Электронный научный журнал "Математическое моделирование, компьютерный и натурный эксперимент в естественных науках" http://mathmod.esrae.ru/
URL статьи: mathmod.esrae.ru/19-71
Ссылка для цитирования этой статьи:
Stepanova A., Mironov S.V., Korobov E. The clusterization of geo-tagged data for finding tourists attractions // Математическое моделирование, компьютерный и натурный эксперимент в естественных науках. 2018. №3
The work was supported by the Russian Fund for Basic Research, project 18-37-00060.

УДК 519.17

КЛАСТЕРИЗАЦИЯ ГЕОТЕГИРОВАННЫХ ДАННЫХ ДЛЯ ПОИСКА ТУРИСТИЧЕСКИХ ДОСТОПРИМЕЧАТЕЛЬНОСТЕЙ

Степанова А.[1], Миронов С.В.[2], Коробов Е.[3]
Саратовский государственный университет имени Н.Г.Чернышевского, Россия, Саратов
[1]ste.nasty@gmail.com, [2]mironovsv@info.sgu.ru, [3]korobovea@yandex.ru

THE CLUSTERIZATION OF GEO-TAGGED DATA FOR FINDING TOURISTS ATTRACTIONS

Stepanova A.[1], Mironov S. V.[2], Korobov E.[3]
Saratov State University, Russia, Saratov,
[1]ste.nasty@gmail.com, [2]mironovsv@info.sgu.ru, [3]korobovea@yandex.ru

Аннотация. В данной работе на основе геоданных фотографий, сделанных в городе Саратове, необходимо было определить области наиболее популярных достопримечательностей. Данная задача решалась с помощью кластеризации графов. Граф строился на основе расположения координат места, где были сделаны фотографии. В работе использовались алгоритмы кластеризации DBSCAN и k-MXT. Решение было реализовано на языке Python в среде Jupyter.

Ключевые слова: кластеризация графа, взвешенный граф, геотегированные данные

Abstract. In this paper we examine the problem of detection of the most popular city attractions using a geo-tagged data of photographs. First, we constructed a graph on the basis of the photographed spot coordinates. Then we reduced the problem to the problem of graph clusterization. In our work we use two clustering algorithms, DBSCAN and k-MXT. The algorithms were implemented in Python using the Jupyter environment.

Keywords: graph clusterization, weighted graph, geo-tagged data

Introduction. Following ideas of paper [1] we perform the process of clustering in a graph formed from geo-tagged photographic data. We use some clustering algorithms such as k-MXT algorithm [1] and the DBSCAN algorithm [2] to solve the problem of clustering geographical data using one of Russian cities as an example. We construct a graph based on a dataset of geographical coordinates extracted from geo-tagged photographs taken from Flickr. To construct the graph we connect every vertex to other vertices if and only if their distance is less than a given threshold level. We show that the constructed graph appears to have the following structural properties: the subgraphs at some location are dense but the connection between subgraphs is sparse. Our experimental results follow the results of the paper [1] and show that the k-MXT algorithm is able to produce clusters which are comparable to the DBSCAN algorithm. An extended DBSCAN algorithm was analysed in [3] to detect stops in individual trajectories using GPS data. The paper [4] proposes a novel niche genetic algorithm (NGA) with density and noise for K-means clustering and apply the algorithm on taxi GPS data sets.

Many clustering algorithms (e.g., density-based, partition-based) have been proposed and examined [5]. One of them is K-means (partition-based), which is a more well-known and used in practice due to its simplicity and effectiveness with respect to other clustering algorithms (such as density-based or model-based). It is known that K-means algorithm has need in providing as an input the number of clusters K [6, 7].

- Data with coordinate marks were obtained from the Flickr site [8] using the flickrapi library. In total, 8449 photos with geodata and user names were received for Saratov. Clustering algorithms were running on two types of data:
- All received data were used (8449 photographs). In this case, $duplicateData = True$.
- No more than one photo from one user with the same geodata was used (about 1000 photos). In this case, $duplicateData = False$.

Since more objective results are obtained when $duplicateData = False$ in the following the cluster results will be given based on the value of parameter $duplicateData = False$. In this problem the distance between two points was considered as the shortest distance between two points on the surface of the sphere given by

$$\Delta\sigma = 2\arcsin\sqrt{\sin^2\left(\frac{\Delta\phi}{2}\right) + \cos\phi_1\cos\phi_2 + \sin^2\left(\frac{\Delta\lambda}{2}\right)}, \qquad (1)$$

where ϕ_1, ϕ_2 are the latitudes of the first and second points, λ_1, λ_2 – the longitudes of the first and second points accordingly, $\Delta\phi = |\phi_1 - \phi_2|$, $\Delta\lambda = |\lambda_1 - \lambda_2|$.

1. Clustering Algorithms

1.1. DBSCAN Algorithm

Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm proposed in the paper [2]. It is one of the most well-known clustering algorithms and most popular in research literature. DBSCAN algorithm is based on the following idea. For a given set of points in some space, it groups together the points that have many nearby neighbors. The points whose nearest neighbors are too far away, are marked as outliers points by the DBSCAN algorithm.

Let $G=(V,E)$ be a graph, where V is the set of vertices of the graph and E is the set of edges of the graph. The vertices of graph G are the points on the map where the photographs were taken. Initially, the graph G does not contain any edges. We need some definitions and notations. Let

- $\rho(p,q)$ be the distance between the vertices p and q;
- $E(p)$ be ε-neighborhood of the vertex (object) p, defined by the formula
- $E(p)=\{q \mid \rho(p,q)\le\varepsilon; p,q\in V\}$,
  where ε is a certain constant;
- a kernel or root object of degree $minPts$ is an object, whose ε-neighborhood contains no less than $minPts$ elements, i. e. $|E(p)|\ge minPts$, where $minPts$ is a constant. Objects that are not root are noise;
- if $q\in E(p)$ and p is a root object, then the object q is directly densely attainable from the object p;
- if there exist $p_1, p_2, \ldots p_n$, such that $p_1=p$, $p_n=q$ and p_{i+1} is directly densely attainable from p_i for all $i\in 1\ldots n-1$, then the object q is densely attainable from the object p.

If root vertex p of the graph G is not assigned to any cluster, then all vertices directly densely attainable from p to be added to the traversal list. For each root node q in the traversal list, we find all directly densely attainable vertices and add them to the same traversal list. The vertex p and all the vertices from the traversal list that do not belong to any cluster are assigned to a new cluster.

1.2. *k*-MXT Algorithm

k-MXT algorithm was proposed in [1]. The algorithm considers a graph fragmentation process in the following way: each vertex v selects the k adjacent vertices which have the largest number of common neighbours. For each selected neighbour u, we retain the edge (v,u) to form subgraph S of the input graph. The object of interest in [1] are the components of S, the k-Max-Triangle-Neighbour (k-MXT) subgraph, and the vertex clusters they produce in the original graph.

The vertices of graph G are the points on the map in which the pictures were taken. Initially, the graph G does not contain edges.

Each vertex of a graph is connected to all vertices located at a distance less than ε. No two vertices are connected more than once.

For the k-MXT algorithm, we need to construct a graph G' based on the graph G. Initially, the graph G' contains all vertices of the graph G, but does not contain any edges.

Let the set E contain all edges emanating from the vertex p. We add k edges from the set E to the graph G' according to the following rule: for each vertex p of the graph G, the weight of all edges originating from this vertex is calculated. The weight of the edge (q,r) is equal to the number of common neighbors of vertex q and vertex r. Let the set E contain all the edges emanating from the vertex p. Add k edges to the graph G' from the set E by the following rule.

- If the set E has less than k edges or exactly k edges, then all edges from the set E are added to the graph.
- If the set E has more than k edges, then k edges with the maximum weight are added to the graph. If it is needed to select from several edges with the same weight, then the edges are selected randomly.

Clusters will be the connected components of the resulting graph G'. In this paper, the connectivity components are distinguished by traversing the graph in depth-first.

2. Features of the Implementation of Algorithms

By neighbors of the vertex p we mean vertices located at a distance less than ε from the vertex p.

To speed up clustering algorithms, k-d-trees are often used. With the help of k-d-trees, the number of vertices which need to be sorted to find all neighbors of vertex p can be reduced.

In this paper the left and right binary searches were used to speed up the algorithms when clustering algorithms are implemented.

Let array *point* for each vertex of the graph contain the number of the vertex, its latitude and its longitude. Sort the array according to the non-decreasing longitude. While calculating the distance between two points using (1) we take into account only longitude, i. e. (1) takes the form

$$\Delta\sigma = 2\arcsin\sqrt{\sin^2\frac{\Delta\lambda}{2}}, \tag{2}$$

where λ_1, λ_2 – are the longitudes of the first and second points, respectively, $\Delta\lambda = |\lambda_1 - \lambda_2|$.

In this case, the left binary search returns for vertex p the number of the first position *left* such that vertex u corresponds to this position in array *point* and the distance between u and p, calculated by the (2), is less than ε. The right binary search returns for node p the number of the last position *right* such that vertex u corresponds to this position in array *point* and the distance between u and p, calculated by (2), is less than ε. Neighbors for p are searched in array *point* at positions from *left* to *right* according to (1).

3. Structural Properties of the Graph

In DBSCAN graph G is constructed implicitly during the execution of the algorithm. Each vertex p of the graph G that is not noise is joined by edges with directly densely attainable neighbors of the vertex p. In Fig. 1, 2 the graphs obtained for different values of the parameters $minPts$ and ε are presented. Table 1 presents the structural properties (density and diameter) of the graph G for different values of the parameters $minPts$ and ε.

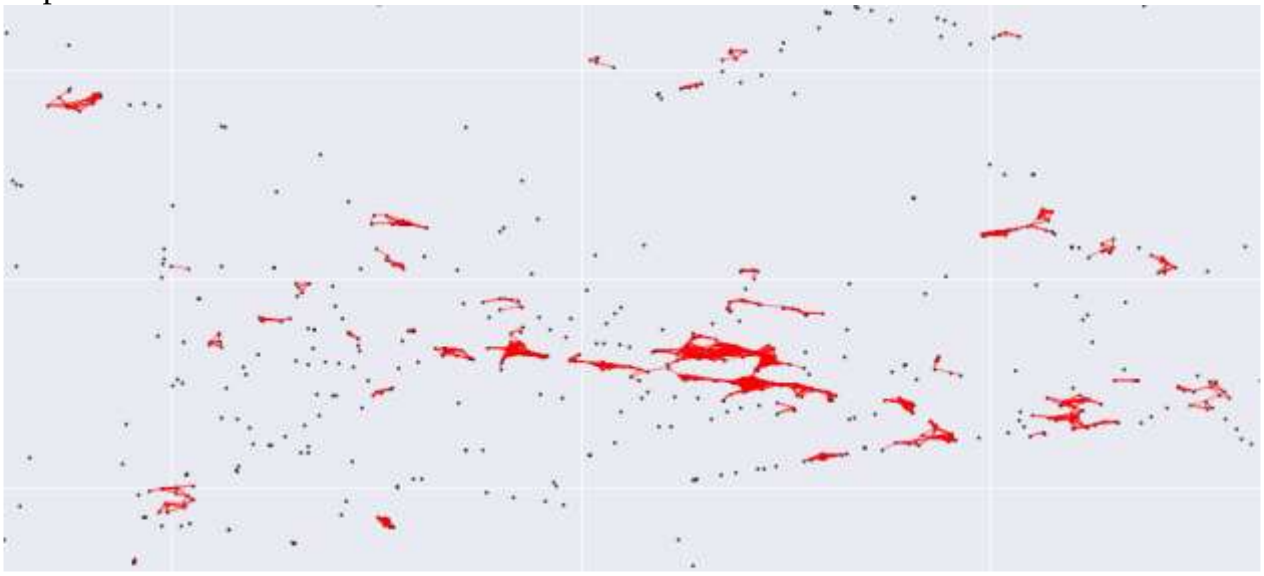


Fig. 1. Part of the graph G. DBSCAN algorithm. $minPts = 3$, $\varepsilon = 100$

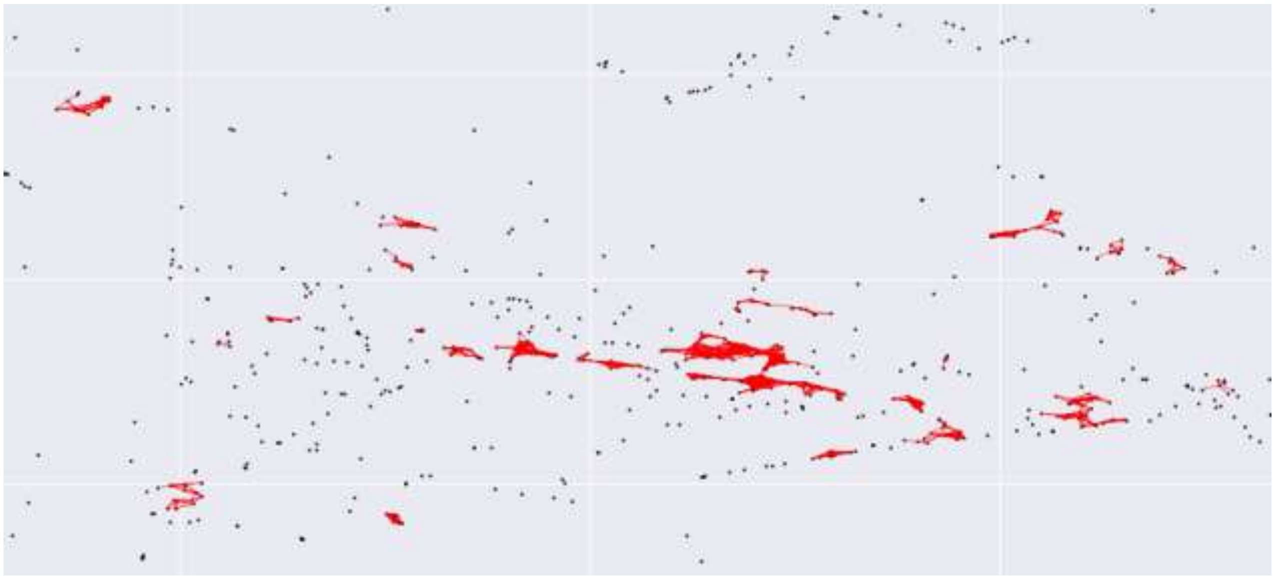


Fig. 2. Part of the graph G. DBSCAN algorithm. $minPts = 5$, $\varepsilon = 100$

Structural properties of the graph G.
DBSCAN algorithm Table 1

minPts	ε	*density*	*diameter*
3	100	0.0165	16
	100	0.0158	15
	75	0.0128	22
	75	0.0120	21

Table 2 shows the structural properties of the graph G constructed for the k-MXT algorithm with different values of parameter ε. Figures 3 and 4 show the graphs G obtained for different values of the parameter ε.

Structural properties of the graph G.
k-MXT algorithm Table 2

ε	*density*	*diameter*
100	0.0094	16
	0.0074	22
	0.0054	12

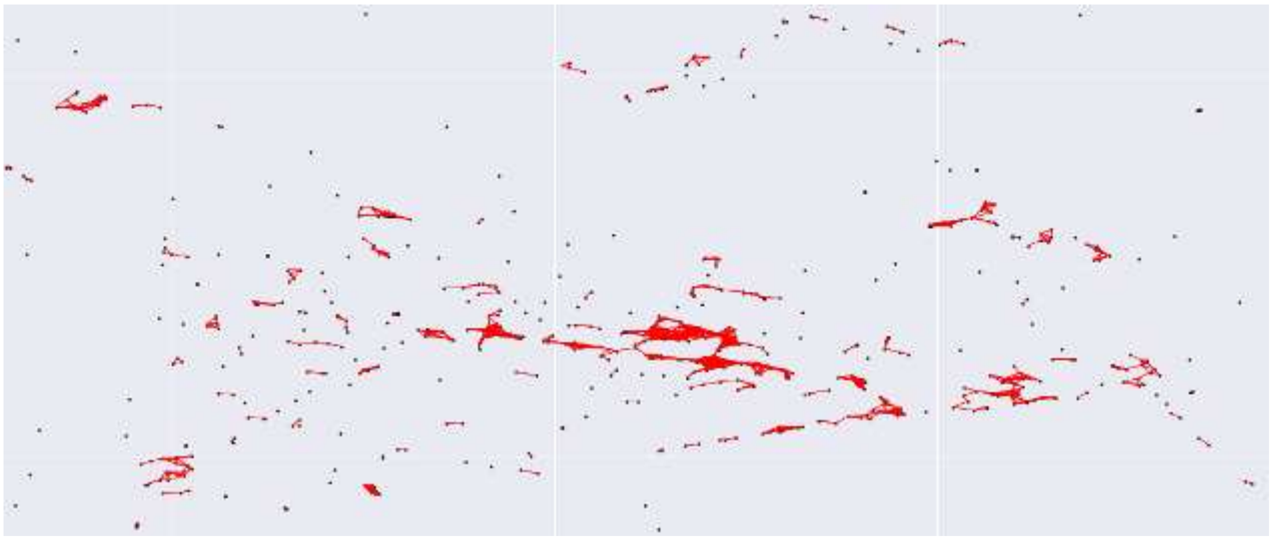


Fig. 3. Part of the graph G. k-MXT algorithm. $\varepsilon = 100$

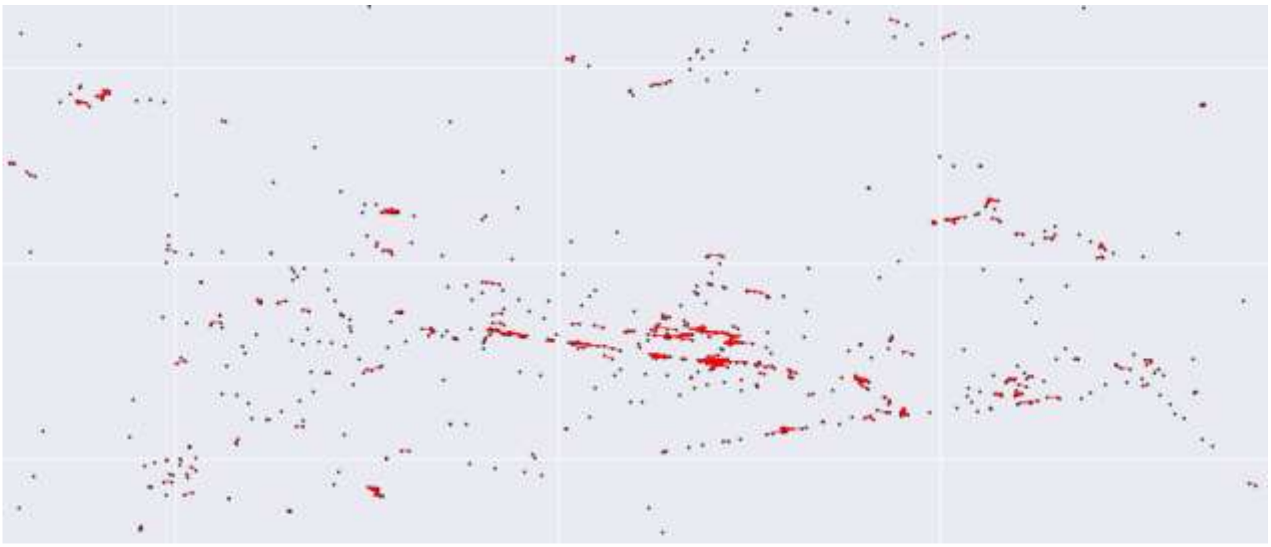


Fig. 4. Part of the graph G. k-MXT algorithm. $\varepsilon = 50$

4. Clustering Results

Further, in all figures, the vertices that are noises will be marked in red, the vertices located in the same cluster will be marked with the same color.

4.1. Results of Clustering Using the DBSCAN Algorithm

Consider partitioning the photographs into clusters using DBSCAN algorithm for different values of $minPts$, ε parameters. As the value of $minPts$ increases, while the value of ε does not change, the number of vertices that are noise rises and the number of clusters decreases. Many photographs, belonging to some clusters when $minPts = 3$, became noise when $minPts = 7$ (Fig. 5).

(*a*) (*b*)
Fig. 5. DBSCAN algorithm: (*a*) $minPts = 3$, $\varepsilon = 100$; (*b*) $minPts = 7$, $\varepsilon = 100$

Taking into account that the parameter $minPts$ with values much smaller than the average cluster size, does not make significant changes for large clusters, further $minPts$ is assumed to be equal to 3

Consider the resulting clusters in the central part of the city using $\varepsilon = 100$ and $\varepsilon = 75$. Looking through the maps (Fig. 6) it can be seen that the clusters given by algorithm using $\varepsilon = 75$ are smaller than the clusters obtained with $\varepsilon = 100$. Therefore, in most cases, the algorithm using $\varepsilon = 75$ divides the photos into more clusters than the algorithm with $\varepsilon = 100$, which is confirmed by the data in Table 3.

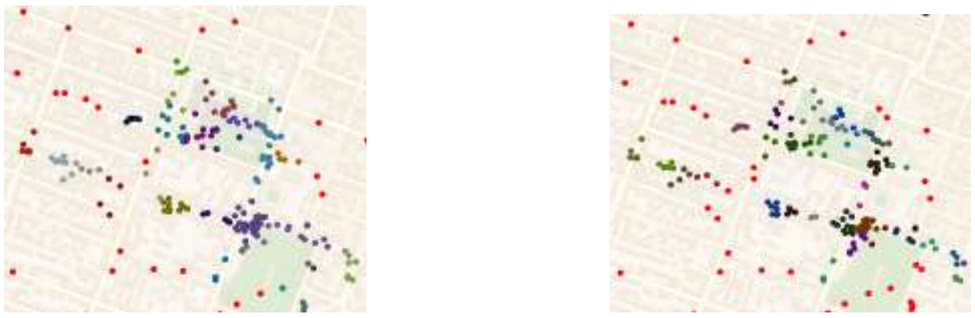

(*a*) (*b*)
Fig. 6. DBSCAN algorithm: (*a*) $minPts = 3$, $\varepsilon = 100$; (*b*) $minPts = 3$, $\varepsilon = 75$

Table 3

DBSCAN

$minPts$	ε	Number of clusters	Number of noises	Max cluster size	Average size of cluster
3	100	114	350	33	5.8
	100	73	484	32	7.3
	100	51	573	32	8.7
	75	109	424	33	5.4
	75	64	446	33	6.9
	75	43	643	33	8.7

50	88	462	22	5.3

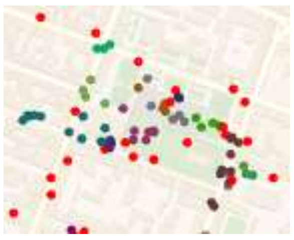
4.2. Results of Clustering Using the k-MXT Algorithm

Consider the results of the k-MXT algorithm clustering for various values of k, ε, w.

Table 4 shows that when the value of k increases, the number of clusters and the number of vertices that are noise are reduced. Figure 7 confirms the strong influence of k when splitting photos into clusters. With $k = 2$, more clusters are allocated in the city area than with $k = 4$.

k-MXT Table 4

k	ε	w	Number of clusters	Number of noises	Max cluster size	Average size of cluster
2	100	0	169	300	24	4.2
	100	3	91	569	25	4.9
	75	0	174	319	31	4.0
	75	3	75	637	29	5.1
	100	0	151	237	25	5.2
	100		77	528	27	6.3
	75	0	162	271	33	4.6
	75	3	62	602	31	6.7
	100	0	138	218	67	5.8
	100	3	67	513	34	7.5
	75	0	147	246	33	5.2
	75	3	61	593	32	6.9
	75	0	140	236	51	5.6
	75	3	57	574	33	7.8
	50	0	161	308	24	4.4
	50	3	44	678	24	7.7


(*a*)

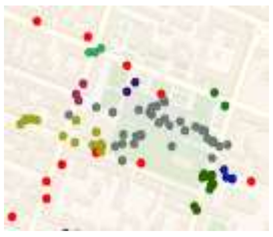
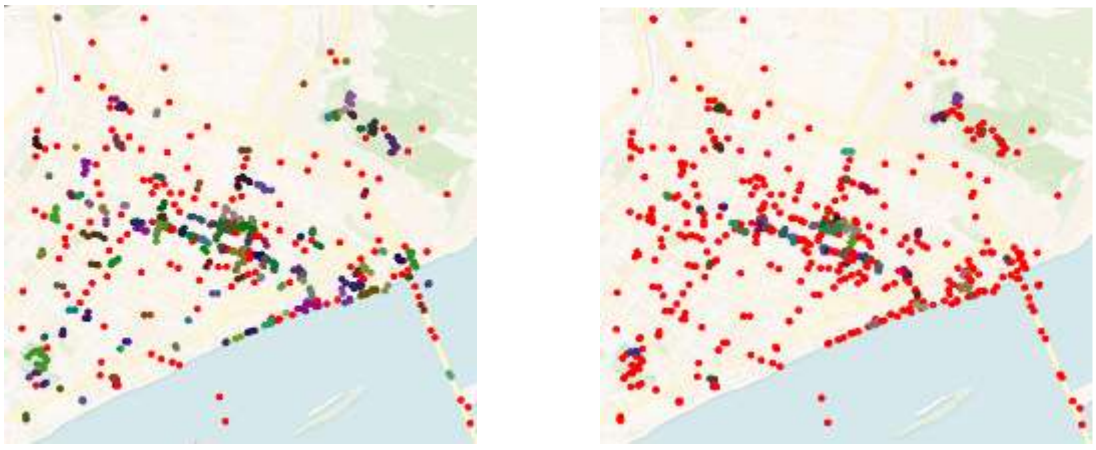

(*b*)

Fig. 7. k-MXT algorithm: (*a*) $k = 2$, $w = 3$, $\varepsilon = 100$; (*b*) $k = 4$, $w = 3$, $\varepsilon = 100$

Consider the influence of parameter w on the clustering results. As the w parameter increases, some of the vertices that belonged to any clusters earlier become noise. Therefore, the number of vertices that are noise increases, which is confirmed by the maps of the central part of the city (Fig. 8) and data of Table 4.


(*a*) (*b*)
Fig. 8. The central part of the city: (*a*) $k = 4$, $w = 0$, $\varepsilon = 75$; (*b*) $k = 4$ $w = 3$, $\varepsilon = 75$

Consider some attractions with a large area. The first example is the park zone. As the value w increases, the photos taken on this object become noise (Fig. 9).

(*a*) (*b*)
Fig. 9. Allocation of clusters in a park facility:
(*a*) $k = 4$, $w = 0$, $\varepsilon = 75$; (*b*) $k = 4$ $w = 3$, $\varepsilon = 75$

However, for the same values of k, ε, an increase in the parameter w has no effect on the clusterization of another large area object such as Teatralnaya Square (Fig. 10). This is due to the fact that graph G has a high density in the area of Teatralnaya Square and a small density in the area of City Park (Fig. 11). Therefore, the algorithm k-MXT allocates objects to clusters for large values of the parameter w, where the density of the graph G is high. It is not advisable to increase the value of w if the graph G has a small density.

(*a*) (*b*)
Fig. 10: Teatralnaya Square: (*a*) $k = 4$, $w = 0$, $\varepsilon = 75$; (*b*) $k = 4$ $w = 3$, $\varepsilon = 75$

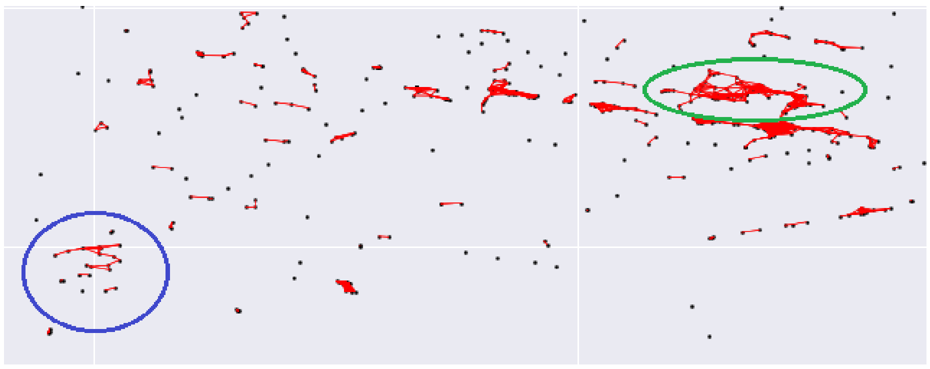


Fig. 11. Graph G at $\varepsilon = 75$. The area of the City Park is marked by blue color, the area of the Teatralnaya Square is marked by red color

As the parameter ε increases, the size of the clusters increases (Fig. 12). Photos taken on several blocks are grouped in one cluster at $\varepsilon = 100$.

(*a*) (*b*)

Fig. 12. 4-MXT algorithm: (*a*) $w = 0$, $\varepsilon = 75$; (*b*) $w = 0$, $\varepsilon = 100$

4.3. Comparison of Clustering Results

Table 5 shows the most popular places in Saratov, obtained with the help of DBSCAN and k-MXT algorithms with different parameters. The popularity of places was determined by the number of photos in clusters corresponding to the objects of the city. Since different algorithms differentiate objects in clusters differently, the lists of popular city places obtained by different algorithms differ. Figure 13 shows the variants of division into clusters of the central part of the city.

The most popular places in the city of Saratov, allocated by various algorithms Table 5

	DBSCAN $minPts = 3$, $\varepsilon = 100$	4-MXT $w = 0$, $\varepsilon = 75$	5-MXT $w = 0$, $\varepsilon = 50$
Conservatory	+	+	+
/ 331 Astrakhanskaya St	+	+	+
Teatralnaya Square		+	+
A-B Chernyshevsky St	+	+	+
Opera and Ballet Theatre		+	+
Flowered Gardens driveway	+	+	+
The Kryty Rynok stop	+	+	
“Cranes” monument	+		
Volgskaya St and Oktyabrskaya St intersection	+		+

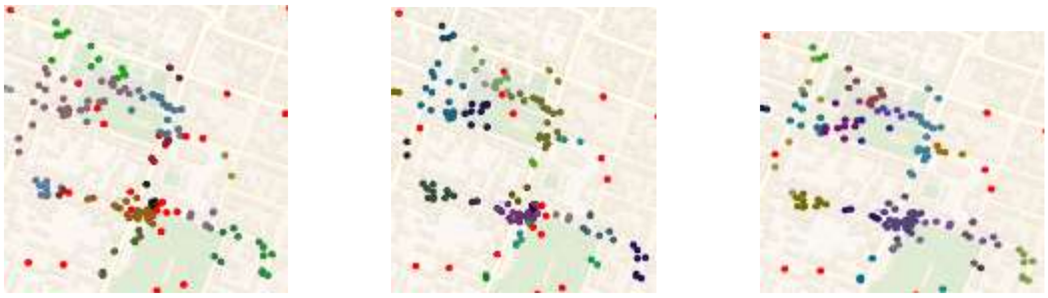

(*a*) (*b*) (*c*)

Fig. 13. Variants of division into clusters: (*a*) 4-MXT, $w = 0$, $\varepsilon = 75$; (*b*) 5-MXT, $w = 0$, $\varepsilon = 50$; (*c*) DBSCAN, $minPts = 3$, $\varepsilon = 100$

Conclusions. The dependence of clustering results on parameters of DBSCAN and k-MXT algorithms on the obtained data was studied. The DBSCAN algorithm works better on data where there are many points in a certain small radius. The k-MXT algorithm works better on chained data.

This work was supported by the Russian Fund for Basic Research, project 18-37-00060.

References

1. Cooper C., Vu N. An experimental study of the k-MXT algorithm with applications to clustering geo-tagged data // LCNS. 2018. V. 10836. P. 145–169. 15th Workshop on Algorithms and Models for the Web Graph, WAW 2018, Moscow, Russian Federation.
2. Ester M., Kriegel H.-P., Sander J., Xu X. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise // Proceedings of the Second International Conference on Knowledge Discovery and Data Mining KDD'96. Portland, Oregon: AAAI Press, 1996. P. 226–231.
3. Luo T., Zheng X., Xu G., Fu K., Ren W. An improved DBSCAN algorithm to detect stops in individual trajectories // ISPRS International Journal of Geo-Information. 2017. V. 6. №3. P. 63.
4. Zhou X., Gu J., Shen S., Ma H., Miao F., Zhang H., Gong H. An automatic k-means clustering algorithm of GPS data combining a novel niche genetic algorithm with noise and density // ISPRS International Journal of Geo-Information. 2017. V. 6. №2. P. 392.
5. Han J., Kamber M., Pei J. Data mining concepts and techniques. Waltham, Mass.: Morgan Kaufmann Publishers, 2012.
6. Jain A. K. Data clustering: 50 years beyond k-means // Pattern Recognition Letters. 2010. V. 31. №8. P. 651–666. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR).

7. Rahman M. A., Islam M. Z. A hybrid clustering technique combining a novel genetic algorithm with k-means // Knowledge-Based Systems. 2014. V. 71. P. 345–365.
8. Flickr. URL: https://www.flickr.com/.