k-Maxoids Clustering

Christian Bauckhage[1,2] and Rafet Sifa[2]

[1]B-IT, University of Bonn, Bonn, Germany
[2]Fraunhofer IAIS, Sankt Augustin, Germany
http://mmprec.iais.fraunhofer.de/bauckhage.html

Abstract. We explore the idea of clustering according to extremal rather than to central data points. To this end, we introduce the notion of the maxoid of a data set and present an algorithm for k-maxoids clustering which can be understood as a variant of classical k-means clustering. Exemplary results demonstrate that extremal cluster prototypes are more distinctive and hence more interpretable than central ones.

1 Introduction

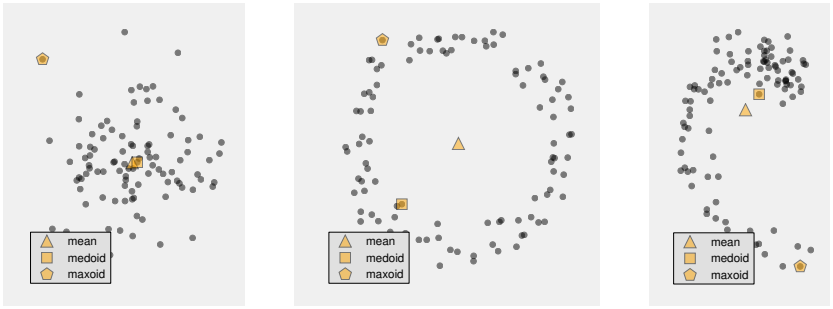
In this paper, we introduce a novel, prototype-based clustering algorithm. Since numerous such algorithms exist already [1, 12], our main goal is to fathom the potential of a paradigm that differs from existing prototype-based methods.

Whereas most prototype-based clustering algorithms produce prototypes that represent modes of a distribution of data (notable examples include the k-means procedure, the mean-shift algorithm, self organizing maps, or DBSCAN [7, 9, 11, 15]), our algorithm determines cluster prototypes that are extreme rather than central. They reside on the convex hull of their corresponding clusters and, in addition, are as far apart as possible.

The idea for this approach was motivated by research on efficient archetypal analysis, a matrix factorization technique that expresses a data set in terms of convex combinations of points on the data convex hull [5, 8, 13, 17]. The resulting representations are easily interpretable by human analysts [8, 19, 21], allow for clustering, and can facilitate classification. However, as their computation involves demanding optimization problems, the quest for more efficient methods and heuristics has become an active area of research [5, 6, 16, 18].

In the following, we first define the notion of the *maxoid* of a data set, prove that it will be furthest from the sample mean and necessarily coincides with a vertex of the data convex hull. We then introduce a simple and efficient clustering algorithm based on maxoids. It can be understood as a variant of the popular k-means procedure, however, whereas k-means determines cluster prototypes based on local information, our approach assumes a global view and selects the

 In: R. Bergmann, S. Görg, G. Müller (Eds.): Proceedings of the LWA 2015 Workshops: KDML, FGWM, IR, and FGDB. Trier, Germany, 7.-9. October 2015, published at http://ceur-ws.org



(a) Gaussian blob (b) ring (c) spiral

Fig. 1: Three data sets and their means, medoids, and maxoids.

prototype of a cluster with respect to those of other clusters. In experiments with synthetic and real world data, we illustrate the behavior of this algorithm and observe that it yields prototypes which are more distinct and hence more amenable to human interpretation than those produced by k-means.

2 Means, Medoids, and Maxoids

In this section, we briefly recall the concepts of the sample mean and sample medoid, introduce the idea of the sample maxoid, and review its characteristics.

Consider a finite set $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^m$ of data points. The *sample mean*

$$\boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} \boldsymbol{x}_i. \tag{1}$$

is arguably the most popular summary statistic of such data. The closely related concept of the *sample medoid*, however, seems less well known. It is given by

$$\mathbf{m} = \operatorname*{argmin}_{\boldsymbol{x}_j} \frac{1}{n} \sum_{i=1}^{n} \left\| \boldsymbol{x}_j - \boldsymbol{x}_i \right\|^2 \tag{2}$$

and coincides with the data point $\boldsymbol{x}_j$ whose average distance to all other points is smallest which is to say that it is the data point closest to the mean [3, 14].

Yet, our focus in this paper is not on central tendencies but on extremal characteristics of a set of data. To make this notion precise, we introduce the idea of the *sample maxoid* and define

Definition 1. *The maxoid of a set $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^m$ is given by*

$$\boldsymbol{m} = \operatorname*{argmax}_{\boldsymbol{x}_j} \frac{1}{n} \sum_{i=1}^{n} \left\| \boldsymbol{x}_j - \boldsymbol{x}_i \right\|^2. \tag{3}$$

Apparently, this definition reverses that of the sample medoid in that it replaces minimization by maximization. It is thus straightforward to prove

Lemma 1. *Given a set $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n$ of real valued data vectors, let $\boldsymbol{\mu} = \frac{1}{n}\sum_i \boldsymbol{x}_i$ be the sample mean and $\|\cdot\|$ be the Euclidean norm. Then*

$$\frac{1}{n}\sum_i \left\|\boldsymbol{x}_j - \boldsymbol{x}_i\right\|^2 \geq \frac{1}{n}\sum_i \left\|\boldsymbol{x}_k - \boldsymbol{x}_i\right\|^2 \tag{4}$$

implies that

$$\left\|\boldsymbol{x}_j - \boldsymbol{\mu}\right\|^2 \geq \left\|\boldsymbol{x}_k - \boldsymbol{\mu}\right\|^2. \tag{5}$$

That is, the maxoid $\boldsymbol{m}$, i.e. the point $\boldsymbol{x}_j \in \mathcal{X}$ with the largest average distance to all other points in $\mathcal{X}$, is farthest from the sample mean $\boldsymbol{\mu}$.

Proof. Note that the left hand side of (4) can be written as

$$\frac{1}{n}\sum_i \left\|\boldsymbol{x}_j - \boldsymbol{x}_i\right\|^2 = \frac{1}{n}\sum_i \left\|(\boldsymbol{x}_j - \boldsymbol{\mu}) - (\boldsymbol{x}_i - \boldsymbol{\mu})\right\|^2.$$

Expanding the squared Euclidean distances in this sum, we have

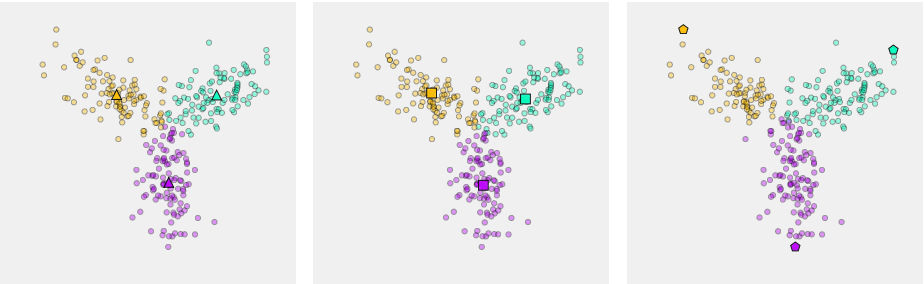
$$\begin{aligned}
&\frac{1}{n}\sum_i \left(\left\|\boldsymbol{x}_j - \boldsymbol{\mu}\right\|^2 + \left\|\boldsymbol{x}_i - \boldsymbol{\mu}\right\|^2 - 2(\boldsymbol{x}_j - \boldsymbol{\mu})^T(\boldsymbol{x}_i - \boldsymbol{\mu})\right) \\
&\quad = \left\|\boldsymbol{x}_j - \boldsymbol{\mu}\right\|^2 + \frac{1}{n}\sum_i \left\|\boldsymbol{x}_i - \boldsymbol{\mu}\right\|^2 - 2(\boldsymbol{x}_j - \boldsymbol{\mu})^T \frac{1}{n}\sum_i (\boldsymbol{x}_i - \boldsymbol{\mu}) \\
&\quad = \left\|\boldsymbol{x}_j - \boldsymbol{\mu}\right\|^2 + \frac{1}{n}\sum_i \left\|\boldsymbol{x}_i - \boldsymbol{\mu}\right\|^2 - 2(\boldsymbol{x}_j - \boldsymbol{\mu})^T(\boldsymbol{\mu} - \boldsymbol{\mu}) \\
&\quad = \left\|\boldsymbol{x}_j - \boldsymbol{\mu}\right\|^2 + \frac{1}{n}\sum_i \left\|\boldsymbol{x}_i - \boldsymbol{\mu}\right\|^2.
\end{aligned}$$

Since these arguments also apply to the right hand side of (4), the inequality in (4) can be cast as

$$\left\|\boldsymbol{x}_j - \boldsymbol{\mu}\right\|^2 + \frac{1}{n}\sum_i \left\|\boldsymbol{x}_i - \boldsymbol{\mu}\right\|^2 \geq \left\|\boldsymbol{x}_k - \boldsymbol{\mu}\right\|^2 + \frac{1}{n}\sum_i \left\|\boldsymbol{x}_i - \boldsymbol{\mu}\right\|^2$$

which is to say that $\left\|\boldsymbol{x}_j - \boldsymbol{\mu}\right\|^2 \geq \left\|\boldsymbol{x}_k - \boldsymbol{\mu}\right\|^2$. □

Given this result, it is easy to understand the behavior of the means, medoids, and maxoids in Fig. 1. In particular, we note a caveat for analysts working with centroid methods: the sample mean is always is located in the center of the data, yet, in cases where there is no clear mode, it is rather far from most data.The medoid may or may not be close to the mean but always coincides with a data point. The maxoid, too, always coincides with a data point but its behavior seems not to depend on whether or not there is a mode. In fact, we can prove the following



(a) k-means clusters (b) k-medoids clusters (c) k-maxoids clusters

Fig. 2: A simple data set consisting of three Gaussian blobs and results obtained from k-means, k-medoids, and k-maxoids clustering.

Lemma 2. *The maxoid of a finite set $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n$ of real valued data vectors coincides with a vertex of the convex hull of $\mathcal{X}$.*

Proof. The maxoid of $\mathcal{X}$ is the maximizer of the convex function

$$f(\boldsymbol{x}) = \frac{1}{n} \sum_{\boldsymbol{x}_i \in \mathcal{X}} \left\| \boldsymbol{x} - \boldsymbol{x}_i \right\|^2. \tag{6}$$

The domain of f is given by the discrete set $\mathcal{X}$ which defines a polytope, that is, a convex set of finitely many vertices. By Jensen's inequality, the maximum of a convex function over a convex set is attained at a vertex. □

3 From k-Means Clustering to k-Maxoids Clustering

Having familiarized ourselves with means, medoids, and maxoids, we ever so briefly revisit k-means clustering and then present our idea for how to extend it towards k-maxoid clustering.

In the simplest setting, k-means clustering considers a set of n data points $\mathcal{X} = \{\boldsymbol{x}_i\}_{i=1}^n \subset \mathbb{R}^m$ and attempts to determine a set $\mathcal{C} = \{\mathcal{C}_\kappa\}_{\kappa=1}^k$ of k clusters where $\mathcal{C}_\kappa \subset \mathcal{X}$ such that data points within a cluster are similar. In order to assess similarity, the algorithm represents each cluster by its mean $\boldsymbol{\mu}_\kappa$ and assigns data point $\boldsymbol{x}_i$ to cluster $\mathcal{C}_\kappa$ if $\boldsymbol{\mu}_\kappa$ is the closest mean. This idea reduces clustering to the problem of finding appropriate means which can be formalized as solving

$$\operatorname*{argmin}_{\boldsymbol{\mu}_1,\ldots,\boldsymbol{\mu}_k} \sum_{i=1}^{k} \sum_{\boldsymbol{x}_j \in \mathcal{C}_\kappa} \left\| \boldsymbol{x}_j - \boldsymbol{\mu}_\kappa \right\|^2. \tag{7}$$

Since this may prove surprisingly difficult [2], k-means clustering is typically realized using the following greedy optimization procedure:

1. initialize cluster means $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k$
2. repeat until convergence
   (a) determine all clusters

$$\mathcal{C}_\kappa = \left\{ \boldsymbol{x}_i \;\middle|\; \left\|\boldsymbol{x}_i - \boldsymbol{\mu}_\kappa\right\|^2 \leq \left\|\boldsymbol{x}_i - \boldsymbol{\mu}_\lambda\right\|^2 \right\} \tag{8}$$

   (b) update all cluster means

$$\boldsymbol{\mu}_\kappa = \frac{1}{|\mathcal{C}_\kappa|} \sum_{\boldsymbol{x}_i \in \mathcal{C}_\kappa} \boldsymbol{x}_i \tag{9}$$

Looking at this procedure, its adaptation towards k-medoids clustering is obvious: we simply have to replace the computation of means by that of medoids and use cluster medoids instead of means in (8). The extension towards meaningful k-maxoids clustering is straightforward, too, but not quite as obvious.

Assuming that k data points have been randomly selected as initial maxoids, we may of course cluster the data with respect to their distance to the maxoids. This is again in direct analogy to (8). However, updating the maxoids only w.r.t. the data points in their corresponding clusters may fail to produce reasonable partitions of the data since initially selected maxoids may be close to each other so that one (or several) of them may dominate the others in the subsequent cluster assignment. Our idea is thus to update maxoids not only w.r.t. the data in their cluster but also w.r.t. to the maxoids. That is, for the update step, we propose to select the new maxoid of cluster C_κ as the data point in C_κ that is farthest from the maxoids in the other clusters. Formally, this idea amounts to solving the following constrained minimizing problem

$$\begin{aligned} &\operatorname*{argmin}_{\boldsymbol{m}_1,\ldots,\boldsymbol{m}_k} \sum_{i=1}^{k} \sum_{\boldsymbol{x}_j \in \mathcal{C}_\kappa} \left\|\boldsymbol{x}_j - \boldsymbol{m}_\kappa\right\|^2 \\ &\qquad \text{s.t.} \quad \boldsymbol{m}_\kappa = \operatorname*{argmax}_{\boldsymbol{x}_j \in \mathcal{C}_\kappa} \sum_{\lambda \neq \kappa} \left\|\boldsymbol{x}_j - \boldsymbol{m}_\lambda\right\|^2. \end{aligned} \tag{10}$$

which is easy to recognize as a variant of the problem in (7). The corresponding greedy optimization procedure is shown in algorithm 1.

Figure 2 shows how k-means, k-medoids, and k-maxoids clustering perform on a data set consisting of three blob-like components. Setting $k = 3$, all three methods reliably identify the latent structures in these data. Observable differences are miniscule and arguably negligible in practice.

However, an important question is how k-maxoids clustering will deal with situations where not all of the clusters contained in a data set are close to the data convex hull. To illustrate this problem and answer the question, Fig. 3 shows a set of 2D data consisting of five clusters where one of them is situated in between the others and does not contain any point on the the data convex hull. The figure illustrates how the updates in algorithm 1 cause five randomly selected maxoids to quickly move away from each other; in fact, in this example, the algorithm converged to a stable clustering within only four iterations.

Algorithm 1 k-maxoids clustering

Require: discrete set $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \subset \mathbb{R}^m$ and parameter $k \in \mathbb{N}$

initialize iteration counter $t \leftarrow 0$ and cluster maxoids $\boldsymbol{m}_1^{(0)}, \boldsymbol{m}_2^{(0)}, \ldots, \boldsymbol{m}_k^{(0)}$

while not converged **do**

determine all clusters

$$\mathcal{C}_\kappa^{(t)} = \left\{ \boldsymbol{x}_i \;\middle|\; \left\|\boldsymbol{x}_i - \boldsymbol{m}_\kappa^{(t)}\right\|^2 \leq \left\|\boldsymbol{x}_i - \boldsymbol{m}_\lambda^{(t)}\right\|^2 \right\}$$

update all cluster maxoids

$$\boldsymbol{m}_\kappa^{(t)} = \operatorname*{argmax}_{\boldsymbol{x}_i \in \mathcal{C}_\kappa} \sum_{\lambda \neq \kappa} \left\|\boldsymbol{x}_i - \boldsymbol{m}_\lambda^{(t)}\right\|^2$$

increase iteration counter $t \leftarrow t + 1$

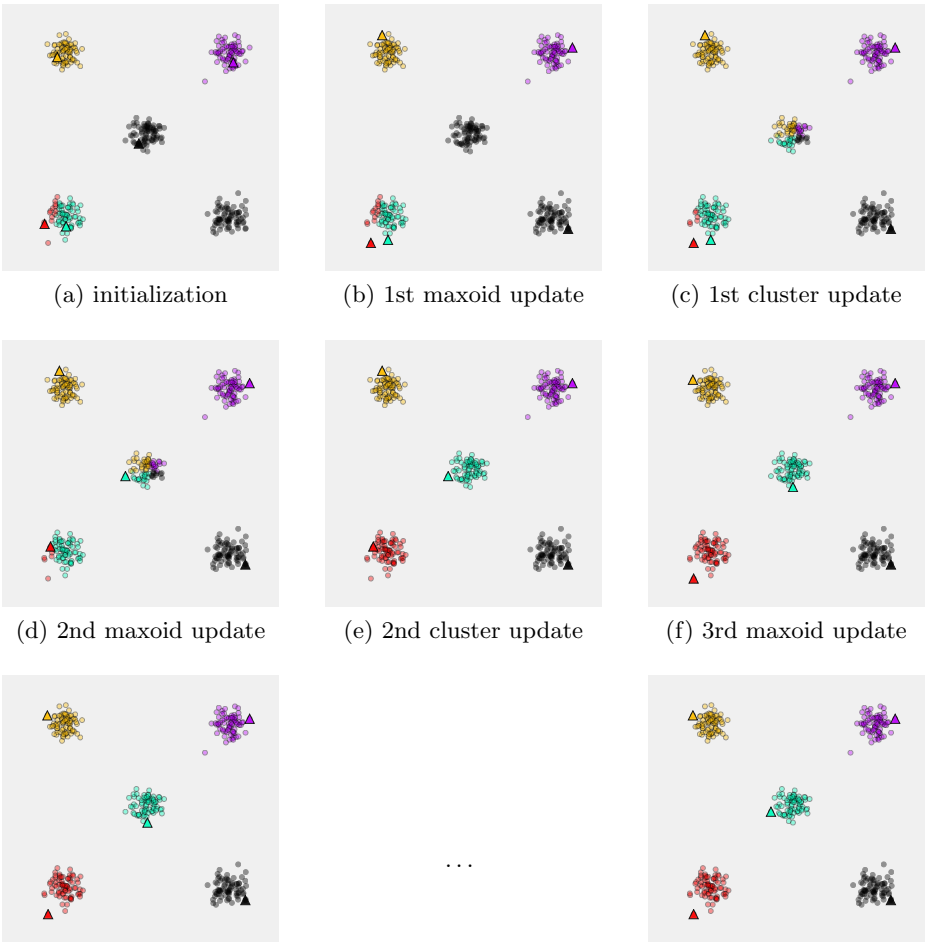
Although we observed this kind of efficiency in other experiments as well, the important point conveyed by this example is that the idea of clustering according to extremes works well even if there are substructures who cannot possibly be represented by prototypes on the data convex hull. This is, again, due to the fact that algorithm 1 inherently causes selected maxoids to be as far apart as possible.

In order to illustrate that extremal cluster prototypes may be more easily interpretable to human analysts than central ones, we conducted an experiment with the CBCL data set of face images[1] which contains 2429 portraits of people each of a resolution of 19×19 pixels. We turned each image into a 361 dimensional vector and applied k-means, k-medoids, and k-maxoids clustering where $k = 9$. The resulting prototypes in Fig. 4 clearly highlight the differences between the three approaches.

Figure 4(b) shows the prototypes returned by k-means clustering. They represent the average face of each cluster and, since each cluster contains several hundred images, are blurred to an extent that makes it difficult to assign distinctive characteristics to these prototypes. A similiar observation applies to the results produced by k-medoids clustering shown in Fig. 4(b). Here, the prototypes correspond to actual data points yet still appear rather similar. The prototypes in Fig. 4(c), on the other hand, resulted from k-maxoids clustering and show clearly distinguishable visual characteristics. Again each corresponding cluster contains several hundred images, yet their prototypes coincide with actual data points far from one another. It is rather easy to identify these faces as prototypes of pale or dark skinned people, of people wearing glasses, sporting mustaches, or having been photographed under varying illumination conditions.

In the next section, we will present and discuss an example of a real world application which further highlights this favorable property of clustering with extremes, namely the property of producing interpretable results.

[1] CBCL Face Database #1, MIT Center for Biological and Computation Learning, http://www.ai.mit.edu/projects/cbcl



(a) initialization (b) 1st maxoid update (c) 1st cluster update

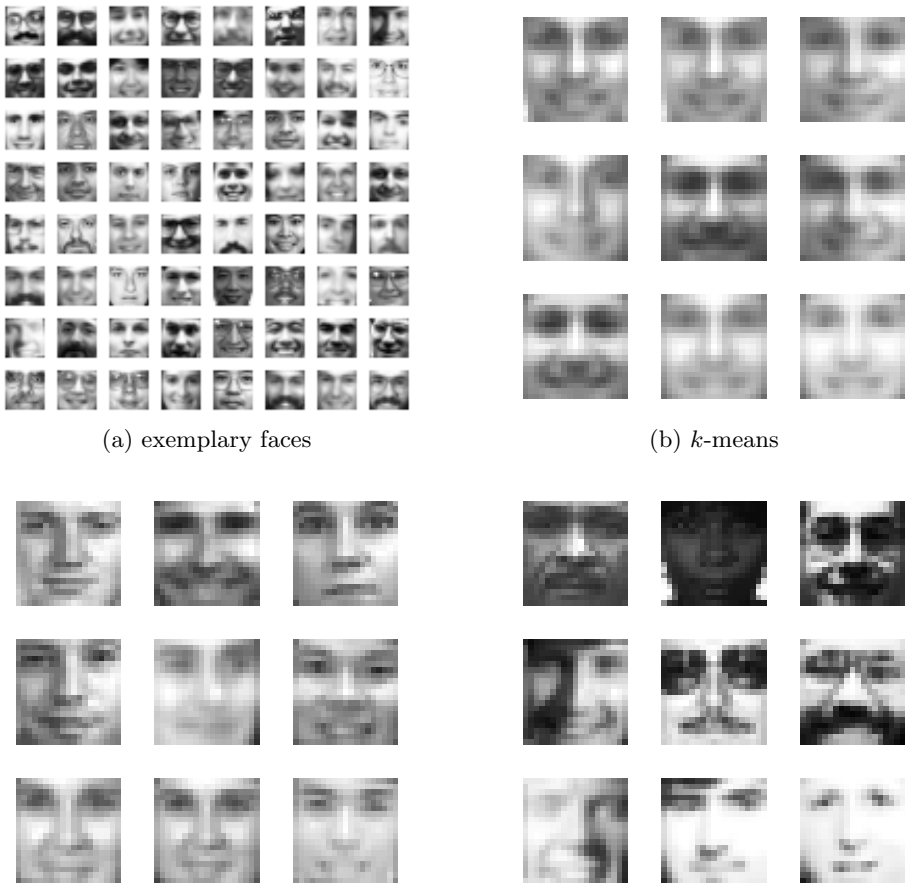
(d) 2nd maxoid update (e) 2nd cluster update (f) 3rd maxoid update

(g) 3rd cluster update ... (h) final result

Fig. 3: Convergence behavior of k-maxoid clustering applied to a 2D data set containing five blob like clusters. Started with a random initialization of maxoids, the algorithm quickly moves them apart and converges within four iterations.

4 A Practical Application: Player Preference Profiling in the Online Game Battlefield 3

With the rise of mobile, console, and PC based games that operate on a so called freemium model, the problem of understanding how players interact with games has become a major aspect of the game development cycle [4, 10, 19, 20]. In this context, analytics provides actionable insights as to player behaviors and allows developers and publishers to quickly adjust their content with respect to the



(a) exemplary faces (b) k-means

(c) k-medoids (d) k-maxoids

Fig. 4: Clustering with $k = 9$ prototypes on the CBCL data base of face images. (a) examples of 64 face images in this data collection which illustrate the range of appearances. (b) k-means clustering produces cluster prototypes with are the means of the corresponding clusters. (c) k-medoids clustering determines prototypes that are actual data points closest to the local mean. (d) k-maxoids clustering yields cluster prototypes that are extremal data points and therefore appear more distinguishable to human observers than means or medoids.

outcomes they receive and thus to increase sales and monetization rates. In this section, we apply the k-maxoids algorithm to a game analytics task, namely the problem of deriving interpretable player profiles from analyzing vehicle usage data of Battlefield 3.

Battlefield 3 is a first person shooter military simulation game published by Electronic Arts in the Fall of 2011 as the eleventh installment in the Battlefield Series which, as of this writing, has a history of over 15 years. The game offers a single- and multi-player game-play experience where the former is composed of a storyline that allows the player to control variety of military characters in different real world locations and the latter puts the player in a imaginary war between the United States of America (USA) and the Russian Federation (RF). The combination of rich storyline, realistic graphics, flexibility through numerous manageable in-game components (such as vehicles and character customization), and the ability of supporting matches with large number of players has made the game one of the most played titles in its genre. Compared to its competitors, one of the most distinguishable features of the Battlefield series is the unique vehicle experience which allows the players to control air-, land-, water-based, and stationary vehicles.

The data we use in this study is a collection of vehicle usage logs of a random sample of 22,000 Battlefield 3 players which we obtained using a Web-based API for the Player Stats Network (https://p-stats.com/). In order to extract vehicle usage profiles from this data that can reveal how players interact with vehicle, we used accumulated activity statistics as to time-spent, number of character kills, and vehicle destroys made with the available 43 vehicles in the game.

Running the k-maxoids algorithm on our data set, we obtained interpretable player profiles that are semantically distinguishable from each other. In Fig. 5, we an example of k-maxoids cluster prototypes indicating different player preferences for vehicles in Battlefield 3. For each maxoid, we also indicate the percentage of players it represents.

Upon a closer look at the maxoids, we observe entirely distinct player profiles each representing different preferences for vehicles in the game. The first maxoid represents a pilot player behavior, that is, a behavior where players spend most of their vehicle time flying multirole fighter jets (F-18 and Su-35) and attack jets (A-10 Thunderbolt and Su 25) where the same vehicle ordering applies for both kills and destroys. Specifically for this particular maxoid the total flying time is 982 hours which is actually comparable to the average yearly flight time of experienced pilots in real life. It is important to note that the players in this cluster particularly chose to fly with the equivalent (*counterpart*) planes for American and Russian teams, which, during gameplay, creates a balance between two teams. In other words, the prototype indicates a habit of choosing a particular type of vehicle during a game. Indeed, behavioral patterns like this are also observed for the profiles represented by the other prototypes.

The second and the fourth most populated profiles represent a preference for land oriented vehicles. Again, counterpart-vehicle mastering is also observed where the players of the second and fourth profiles prefer to mostly use the counterpart heavy and light tanks the American M1 Abrahams and the Russian T-90 and the infantry fighting vehicles BMB and LAV respectively.

A more distinct tower defense behavior is observed for the third maxoid where players in the corresponding cluster spend 85% of their time on two counterpart

Fig. 5: Seven player vehicle usage profiles obtained from k-**maxoids** clustering. Each column visualizes a cluster prototype $\boldsymbol{m}_\kappa$ which indicates the most popular vehicles in the corresponding cluster of players. Note that prototypes are sorted according to the percentage of players they represent and that we show the top 5 elements of each prototype.

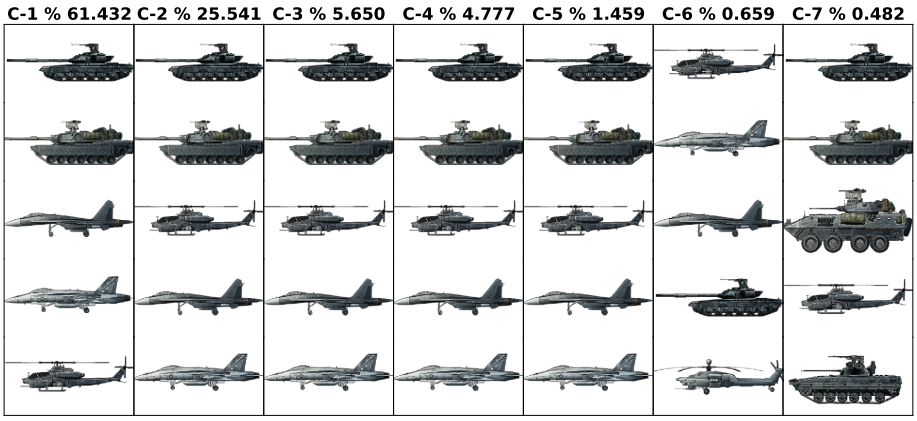


Fig. 6: Seven player vehicle usage profiles resulting from k-**means** clustering.

stationary anti-aircraft vehicles. Similar to the second profile we observe the use of two counterpart heavy tanks M1 Abrahams and T-90 for this profile as well.

The fifth profile, on the other hand, shows a helicopter pilot profile where the maxoid player in this cluster spends 68% of his time flying light fighter helicopters AH 6 and Z 11.

Finally, the least populated profile, represented by the right most maxoid in the figure, indicates that some players spend most of their time on the heavy counterpart tanks and the light fighting helicopters.

For comparison, we present results obtained from k-means clustering in Fig. 6. Similar to the face clustering example discussed in the previous section, we find k-means profiles to indicate general averages or mixed preferences for (counterpart) heavy tanks, jets, and light fighting helicopters where each of the vehicles in a prototype ranks high among the overall most frequently played vehicles in our data set. Hence, while k-means results represent average behavioral profiles (as already hinted at in [10]), the maxoids found by k-maxoids clustering represent more extreme or archetypal behavior that can help game developers to develop a deeper understanding of truly different types of user preferences and profiles that cannot be captured k-means clustering but are important w.r.t. balancing the game mechanics.

5 Conclusion

In this paper, we investigated the idea of clustering according to extremal rather average properties of data points. In particular, we defined the notion of the maxoid of a data set and presented an algorithm for k-maxoids clustering. This algorithm can be understood as a modification of the classical k-means procedure, where, in contrast to the classical approach, we determine cluster prototypes not only w.r.t. the data points in a cluster but w.r.t. the prototypes of other clusters. In a couple of didactic examples, we illustrated the behavior of this algorithm and then applied it to a practical problem in the area of game analytics.

In our didactic examples, as well as in our real world application, we observed our algorithm to produce cluster prototypes that are well distinguishable from one another and are thus more easily interpretable for human analysts. This property of clustering with extremes is particularly interesting for practitioners in game analytics for it allows them to quickly identify potentially imbalanced game mechanics.

In addition to these kinds of practical applications of k-maxoids clustering, we are currently investigating more theoretical aspects of its use. In particular, we examine its use as a mechanism to preselect archetypes for efficient archetypal analysis and hope to be able to report corresponding results soon.

References

1. Aggarwal, C., Reddy, C. (eds.): Data Clustering: Algorithms and Applications. Chapman & Hall/CRC (2013)
2. Aloise, D., Deshapande, A., Hansen, P., Popat, P.: NP-Hardness of Euclidean Sum-of-Squares Clustering. Machine Learning 75(2) (2009)
3. Bauckhage, C.: NumPy / SciPy Recipes for Data Science: k-Medoids Clustering. researchgate.net (Feb 2015), https://dx.doi.org/10.13140/2.1.4453.2009
4. Bauckhage, C., Kersting, K., Sifa, R., Thurau, C., Drachen, A., Canossa, A.: How Players Lose Interest in Playing a Game: An Empirical Study Based on Distributions of Total Playing Times. In: Proc. IEEE CIG (2012)

5. Bauckhage, C., Thurau, C.: Making Archetypal Analysis Practical. In: Denzler, J., Notni, G. (eds.) Pattern Recogntion. LNCS, vol. 5748. Springer (2009)
6. Chang, C.I., Wu, C.C., Liu, W.M., Ouyang, Y.C.: A New Growing Method for Simplex-Based Endmember Extraction Algorithm. IEEE Trans. on Geoscience and Remote Sensing 44(10) (2006)
7. Cheng, Y.: Mean Shift, Mode Seeking, and Clustering. IEEE Trans. on Pattern Analysis and Machine Intelligence 17(8) (1995)
8. Cutler, A., Breiman, L.: Archetypal Analysis. Technometrics 36(4) (1994)
9. Drachen, A., Canossa, A., Yannakakis, G.: Player Modeling using Self-Organization in Tomb Raider: Underworld. In: Proc. IEEE CIG (2009)
10. Drachen, A., Sifa, R., Bauckhage, C., Thurau, C.: Guns, Swords and Data: Clustering of Player Behavior in Computer Games in the Wild. In: Proc. IEEE CIG (2012)
11. Ester, M., H.-P.Kriegel, Sander, J., Xu, X.: A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. In: Proc. ACM KDD (1996)
12. Estivill-Castro, V.: Why So Many Clustering Algorithms: A Position Paper. ACM SIGKDD Explorations Newsletter 4(1) (2002)
13. Eugster, M., Leisch, F.: Weighted and Robust Archetypal Analysis. Computational Statistics & Data Analysis 55(3) (2011)
14. Kaufman, L., Rousseeuv, P.: Clustering by Means of Medoids. In: Dodge, Y. (ed.) Statistical Data Analysis Based on the L1-Norm and Related Methods. Elsevier (1987)
15. MacQueen, J.: Some Methods for Classification and Analysis of Multivariate Observations. In: Proc. Berkeley Symp. on Mathematical Statistics and Probability (1967)
16. Miao, L., Qi, H.: Endmember Extraction From Highly Mixed Data Using Minimum Volume Constrained Nonnegative Matrix Factorization. IEEE Trans. on Geoscience and Remote Sensing 45(3) (2007)
17. Morup, M., Hansen, L.: Archetypal Analysis for Machine Learning and Data Mining. Neurocomputing 80 (2012)
18. Ostrouchov, G., Samatova, N.: On FastMap and the convex hull of multivariate data: toward fast and robust dimension reduction. IEEE Trans. on Pattern Analysis and Machine Intelligence 27(8) (2005)
19. Sifa, R., Bauckhage, C., Drachen, A.: Archetypal game recommender systems. In: Proc. LWA KDML (2014)
20. Sifa, R., Ojeda, C., Bauckhage, C.: User Churn Migration Analysis with DEDICOM. In: Proc. ACM RecSys (2015)
21. Thurau, C., Kersting, K., Wahabzada, M., Bauckhage, C.: Descriptive Matrix Factorization for Sustainability: Adopting the Principle of Opposites. Data Mining and Knowledge Discovery 24(2) (2012)