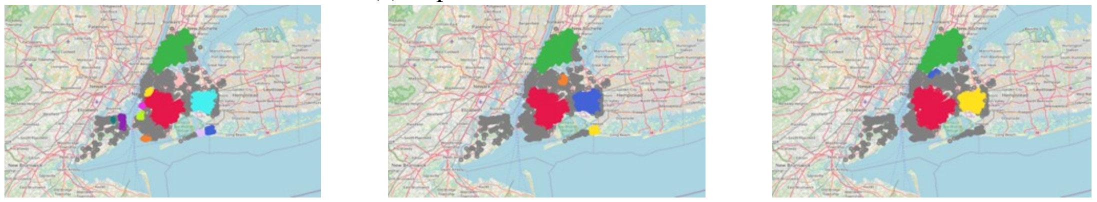
$s = 0.261$ DB = 0.570 $s = 0.421$ DB = 0.461 $s = 0.531$ DB = 0.412

(d) Cosine kernel DBSCAN



$s = 0.458$ DB = 0.456 $s = 0.261$ DB = 0.531 $s = 0.426$ DB = 0.538

(e) Exponential kernel DBSCAN



$s = 0.448$ DB = 0.435 $s = 0.526$ DB = 0.437 $s = 0.436$ DB = 0.529

(f) Gaussian kernel DBSCAN

Figure 6: Various Kernel DBSCAN results on NYPD Shooting Incident Data.

where a lower DB index signifies better cluster quality.

The Silhouette method (Rousseeuw 1987) calculates the silhouette value $s(x)$ for a given data point x as:

$$s(x) = \frac{b(x) - a(x)}{\max a(x), b(x)} \tag{14}$$

where $a(x)$ represents the average intra-cluster distance, and $b(x)$ is the minimum average distance to points in other clusters. The silhouette value $s(x)$ ranges from -1 to 1, with values close to 1 indicating well-clustered points, values near 0 suggesting overlapping clusters, and negative values indicating potential misclassification. The mean silhouette value across all points provides an interpretable measure of overall cluster validity.

Based on these evaluation metrics, the triangular (conical) kernel DBSCAN with a bandwidth of $\epsilon = 0.02$ achieved the best performance, exhibiting the lowest Davies-Bouldin index and the highest silhouette score. In comparison to standard DBSCAN, which uses a uniform kernel, kernel-based DBSCAN consistently outperformed across different bandwidth values, underscoring the flexibility and adaptability of kernel density-based approaches.

V. Conclusion

This study demonstrates the efficacy of kernel density-based clustering as a robust alternative to traditional density-based methods such as DBSCAN. By integrating Kernel Density Estimation (KDE) with various kernel functions, the proposed approach mitigates the limitations associated with fixed parameter dependencies and uniform weighting in conventional algorithms. The flexibility of KDE facilitates adaptive clustering, enabling the effective handling of datasets with varying densities and irregular cluster shapes.

The experimental results validate the superiority of kernel density-based clustering, showing improved silhouette scores and more precise cluster delineations. The use of customizable kernel functions - such as the uniform, triangular (conical), Epanechnikov, cosine, exponential, and Gaussian kernels - enhances the method's adaptability to diverse datasets. Furthermore, kernel-based approaches demonstrate significant resilience in distinguishing meaningful clusters from noise, a challenge that often impedes traditional DBSCAN methods.

The findings of this study lay the groundwork for broader applications of kernel-based clustering in fields requiring nuanced data segmentation. The proposed methodology shows promise for integration with real-time systems and large-scale datasets, representing a notable advancement in clustering and density estimation techniques. Future research will focus on optimizing computational efficiency, exploring advanced kernel functions, and extending the methodology to high-dimensional data spaces.

References

Altman, N. S. 1992. An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician* 46(3):175–185.

Botev, Z. I.; Grotowski, J. F.; and Kroese, D. P. 2010. Kernel density estimation via diffusion. *The Annals of Statistics* 38(5):2916–2957.

City of New York. 2021. NYPD Shooting Incident Data (Historic): 2006-2021. Accessed: 2025-01-24 `https://data.cityofnewyork.us/Public-Safety/NYPD-Shooting-Incident-Data-Historic-/833y-fsy8/about_data`.

Davies, D. L., and Bouldin, D. W. 1979. A cluster separation measure. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 1(2):224–227.

Duda, R. O.; Hart, P. E.; and Stork, D. G. 2001. *Pattern Classification*. Wiley.

Epanechnikov, V. A. 1969. Non-parametric estimation of a multivariate probability density. *Theory of Probability and Its Applications* 14(1):153–158.

Ester, M.; Kriegel, H.-P.; Sander, J.; and Xu, X. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 226–231. AAAI Press.

Parzen, E. 1962. On estimation of a probability density function and mode. *The Annals of Mathematical Statistics* 33(3):1065–1076.

Pla-Sacristán, E.; González-Díaz, I.; Martínez-Cortés, T.; and Díaz-de María, F. 2019. Finding landmarks within settled areas using hierarchical density-based clustering and meta-data from publicly available images. *Expert Systems with Applications* 123:315–327.

Rousseeuw, P. J. 1987. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics* 20:53–65.

Sander, J.; Ester, M.; Kriegel, H.-P.; and Xu, X. 1998. Density-based clustering in spatial databases: The algorithm gdbscan and its applications. *Data Mining and Knowledge Discovery* 2:169–194.

Silverman, B. W. 1986. *Density Estimation for Statistics and Data Analysis*. London: Chapman and Hall.