



K-Nets: Clustering through nearest neighbors networks

Ioannis A. Maraziotis^{a,c,*}, Stavros Perantonis^a, Andrei Dragomir^b, Dimitris Thanos^c

^a Institute of Informatics and Telecommunications, National Center for Scientific Research “Demokritos”, Greece

^b Singapore Institute for Neurotechnology (SINAPSE), Center of Life Sciences, National University of Singapore, Singapore

^c Center of Basic Research, Biomedical Research Foundation, Academy of Athens, Greece

ARTICLE INFO

Article history:

Received 25 September 2017

Revised 10 October 2018

Accepted 16 November 2018

Available online 17 November 2018

ABSTRACT

Importance of clustering is signified by its applications in many scientific fields and the uninterrupted presence of related algorithmic methods in the literature for more than half a century. The majority of these methods have complex operation and most of the times aim on a specific application field. Here we present K-Networks (K-Nets), a deterministic algorithm based on a network structure whose nodes represent data points and can intrinsically work within a single or multi-layer architecture. The number of interconnections of each node with the rest of the network indicates the clustering resolution degree and can be the same or different in every layer. K-Nets automatically detects symmetric structures in data that can be utilized to recover clusters of different size, shape, or partitions composed of a specific number of clusters. We provide evidence that K-Nets performance compares favorably with other well established clustering algorithms in a wide range of fields, while it can be employed in cases when data dimensionality prohibits their application.

© 2018 Elsevier Ltd. All rights reserved.

1. Introduction

Clustering is the process of assigning a set of data objects to a number of groups, or clusters, based on certain similarities they exhibit and is essential in many fields of science. The importance of clustering has lead over the years to the development of a large number of different clustering algorithms [1,2]. The diversity, and in most of the cases, application-oriented nature of these methods is rooted in the large degree of complexity inherent to clustering.

The diversity of clustering methods is portrayed by the fact that they can differ in one or several of their properties including but not limited to: (I) whether they are stochastic or deterministic, (II) the type of clusters they can recover, (III) the way they represent clusters, (IV) the type, range and sensitivity of the resolution parameter, (V) the type of similarity measures they can be utilized with, (VI) the way they handle time and space complexity, (VII) their sensitivity to noise. With the exception of some straightforward decisions (i.e. III) selecting an appropriate method for a specific application is generally a non-trivial task [3].

Based on the aforementioned aspects, we have devised K-Networks (or K-Nets), a simple and fast exemplar-based algorithm that incorporates the ability of density-based approaches to find natural clusters in the data by utilizing the technique of nearest neighbors [1,4–6]. K-Nets [7], similar to algorithms like Affinity Propagation (AP) [8], and conversely to K-means [9,3], or K-centers [9], is deterministic and hence immune to the problematic effect of repetitive executions from different initializations in quest of an optimal solution.

K-Nets can be utilized under a versatile framework benefiting from two classical clustering paradigms: exploratory clustering and clustering into a specified number of clusters. Specifically, unlike algorithms such as K-means, K-centers, it promotes exploratory data analysis in cases when the number of clusters is not known a-priori, by recovering landmark partitions based on the value of a positive integer k ($k > 0$) indicating the degree of clustering resolution. Optionally, it accepts as input in conjunction with k , a user-defined number of clusters in which the dataset will be partitioned to. The landmark partitions extracted by K-Nets, unlike partitions extracted by algorithms like AP, DBSCAN [10], Meanshift [11], represent a fraction of the solution space composed of all possible partitions and hence endorse its efficient navigation.

The nodes of a single-layer K-Network, refereed to as pre-clusters, are constructed based on the input k value and correspond to near symmetric K -sized neighborhoods formulated

* Corresponding author.

E-mail addresses: imaraziotis@iit.demokritos.gr (I.A. Maraziotis), sp@iit.demokritos.gr (S. Perantonis), lsiandr@nus.edu.sg (A. Dragomir), thanos@bioacademy.gr (D. Thanos).

around each point of the dataset. The number of exemplars formulating a landmark partition is determined by the winners of a non-iterative competitive process based on a simple overlapping criterion, among the pre-clusters. In cases that a specific number of clusters is requested, a cascade of K-Nets with decreasing k values (i.e. initiating from the input k) is built sequentially. Each new member of the cascade provides a number of exemplars to the final solution until the requested number is reached or exceeded. These exemplars contribute to the construction of the nodes of the next K-net in the cascade but do not compete with them. In both cases, the output partition is finalized by an iterative phase that consists of either a single or a very small number of iterations, hence the algorithm concludes its operation in fractions of time compared to methods like AP.

The gradual construction of a partition composed of a specified number of clusters, differentiates K-Nets from algorithms like DB-SCAN or AP. To achieve the same goal with these methods an appropriate value of their resolution parameter(s) has to be detected through time consuming trial and error process [12]. Furthermore, the same approach described above allows K-Nets, in contrast to methods like AP, K-Centers or K-means, to identify non-isotropic and irregular (in terms of their size or shape) clusters present in certain data manifolds or degrees of clustering resolution [3,12,13].

K-Nets like most exemplar-based methods, can utilize various distance measures, such as the geodesic distance [14], and expand its application to non-linear spaces. However, the calculation of geodesic distances is most of the times problematic or impractical due to their augmented noise sensitivity that prevents their correct retrieval or their exceeding time and space complexity [15]. K-Nets can adequately address these problems, present also on spectral algorithms [16] like the SPK-means [17], through multi-layer architectures.

Based on the symmetric nature of landmark partitions, and the fixed range of its resolution parameter, a number of single-layer K-Nets, can be serially or parallel connected to formulate multi-layer architectures. The key idea for their construction is to compress the original dataset through the first layer(s) partition(s) and perform the actual exploratory analysis on the compressed dataset in the deepest, or last layer.

Multi-layer architectures inherit all properties of the single-layer K-Nets, while improving their space and time complexity as well as accelerate and simplify the successful utilization of metrics such as geodesic distances through their application on the last K-Nets layer.

2. Methods

2.1. Single layer K-Nets

One of the main goals in the design of K-Nets was to capture different aspects of the underlying data structure through a relatively small number of informative partitions (i.e. landmarks) extracted under different degrees of clustering resolution. Landmarks are utilized to cluster the dataset into any number of requested clusters. For clarity, in the followings we will describe separately the dual behavior of K-Nets allowing our algorithm to perform exploratory clustering (normal mode) and clustering into a specified number of clusters (exact mode), depending on the problem at hand. The description of single-layer K-Nets and their hybrid behavior, accomplished through the application of three distinct operational phases, will be followed by the methodology adapted for constructing multi-layer architectures through their serial and parallel connections. An implementation of the single layer K-Net can be found in [Appendix. A](#).

2.1.1. Normal operational mode (NOM)

A single layer K-Nets under NOM ([Fig. 1](#)) requires as input the data to be partitioned and a positive integer number k (i.e. $k > 1$), indicating the degree of clustering resolution. Its operation is composed of three sequential operational phases. During the first, or Construction phase, K-Nets creates a number of special clusters called pre-clusters, matching the number N of points in the dataset. Every pre-cluster PC_i is composed of $K = k + r$ points, where k is the number of the k nearest neighbors of point x_i and r is the number of any additional points, having the exact same distance from its k -th furthest neighbor. The construction of a pre-cluster is followed by the calculation of a score, indicating how dense or compact it is. The score equals the ratio of the sum of distances of the k nearest neighbors from the considered point to the total number K of members in the pre-cluster:

$$S_i = \frac{1}{K} \sum_{j=1}^k d_{ij} \quad (1)$$

In (1) we can utilize any distance metric like Euclidean distance to measure d_{ij} . The construction of the pre-clusters is followed by the Selection phase of the algorithm, during which only a fraction of the data points will be selected as input for the next phase. The number C of selected points, referred to as pre-exemplars, indicates the number of clusters in which K-Nets will partition the data set. This phase initiates by sorting pre-clusters based on their corresponding scores. Next, starting from the pre-cluster with the smallest score value, we run through all pre-clusters. A point is selected to be one of the pre-exemplars, if none of the members of its corresponding pre-cluster was part of a previously selected one.

The third, and final, assignment phase of K-Nets, is composed of two steps. In the first step clusters are formulated by assigning every data point of the dataset to its nearest pre-exemplar. The second step fine-tunes the clustering result by selecting the most central point of every cluster (i.e. the one that has minimum mean distance from all cluster members) as its new exemplar and subsequently reassigning the data points to their nearest exemplars. The second step can be repeated a number of times until convergence (i.e. no reassignments occur).

As mentioned above, the only input of K-Nets NOM is a positive integer parameter k , which indicates the degree of clustering resolution. The minimum value for resolution parameter k is 1 in which case every node is connected to itself and (since there is no overlapping among the pre-clusters) every data point will be selected as exemplar of a singleton cluster. For very large values of k (typically larger than one third of the total number of points in the dataset, based on our experimentation) the pre-clusters have a large degree of overlap and hence the final partition is composed of a single cluster.

A common characteristic of algorithms like AP and K-centers is their iterative nature. By breaking their operation too soon (in comparison to the iterations requested for convergence) the final set of exemplars and hence, in most of the cases, the final solution will be severely distorted. As we have seen, K-Nets follows a different path under which, like AP and unlike K-centers, all points are initially considered as candidate exemplars. However, in contrast to AP, the final iterative assignment phase, is utilized to fine-tune the output of the two previous non-iterative phases. Hence, in most of the cases, a single or only a few number of iterations are required for K-Nets to conclude their operation successfully.

The goal of the three K-Nets operational phases, is to perform clustering based on the centrality principle (centrality in terms of the mean distance of all cluster members from their corresponding exemplar). Specifically, the Construction phase builds highly overlapping symmetric pre-clusters centered around every pattern/sample of the dataset, while the Selection phase identifies the

Phase	Instructions
Construction	For every data point refereed to as pre-Exemplar (PE) I. Build pre-Cluster (PC) composed of its K nearest neighbors. II. Assign a score based on (1) and promote in S .
Selection	Sort S . For every PC with score from $\min(S)$ to $\max(S)$ I. If None of the current PC members belong to L 1. Insert current PE in M 2. Insert all members of current PC in L
Assignment	Consider the $ M $ PEs as exemplars I. Assign every point to its nearest exemplar. II. Break OR Optional: Set as exemplar for every cluster its most central point AND GOTO (I) until convergence.

Fig. 1. K-Nets Normal Operational Mode (NOM). Pseudo-code description of a single layer K-Nets under NOM, with input a similarity matrix, and a positive integer k indicating the resolution degree of the partition. S , L and M are initially empty lists utilized to store respectively PCs scores, and the selected PCs and PEs.

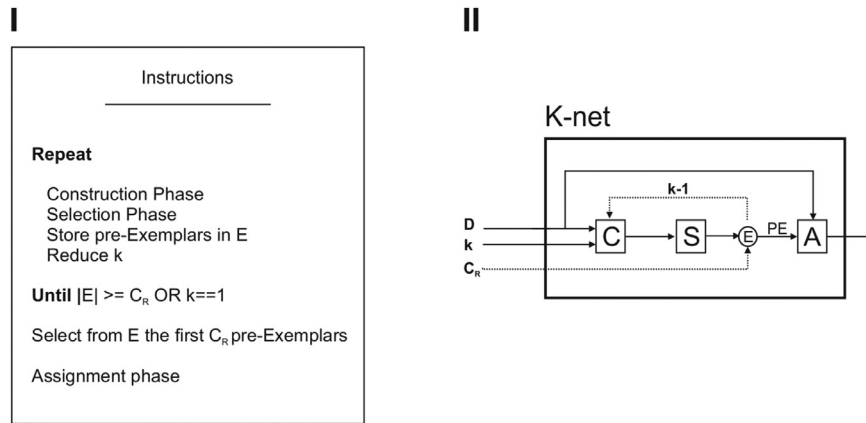


Fig. 2. K-Nets Exact Operational Mode (EOM). In (I) we can depict a pseudo-code description of K-Nets EOM (see Appendix A for an implementation). The difference between NOM (Fig. 1) and EOM is the number of times the Construction and Selection Phases, depicted in the block diagram (II) as (C) and (S) respectively and described in Fig. 1, are executed. Specifically, under EOM (C) and (S) are executed repeatedly with the same similarity matrix D , but with decreasing values of the resolution parameter k until the initially empty list (E) has a number of pre-exemplars larger or equal to the requested number of clusters C_R . The first C_R pre-exemplars are provided as input in an Assignment phase (Fig. 1). In the case that the number of selected pre-exemplars for the input k value equals C_R NOM and EOM are identical.

most compact/dense ones for the requested clustering resolution degree. The Assignment phase aims in resolving noise related distortions from the Centrality principle. These distortions mainly occur from points located on the borders among pre-clusters or from points that were members of not selected pre-clusters.

2.1.2. Exact operational mode (EOM)

In cases that a specific partition composed of C_R clusters is requested, and irrespectively of whether or not it is a landmark, the algorithm can accept two input values. The first input, is a k_L value under which K-Nets extracts a landmark partition with a number of clusters C_L near the neighborhood of C_R , without this being a prerequisite, and the second one is C_r . As can be deduced, in the case that C_R is equal to C_L EOM is simply NOM.

Under this hybrid framework (Fig. 2), the first two phases of the algorithm are executed once, as we described in the previous section, if the number of pre-exemplars extracted for k_L is larger than or equal to C_R . Alternatively, if the number of pre-exemplars is smaller than C_R , they are automatically repeated a number of cycles (or epochs) with decreasing values of k until the number of pre-exemplars equals or exceeds C_R .

In the latter case, by the end of the first epoch, a certain number of pre-exemplars has been selected. In the second epoch the same operation is repeated by preserving the pre-clusters whose corresponding points were initially selected as pre-exemplars. The new pre-clusters are formulated with a decreased value of the resolution parameter for all data points that were not selected as pre-exemplars in the first phase. Therefore, the pre-cluster of a data point that had been rejected in the first epoch will decrease its size and hence it could be considered as a pre-exemplar (i.e. by loosing its connection to a selected pre-cluster), in the new epoch. The same process is repeated a number of epochs until a number of pre-exemplars larger or equal to the requested C_R is reached.

As can be deduced, the minimum number of epochs in this case is two, while the maximum mainly depends on the relation between numbers C_R and C_L . If C_L is significantly smaller than C_R the connectivity of the initiating landmark will be relatively large and hence K-Nets could not reach C_R . The process of locating a C_L number “near” C_R is greatly simplified by the number of landmarks that represent a very small percentage (i.e. most of the

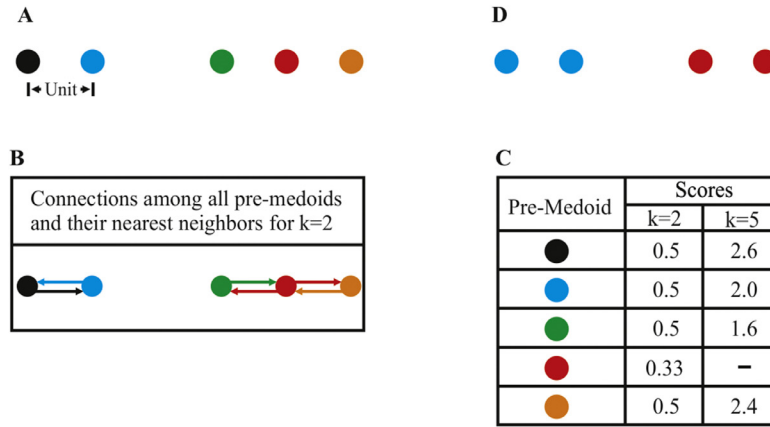


Fig. 3. Pre-clusters scoring of K-Nets. In (A) we present a toy-dataset. In table (B) we can depict the pre-clusters that are formulated for $k=2$. Based on the pre-clusters scoring presented in (C) we deduce that K-Nets will always select as pre-exemplar of the first cluster the red point. For the second cluster however, the algorithm cannot decide which one of the blue or black points should be considered as pre-exemplar since they both have the same score. In cases such as these K-Nets retains the same pre-clusters but assigns a new score utilizing information from all dataset (i.e. in this case $k=5$) to the points that share the same score value with other points (C). Under this framework the conflict is resolved and K-Nets will always provide blue as the pre-exemplar of the second cluster (D).

times less than 3%) of all possible partitions. In both of the two aforementioned cases the C_R pre-exemplars of the most compact pre-clusters, in terms of the score described in (1), are provided as input in the Assignment phase and K-Nets conclude its final phase as described in the previous section.

2.1.3. Detecting data instabilities

In certain datasets, and under specific degrees of clustering resolution, there can be cases under which more than one pattern can equally well be considered as the exemplar of a certain cluster. Commonly, this conflict occurs when a set of points have the exact same distance from two or more other reference points. Methods using one of their input parameters to provide partitions of different granularity cannot identify these cases. Hence, algorithms like AP, alter their normal deterministic operation and lead to instabilities under which for the same value of their parameters (i.e. preference), different number of clusters is extracted per execution [8]. K-Nets, based on the scoring system adopted in the first phase of its operation, automatically detects and resolves these types of data peculiarities (Fig. 3). Suppose that on a dataset consisting of N patterns, under a certain value of clustering resolution k_c , a number N_c of pre-clusters share the same score value S_c . In such a case, one of the following conditions might stand: (A) there are no common neighbors among the pre-clusters, (B) there are N_c common neighbors among the pre-clusters but at least one of them has a pre-cluster PC_r with a smaller score value than S_c , and finally (C) there are common neighbors but for none of them the condition described in (B) is valid.

As we can deduce from K-Nets operation, only under the last condition the algorithm cannot decide with absolute certainty the identity or number of final pre-exemplars. Specifically in (A) all of respective pre-clusters will be considered as pre-exemplars, in (B) none of the respective pre-clusters will be selected as a pre-exemplar, since either PC_r or some other pre-cluster with which PC_r shares common neighbor(s) has already been considered as pre-exemplar.

K-Nets resolves problematic cases (C), whenever detected, based on a partial score reassignment scheme, realized by the assignment of a new score on the N_c problematic pre-clusters, utilizing a larger amount of information. Specifically, in current K-Nets implementation, the new score for each one of the N_c pre-clusters, is based on a k value equal to the total number of points present in the data set. The new unique scores are used in the subsequent phases

of K-Nets operation while the members of the pre-clusters for the original k_c value will be retained for all pre-clusters.

At this point it should be mentioned that instead of the adapted global criterion (i.e. total number of points) we could have selected a local one (i.e. a fraction of the total number of points) and repeat the process for a number of times until we had no pre-clusters sharing the exact same score. Simulations performed on several different datasets (data not shown) showed no differences in the performance of the algorithm. Under the score reassignment phase we just described, instability problems will occur only in cases when two or more points are located in the exact same position of the input space, in which case selecting one over the other will bring no actual changes to the final outcome of the algorithm.

2.2. Multi-layer K-Nets

In this section we will describe how a number of single-layer K-Nets can be connected in series or parallel to produce multi-layer architectures.

2.2.1. Serial and parallel connections

A single-layer K-Nets can be regarded as a network whose input is the points of a dataset and the output is a subset of the input represented by the exemplars, controlled by the resolution parameter k . In serial connections a number of different single-layer K-Nets, simply referred to as layer, are connected in such a way so as the output of one to be the input of another and hence every layer outputs a compressed representation of its input.

Instead of just a single one, a layer can also be composed from a number of single-layer K-Nets connected in parallel and in this case it is referred to as parallel layer. In parallel connections, the original data set described through a matrix X , consisting of $N \times M$ dimensional points, is divided into D_c smaller data subsets of roughly equal size, each one described through a matrix $D_s \times M$, so as to have $D_s = N/D_c$ (see [Supplementary Materials–SM](#)). Subsequently, an elementary K-Nets structure is applied to each one of the D_c data subsets with the same value of resolution parameter for all of them. The exemplars from every K-Nets of the parallel layer are subsequently pooled and formulate a compressed data set provided as input to the next serially connected layer (Fig. 4B).

The only limitation in setting the size for each one of the data subsets is imposed by the available computational resources [SM]. As can be deduced, the same approach can be applied in cases we only have similarities amongst samples instead of

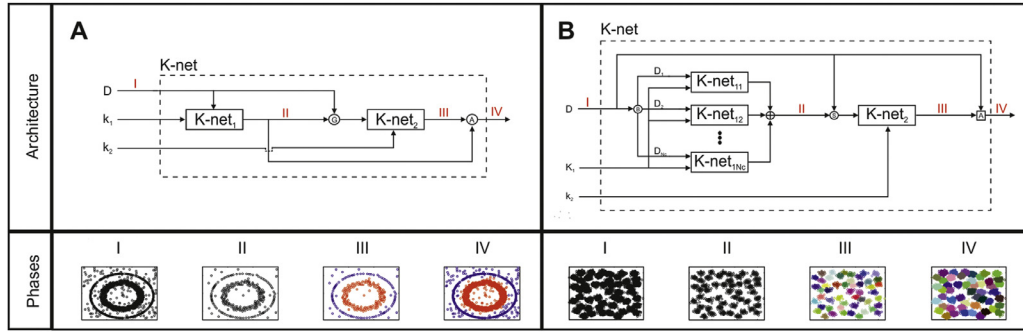


Fig. 4. Multi-layer K-Networks. Two-layer K-Nets architectures (i.e. dotted frames) produced by serial and/or parallel connections of single layer K-Nets (i.e. solid frames). In both cases K-Nets utilizes first layer to compress input (Phase I) into a smaller dataset composed of the extracted exemplars (Phase II). In both cases along with data K-Nets request as input two integers each one representing the clustering resolution of every layer. The differences in the two architectures are: (1) the first layer in (A) is composed of one while in (B) from an automatically derived number of single layer K-Nets connected in parallel, (2) in (A) we calculate the geodesic distances of the exemplars extracted in the first layer, while in (B) euclidean distances, (3) in (A) the members of the first layer clusters are assigned the label that the corresponding exemplars have in Phase III, while in (B) the exemplars extracted from Phase III are provided as pre-exemplars in an assignment component (Fig. 2) along with original data set.

the actual features values. In contrast to serial connections, the operation of each one of the D_s elementary K-Nets structures implicated in a parallel connection, is independent of the rest and hence can be utilized in parallel processing environments.

2.2.2. Multi-layer architectures

Depending on the available computational resources, we can design a large repertoire of different K-Nets architectures. In the followings, we will present simple K-Nets architectural schemas whose main goal is the illustration, through the simulations we will present in the Results section, of K-Nets capabilities.

One of the simplest architectures is to have two serially connected single layers, under which the output of the first is partitioned into a number of clusters in the second layer. The exemplars of the second layer are considered as pre-exemplars and are provided as input to an assignment component that will conclude the overall process using the original dataset. This final assignment phase is utilized to assign labels to all points of the dataset. Such a network can be realized in cases when we want to partition a dataset into a medium to small number of clusters and through it we can reduce the range of possible k values for the various partitions performed by the final layer (Table S2). It must be underlined at this point that the construction of multi-layer architectures is embedded into the three phases of the algorithm, as can be noted in Fig. 4A and B and does not consist of simply stacking individual single-layer K-Nets.

In this context, another simple schema would be to change the first single layer with a parallel layer (Fig. 4B). Parallel layers can be utilized in cases when data dimensionality constitute the use of a similarity matrix problematic, in terms of time complexity, or even impossible, in terms of space complexity. The main difference between this and the architecture discussed previously is that the input to the second layer is the union of all K-Nets outputs formulating the parallel layer. The aforementioned final assignment phase can be repeated a number of times (i.e. iterations) till convergence. However, given the efficiency of the previous layer partitions, and based on the results of numerous simulations on different datasets, in this study we have restricted the number of iterations to one.

In current K-Nets implementation we can apply any number of additional parallel layers between the parallel and the single layer of Fig. 4B (and Table S2).

Multi-layer K-Nets can be employed under various setups including to enhance the use of complex similarity measures like geodesic distances in cases when simple measures like Euclidean distance do not suffice. Methods for calculating geodesic distances

are both time consuming and especially sensitive in the selection of the parameter concerning the number of connections in the proximity graph which has to be build for their calculation [15, SM]. Nevertheless, applying their calculation on the last layer of a multi-layer K-Nets, resolves both problems to a large degree.

Specifically, under a two-layer K-Nets architecture (Fig. 4A), we provide as input a similarity matrix based on Euclidean distance. The network will partition the input dataset into a number of clusters based on the resolution parameter of the first layer. The output exemplars are utilized to build a geodesic distances-based similarity matrix, that is subsequently partitioned into a number of clusters based on the k value of the second layer. Finally, every pattern of the original data set is assigned to the label that its corresponding exemplar has in the second layer.

In multi-layer architectures the size of the compressed dataset to which a K-Nets layer is applied to, is controlled by the resolution parameter of the previous layer. As can be deduced, there is no need to know in advance the exact number of clusters that any of the two-layer architectures we described will realize in the first layer. The smaller is the value of k in the first layer, the larger the size of the compressed data set and hence the range of possible partitions (i.e. in terms of the number of clusters) that can be extracted utilizing the resolution parameter of the second layer. However, in cases when we want to test partitions composed of a medium or small number of clusters, utilizing a medium-size k value for the first layer(s) will further reduce the computational complexity of the last layer partition, without affecting the final result.

The time complexity of a multi-layer K-Nets architecture composed of any number of parallel layers is bounded by the complexity of the last layer K-Nets that will be applied on the largest portion of the input dataset. Hence, for D_s exemplars, the construction phase is bounded by the sorting operation applied to detect the nearest neighbors of every pattern and hence depending on the sorting algorithm it can be $O(D_s^2)$ or $O(D_s^2 \times \log D_s)$. The time complexity of the selection phase is $O(D_s)$. The complexity of the final assignment phase is $O(NCT)$ where N is the number of samples, C is the number of clusters, and T is the number of iterations that, as mentioned, is set to 1 for all simulations performed in this study. In the case of single-layer K-Nets D_s is replaced by N .

3. Results

We benchmarked the main properties of K-Nets by presenting results on synthetic and real datasets, in both operating modes, in comparison with well-established methods.

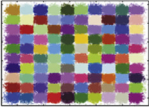
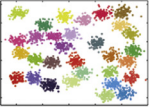
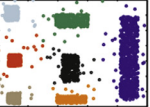
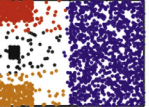

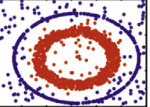
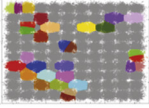
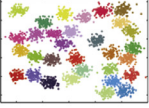
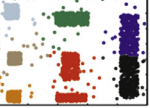
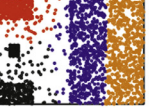

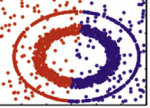
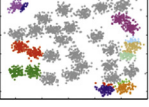
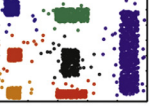
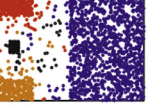

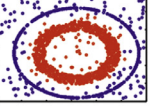

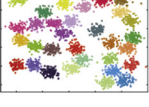
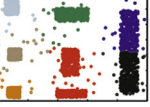
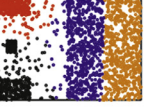

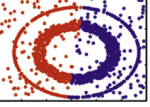

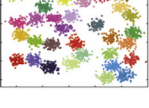
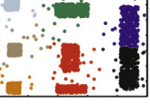
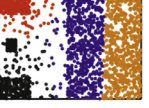

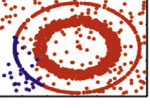
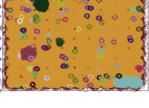
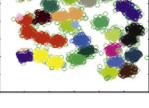

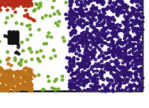

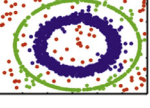
		DataSet : I N = 100000, C = 100	DataSet : II N = 3100, C = 31	DataSet : III N = 2377, C = 7	DataSet : IV N = 2220, C = 4	DataSet : V N = 2990, C = 3	DataSet : VI N = 1024, C = 2
Algorithm	K-Nets						
	K-means and/or K-centers						
	SPK-means	X					
	AP						
	Meanshift and/or Medoid shift						
	DBScan						

Fig. 5. K-Nets on artificial data sets. Each entry in the table illustrates the partition of a data set composed of N points into C clusters by one of the considered algorithms. Every cluster is represented by a color with the exceptions of K-means and SPK-means on (I), (II) where only some of the clusters are colored for visualization reasons. Indicatively, on the same computer, SPK-means partitioned (V) in 200 s while dimensionality restricted its application on (I). K-means requested 700 s (i.e. 50 replications) to partition (I). AP that had similar results with k-means in (III) and (IV) partitioned the two data sets into 100 s. K-Nets concluded all partitions into 15 s including the calculation of geodesic distances in (V, IV).

K-Nets implementation along with datasets used in this study can be downloaded from [7].

3.1. Experiments on synthetic datasets

In a first batch of experiments we tested our algorithm on several synthetic datasets with varying number, size and shape of clusters, as well as noise levels. Fig. 5 presents the performance of K-Nets in comparison to several well established clustering methods. As mentioned above, the number of clusters C in which K-Nets partitions a dataset composed of N samples can be either automatically detected (NOM) or defined (EOM).

In the first case, based on the value of the resolution parameter k , the algorithm recovers a fraction of the total number of all possible N partitions, which we term landmarks. For example, applying a single-layer K-Nets under Euclidean distance (i.e. unless otherwise stated the same metric will be used for all simulations) on the datasets of Fig. 5II to 5VI the number of landmarks for all different k values, was between 2.3 and 3.2% of all possible partitions.

This property mainly originates from the strict locality criteria imposed to K-Nets discrete parameter of density control. In other

words, the locality criteria implemented by providing as input a small number of nearest neighbors, drive the algorithm to recover under large degrees of resolution a small number of different partitions. As described in Methods section, K-Nets can detect and resolve data instabilities occurring mainly under large degrees of resolution, in contrast to algorithms like AP that present oscillations disturbing their otherwise deterministic nature [8].

For the dataset of Fig. 5II, for k values from $k=1$ to $k=10$, single-layer K-Nets provides ten different partitions with number of clusters C ranging from $C=3100$ to $C=202$ respectively (Fig. S1). All of the intermediate partitions, in terms of the number of clusters formulating them, can be recovered through EOM. It is reported that concerning the number of landmarks we had analogous results, for all datasets utilized in this paper.

As we are moving towards larger k values, that drive K-Nets to provide partitions from medium to small clustering resolution degrees, the difference between landmarks in terms of the number of clusters formulating them, is reduced or in some cases eliminated. On the dataset of Fig. 5II, for k values ranging from 30 to 100, K-Nets recovered partitions with number of clusters ranging from 54 to 21. Furthermore, the algorithm extracted the actual number of

clusters (i.e. 31) for k values ranging from 42 to 91, providing however for all of them the same set of exemplars as its final output. While in cases like the one at hand this behavior can be an indication of the true number of clusters, it can also introduce difficulties in the application of NOM for recovering a partition composed of an a-priori known number. In these cases, we can apply EOM. Specifically, the exact same partition composed of the 31 clusters (Fig. 5II), was determined under EOM for 40 different k values under which the number of clusters of the corresponding landmark partitions range from 55 to 15.

The isotropic nature of clusters in Fig. 5II lead K-Nets to detect the same partition irrespectively of whether EOM was initiated from a k value indicating a landmark composed of a number of clusters C_L smaller or larger than the requested one C_R . However, the latter case can be utilized to recover clusters of irregular size or shape like the cases of Fig. 5III and 5IV [7] that were detected for k values corresponding to landmark partitions with number of clusters ranging around $C_L = C_R + 1$ to $C_L = 2 \times C_R$.

It should be mentioned at this point that the main purpose of multi-layer K-Nets composed of one or more parallel layers is to facilitate partitioning in cases when the size of the similarity matrix does not allow the application of single-layer K-Nets or other clustering algorithms (such as the dataset in Fig. 5I). Furthermore, it can reduce the response time of K-Nets. For example, a single-layer K-Net partitioned the 7500 2-dimensional points arrangement of 50 isotropic clusters depicted in Fig. 4B into 3 s while the same dataset was partitioned through a two-layer K-Net (Fig. 4B) in 0.2 s. The k values provided as input to the specific network were $k=[15,6]$. The first parameter $k_1=15$ is utilized by every single-layer K-Net formulating the first parallel layer to partition some part of the input dataset into an a-priori unknown number of clusters. The resulting 329 exemplars of the first layer are provided as input to the second, and are partitioned under parameter $k_2=6$ –50 clusters. Utilizing NOM or EOM, we had the same results for k_1 values ranging from 2 to 35 under which K-Nets provided as input in the second layer a number of exemplars from 2875 to 136.

Simulations in datasets of different sizes have revealed that the actual values of the resolution parameter used on the first layer(s) are not so critical. The only restriction we have discovered is that the number of exemplars provided as input to the last layer must be about 3–7 times larger than the maximum number of clusters we want to partition a dataset into. Characteristically, we applied on the dataset depicted in Fig. 5IV a two-layer K-Net with resolution in the first layer ranging from $k=2$ to $k=50$. Under this range of values, in the last layer we had a number of exemplars outputted from the first layer, ranging from 848 to 28. In all cases K-Nets under EOM provided the same result with the single-layer K-Net we previously described. Furthermore, the same results were identified for both architectures depicted in Fig. 4. It is reported that analogous results we had for all datasets presented in this paper.

Another interesting application of the multi-layer architectures is in combination with similarity measures like geodesic distances. While their application can allow the detection of non-linearly separable clusters, their calculation, as mentioned, is an especial time consuming process hampered by problems like short-circuiting on noisy datasets. Hence, on the datasets Fig. 5V and 5VI, algorithms like K-centers, Medoidshift [18], and AP provided under Euclidean distance erroneous yet better results (Fig. 5) in comparison to the ones they had with geodesic distances (results not shown). In contrast, the data compression achieved in the first K-Nets layer(s) makes the performance of our method more robust in such cases, due to the fact that geodesic distances are used on exemplars of the last layer instead of actual data samples. Thus, a two-layer K-Nets (Fig. 4A) partitioned correctly the aforementioned cases, since the calculation of geodesic distances on the compressed dataset

of the second layer has significantly reduced noise levels. Finally, while SPK-means adequately partitioned the dataset in Fig. 5VI, it failed on the one of Fig. 5V, due to the K-means originating inability in detecting non-isotropic clusters (Fig. 5III, 5IV). Utilizing a method [19] from the vast literature on K-means initialization and improvement variants [20,21] did not alter its results in Fig. 5. K-Nets, through EOM partitioned this dataset correctly (see also Fig. S4, S6).

3.2. Experiments on real datasets

Initially, we tested the performance of K-Nets against the standard exemplar-based algorithm K-centers on real datasets with a wide range of clustering resolution degrees. To this end we utilized the single-layer K-Nets (Figs. 1 and 2) on a faces dataset composed of 2700 photographic images of 135 persons [7,22]. A 9000-dimensional vector represents each image, while a similarity matrix was constructed using the Euclidean distance and given as input to both algorithms. The same procedure was applied on a pen digit dataset, consisting of 10992 16-dimensional vectors each corresponding to digits in the range 0–9 [7,23]. This time the similarity matrix of the data set was given as input to a two-layer K-Nets (Fig. 4B). In both cases K-Nets NOM for various values of k was utilized and the K-centers methods was run to obtain the same number of clusters.

Fig. 6 displays the comparative results for faces (Fig. 6A and C) and pen digits (Fig. 6B and D), respectively. Partitioning efficiency was evaluated using the average squared error (ASE). As can be observed, K-Nets consistently achieves better partitioning in terms of ASE under all clustering resolutions (ranging from 20 to 60% for faces dataset and from 15 to 50% for pen-digits dataset—Fig. 6A and B, respectively). We further evaluated the partitions of the two algorithms on the real number of clusters, by employing the normalized mutual information (NMI) which is frequently adopted [23,24] when labeling of the data is known (i.e. a value of 1 indicates a perfect clustering result, while 0 the opposite). In both data sets the NMI value of K-Nets (i.e. 0.91 and 0.7) was improved by 35 and 40% respectively in comparison to the best corresponding values of K-centers selected out of 10,000 repetitive executions of the algorithm. It is also reported at this point that the best cases of K-centers results were unchanged by utilizing the initialization method proposed in [19] and maintaining the number of repetitive executions. Results in Fig. 6C and D exemplify the improved performance of K-Nets in comparison with K-centers, in terms of time complexity on the whole range of resolution degrees, for both faces and pen-digits datasets. As mentioned in the Methods section, the first two phases of K-Nets, allow the algorithm to conclude its overall operation in a single or very small number of iterations in the assignment phase and hence improve substantially its time complexity over other methods. As an illustration of this property, related results are displayed in Fig. 7A and B for the faces and pen-digits datasets, respectively. The final partitioning solutions, obtained after the number of iterations required to reach convergence, were only less than 2% improved in terms of ASE in comparison to the value of the first iteration. The same characteristic applies to K-Nets under EOM (Fig. 8).

Another advantage originating from the first two non-iterative phases of K-Nets is its stable operational behavior, irrespectively of data peculiarities. To test this, we applied K-Nets on a sparse text dataset (more than 98% of the pattern matrix entries equals zero), consisting of 3893 abstracts from 3 different document collections. Every pattern/abstract is represented by a 3303 dimensional vector [7,25]. As can be observed in Table 1, K-Nets performance is stable regardless of the sparsity present in this dataset, which can introduce significant delays in the operation of other algorithms, such as AP.

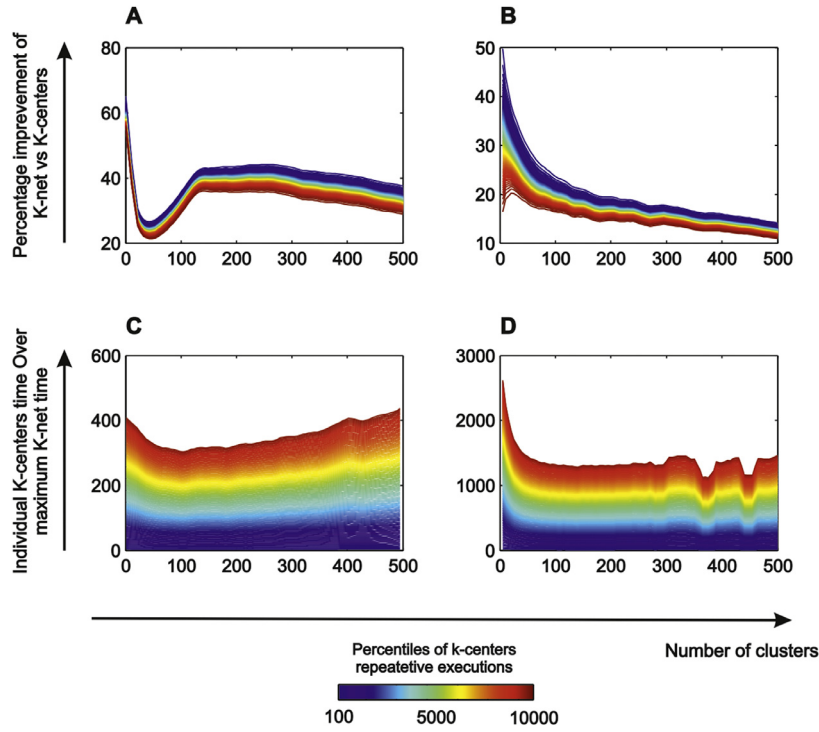


Fig. 6. Single and two-layer K-Nets. In (A), (B) we report results for the faces and pen-digits dataset respectively, the percentage improvement per cluster of K-Nets ASE value, in comparison to the corresponding best ASE value of k-centers, under a certain percentile of its repetitive executions. In (C), (D), we can depict, for the same data sets, the number of times that K-Nets most time consuming partition per data set, was faster than the time requested for each K-centers individual partition. Euclidean distance was employed as the similarity measure for both algorithms, while a single and a two-layer K-Nets was employed in the faces and pen-digits case.

Table 1

Results concerning the NMI values and time complexity for the algorithms of Fig. 5 that had the best performance (i.e. first column), on the text (I), subset of the MNIST (II), pen-digits (III) and USPS (IV) data sets. In all algorithms, a similarity matrix based on Euclidean distance was given as input except K-means (for the same algorithm we performed 10 replications and the best value was selected) while for dataset IV we utilized the sparse version of AP providing as input around 60% of the similarities. For coherency all results in this table are based on a two-layer K-Nets (Fig. 4A) without parallel layer. In the second and third row respectively we can see the results of the algorithm with geodesic and Euclidean similarities on the last layer. Providing as input geodesic distances to AP its performance declined in all datasets from 10 to 50% compared to the results of the table that correspond to Euclidean distances.

		Data set I: 3854 × 400, C = 5	Data set II: 3893 × 3303, C = 3	Data set III: 3498 × 16, C = 10	Data set IV: 7291 × 256, C = 10
Algorithms	K-Nets Geodesic	NMI = 0.9 t = 2 s	NMI = 1.0 t = 2 s	NMI = 0.8 t = 1.3 s	NMI = 0.77 t = 10 s
	K-Nets Euclidean	NMI = 0.58 t = 1.8 s	NMI = 0.91 t = 1.7 s	NMI = 0.72 t = 1.1 s	NMI = 0.59 t = 3.3 s
	SPK-Means	NMI = 0.58 t = 453 s	NMI = 1.0 t = 550 s	NMI = 0.7 t = 300 s	NMI = 0.68 t = 1800 s
	AP	NMI = 0.45 t = 270 s	NMI = 0.91 t = 390 s	NMI = 0.65 t = 140 s	NMI = 0.54 t = 800 s
	K-means	NMI = 0.61 t = 270 s	NMI = 1.0 t = 3000 s	NMI = 0.68 t = 5 s	NMI = 0.63 t = 140 s

Additionally, Table 1 presents comparative results on a down-sampled subset of MNIST dataset [726] composed of 3854 images, each one represented through a 400-dimensional vector corresponding to a handwritten digit in the range 5 to 9 (5 clusters). Furthermore, on the same table we provide results for the test subset of the pen-digits dataset composed 3498 samples [7] as well as the training part of the well known USPS dataset [27] composed of 7291 monochrome digits, each one defined through a 256 vector (i.e. 16×16 image). K-Nets partitions these datasets in a fraction of times needed by the compared methods, irrespective of the data dimensionality.

Next, we tested our algorithm on nonlinear measures of similarity like geodesic distances. Initially, we applied a two-layer K-Nets

(Fig. 3A) under EOM to partition the aforementioned MNIST subset. We provided as input a similarity matrix based on Euclidean distances, while geodesic distances were adopted in the second layer. The partition of K-Nets, resulted in a 30% improvement in terms of the NMI metric, in comparison to the corresponding partition of K-means (Table 1), which had the best NMI value among the other centroid or exemplar based algorithms considered. Similar results were obtained on the training part of the USPS dataset. K-Nets performance was improved by 15% in comparison to that of SPK-means, which had the next best performance. Methods such as AP and Medoidshift have difficulties in partitioning optimally the data set using geodesic distances due to the noise in the data. As highlighted also by the results on synthetic data sets in Fig. 5IV,

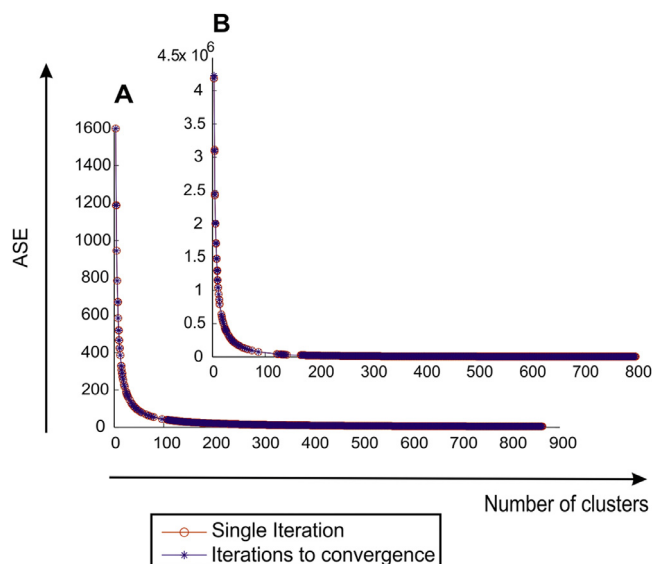


Fig. 7. K-Nets iterations under NOM. Performance of K-Nets in terms of ASE, over the number of clusters in the faces (A), and text (B), data sets. As can be depicted in both cases, the results of K-Nets utilizing a single iteration in the assignment phase were either the same or at most 2% worst than the one utilizing the full number of iterations until convergence.

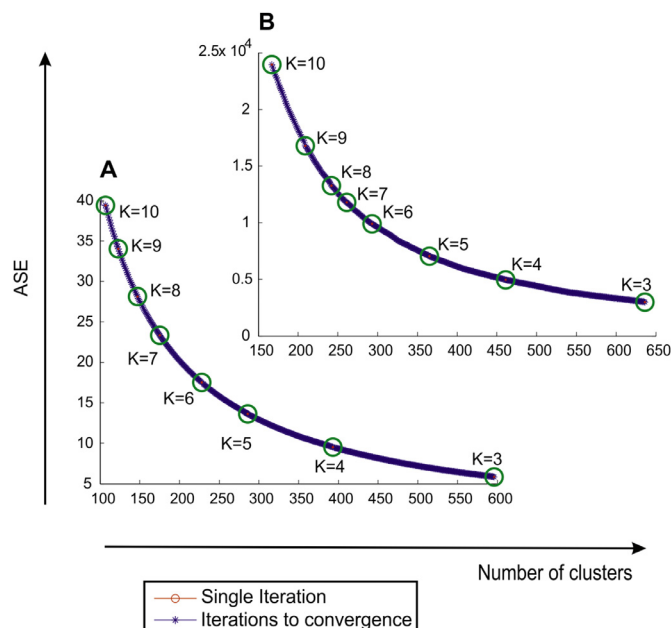


Fig. 8. K-Nets iterations under EOM. For all partitions in (A) and (B), corresponding to the faces and text datasets respectively, EOM was utilized, except the ones indicated with green circles that were extracted under NOM for a fixed value of k . Specifically, we provided as input in K-Nets the requested number of clusters C_R , and a k value for which the algorithm partitions the data set into a smaller number of clusters than C_R . The improvement, during Assignment phase, between 1 and full convergence iterations (i.e. up to 5) was up to 1%.

these methods provide better, even if sub-optimal, results using the Euclidean distance. It is reported that the parameters for the algorithms SPK-Means and AP were set following guidelines in [28, 29].

Simulations performed for further comparison on a variant of K-Means, `kmeans ++` [19] and for the same number of repetitive executions introduced no improvement to the results of K-Means.

Additionally, for the training parts of pen-digits and USPS datasets, K-Nets results (Table 1), compare favorably to the state-of-the-art multi-level AP approach reported in [28]. Worth noting, the respective approach implicates more than three clustering algorithms, pre-processing frameworks and the setting of several parameters. Furthermore, we tested K-Nets on the full pen-digits and USPS datasets composed from their corresponding training and test parts. For these datasets K-Nets had NMI values of 0.83 and 0.8 respectively. These results are competitive with another state-of-the-art clustering framework, that incorporates various methods like robust estimation techniques and linear least-square solvers to optimize criteria for its operation [29].

K-Nets, in contrast to many algorithms (as illustrated also by results presented in Fig. 5I for synthetic data), can be utilized to partition datasets in arbitrary degrees of clustering resolution and hence detect interesting signals which may be lost in partitions composed of a few over-sized clusters. To outline this property, we adapted a dataset based on an RNA Seq experiment studying the human transcriptome through the expression levels of 41208 protein and non-protein coding Ensembl-annotated genes, measured across 18 samples of different somatic tissues [30]. We applied a two-layer K-Nets under EOM setup to partition the dataset into 100, 200, 300, 400 and 500 clusters. Under large clustering resolution, evaluation indices like ASE disintegrate and cannot provide faithful results [8]. Hence, we adapted a simple score referred to as cluster discrimination score (CDS) [Table S4] as a performance measure for this experiment. The range of CDS is from 0 to 1, the larger its value is, the better is the discrimination capacity of the clustering algorithm. In this experiment the mean CDS value of K-

Nets over all clusters, was improved by 35% in comparison a sparse version of AP that had the next best performance from all considered algorithms.

In Fig. 9A we present the 10 top ranked in terms of CDS value clusters (i.e. more than 0.8) that fulfilled 2 additional criteria, which reflect the biological relevance of the retrieved clusters. The first one is that their members should have less than 2% overlap with neighboring genes, since in the opposite case their expression level estimates could be uncertain. The second one, relevant to the essence of the study in [30], is that more than 40% of its members should be orthologous genes. In contrast to K-Nets none of the aforementioned criteria were fulfilled by any of the algorithms of Table 1. We obtained similar results with microarrays experiments (Fig. S2, S3).

All simulations presented to this point utilized either a single or one of the two-layer architectures of Fig. 4. Nevertheless, it is a straightforward process to build a large repertoire of different K-Nets architectural schemas. As an example, we will utilize the K-Nets of Fig. 4B and A connected in series and apply it on the problem of image segmentation.

We utilized 4 RGB images each one of size $321 \times 481 \times 3$, transformed into the CIELAB color space and built the corresponding 4-dimensional matrices each one of size 154401×3 . We utilized the algorithms of Fig. 5 to partition the images into 2 clusters. Under this dimensionality we could not apply SPK-means, Medoidshift on the problem without some sort of external pre-processing method (Fig. S5). As can be depicted in Fig. 10, K-Nets correctly detected the main figure from its background in all four cases in contrast to K-means and Meanshift, which had the next best performances. Matching the partitions of the three algorithms, against a set of ground-truth segmentations, the average value over all four K-Nets partitions was improved by more than 45% in comparison to the

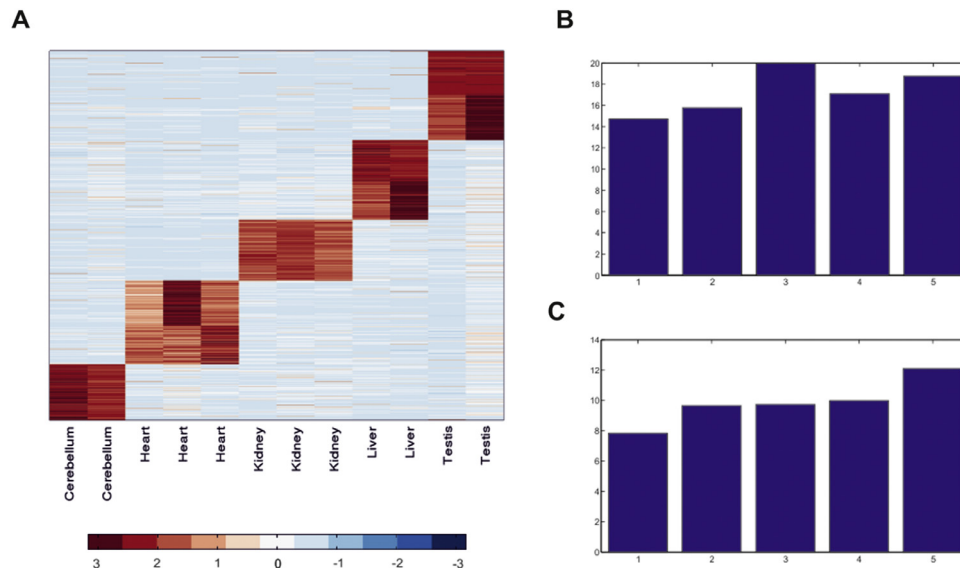


Fig. 9. K-Nets on RNA-seq dataset. In (A) we illustrate 10 of the most discriminative in terms of CDS value (i.e. larger than 0.8) and enriched in terms of orthologues genes clusters (i.e. more than 40% of their members) from the 200 K-Nets partition. A sparse version of AP with around 60% of the similarities that had the next best performance could not detect any clusters with these properties. Similar results we had for all partitions. In (B) and (C) we illustrate the percentage improvement of K-Nets against mean and maximum CDS values respectively (i.e. y-axis) over 100–500 clusters (i.e. x-axis) with a step of 100 in comparison to AP.



Fig. 10. Image segmentation. Segmentation results of 4 images (column 1) into 4 clusters utilizing K-Nets (column 2), k-means (column 3), and Meanshift (column 4). For every algorithm we illustrate the cluster that best captured the dominant foreground figure of the corresponding image. K-Nets, was applied under EOM [14].

corresponding values of K-means and Meanshift, in terms of the measures utilized in [31].

4. Conclusions and future work

In this paper we presented K-Nets, a novel deterministic, exemplar-based clustering algorithm that can be utilized to partition a dataset into a predefined or automatically derived number of clusters. K-Nets can be utilized to perform clustering without any

type of data selection, or pre-processing, even in cases when data type, dimensionality or resolution, render the application of other methods impractical or even impossible. Our results show that the algorithm can be utilized to detect the true underlying data structure in benchmark datasets in which well established methods fail.

For comparison with other methods, we have utilized a brute force attack to detect the nearest neighbors of every sample, while recent studies present significantly more efficient both in time and space complexity methods for their determination [32]. Further-

more, the first two phases of K-Nets that are mainly responsible for its fast response, could be further accelerated in applications like the segmentation of images [33] and protein-protein interactions graphs [34], that the nearest neighbors set of every sample is known in advance.

The range of the resolution parameter k , that is independent of the dataset (Table S1), leads to the predictability of the final number of exemplars extracted by K-Nets (Fig. S1), and hence significantly simplifies the construction of multi-layer architectures with large degree of complexity that would otherwise be difficult or prohibitive. While in this study for simplicity we have utilized, for the majority of the simulations, architectures with a single parallel layer we can have any number of them connected in series (Tables S2, S3). For future work, we are testing architectures that follow the same connectivity assembly and architectural principles with the ones presented here but the elementary structure utilized for their construction is a multi-layer K-Net (like the aforementioned one). This type of nested K-Nets schemas can be repeated any number of times depending on the data dimensionality and can further improve algorithm's performance on big-data problems.

Another direction of K-Nets improvement lies in the approach followed to consider exemplars in the assignment phase. For simplicity, in current K-Nets implementation, all members of a cluster are considered as possible exemplars in the assignment phase. However, it has been shown that the number of iterations in this phase is from 1 to 5 in the vast majority of the cases. Hence only a small number of samples near the neighborhood of the pre-exemplar of every cluster should be considered for its replacement. This could further improve the time complexity of K-Nets in cases when is applied on datasets composed of extremely large number of samples. Additionally, the simplicity of the proposed algorithm could enhance the adaptation of various methods that reduce the number of essential distance calculations amongst samples [35,36], or the incorporation of techniques like hashing or binary code learning [37].

Finally, the aforementioned K-Nets properties constitute the algorithm a highly promising candidate to be utilized in conjunction with both supervised (e.g. hidden layer of RBF networks) and unsupervised methods (kernel-based PCA variations) for the linearization of input space.

Acknowledgments

We would like to thank Maria Tsekoura for useful discussions, George Dimitrakopoulos for commenting on earlier versions of the manuscript, Alexander Polyzos for his help in building Hela/Namalwa dataset. This work was partially supported by the KMW offsets and Greek Secreteriat for Research and Technology Cooperations (EDGE and NoisePlus) and Excellence (Stochagen) grants to Dimitris Thanos.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:[10.1016/j.patcog.2018.11.010](https://doi.org/10.1016/j.patcog.2018.11.010).

Appendix A

In the following we provide a Matlab® listing of single-layer K-Network as described in Section 2.1 (without the check for data instabilities in Section 2.1.3). For a complete listing of single and multi-layer K-Networks please refer to <http://users.iit.demokritos.gr/~imaraziotis/knets> [7].

```
function idx = single_layer_knet(D, k, c)
% D: similarity matrix, k: integer resolution parameter (k > 0)
% c: (optional) exact number of clusters.
if margin < 3, c = 0; end; [SV, SI] = sort(D, 2); PEs = []; idx = zeros(1, size(D, 1));
for k = k:-1:1
    for i = 1:size(D, 1) % Construction Phase
        EDM = find(SV(i,:) <= SV(i, k)); PCs{i} = SI(i, 1:EDM(end));
        S(i) = (1/length(PCs{i})) * sum(SV(i, 1:k));
    end;
    % Selection Phase
    [~, si] = sort(S);
    for i = 1:size(D, 1)
        if sum(idx(PCs{si(i)})) == 0, PEs = [PEs si(i)]; idx(PCs{si(i)}) = 1; end
    end;
    if length(PEs) >= c, break; end
end
if c > 0, PEs = PEs(1:c); end; c = length(PEs);
% Assignment Phase.
[~, idx] = min(D(PEs, :)); idx = PEs(idx);
for iters = 1:10
    for i = 1:c
        M = find(idx == PEs(i)); [~, tmp] = min(mean(D(M, M))); PEs(i) = M(tmp);
    end; last = idx; [~, idx] = min(D(PEs, :)); idx = PEs(idx);
    if ~any(idx ~= last), break; end;
end
```

References

- [1] A.K. Jain, M.N. Murty, Data Clustering: A review, *ACM Comput. Surv. (CSUR)* 31 (3) (1999) 264–323.
- [2] R. Xu, D. Wunsch, Survey of clustering algorithms, *IEEE Trans. Neural Netw.* 16 (3) (2005) 645–678.
- [3] A.K. Jain, Data clustering: 50 years beyond K-means, *Pattern Recognit. Lett.* 31 (8) (2010) 651–665.
- [4] M. Muja, D.G. Lowe, Scalable nearest neighbor algorithms for high dimensional data, *Pattern Anal. Mach. Intell. (PAMI)* 36 (2014).
- [5] K.M. Kumar, A.R.M. Reddy, A fast DBSCAN clustering algorithm by accelerating neighbor searching using Groups method, *Pattern Recognit.* 58 (2016) 39–48.
- [6] Y.Q. Zhu, L.Y. Chang, D.W. Zhenghui, G.Y. Lia, A novel clustering method based on hybrid K-nearest-neighbor graph, *Pattern Recognit.* 74 (2018) 1–14.
- [7] Download codes and datasets from: <http://users.iit.demokritos.gr/~imaraziotis/knets/>
- [8] B.J. Frey, D. Dueck, Clustering by passing messages between data points, *Science* 315 (5814) (2007) 972–976.
- [9] J. MacQueen, in: *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, 1, 1967, pp. 281–297.
- [10] M. Ester, H.P. Kriegel, J. Sander, X. Xu, A density based algorithm for discovering clusters in large spatial databases with noise, *KDD'96 Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, 1996, pp. 226–231. KDD (34) Portland, Oregon.
- [11] C. Yizong, “Mean shift, mode seeking and clustering”, *IEEE Trans. Pattern Anal. Mach. Intell.* 17 (8) (1995) 790–799.
- [12] X. Zhang, W. Wang, K. Norvag, M. Sebag, “K-AP: generating specified K clusters by efficient affinity propagation”, in: *2010 IEEE International Conference on Data Mining*, 2010, December, pp. 1187–1192.
- [13] M. Leone, M. Weigt, “Clustering by soft-constraint affinity propagation: applications to gene-expression data”, *Bioinformatics* 23 (20) (2007) 2708–2715.
- [14] J.B. Tenenbaum, V. Silva, J.C. Langford, A global geometric framework for non-linear dimensionality reduction, *Science* 290 (5500) (2000) 2319–2323.
- [15] M. Balasubramanian, E.L. Schwartz, The isomap algorithm and topological stability, *Science* 295 (5552) (2002) 7.
- [16] U. Von Luxburg, “A tutorial on spectral clustering”, *Stat. Comput.* 17 (4) (2007) 395–416.
- [17] A. Ng, M. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, *Adv. Neural Inf. Process. Syst.* 14 (2002) 849–856.
- [18] Y.A. Sheikh, E.A. Khan, T. Kanade, “Mode-seeking by medoidshifts”, in: *2007 IEEE 11th International Conference on Computer Vision*, 2007, October, pp. 1–8. IEEE.
- [19] D. Arthur, S. Vassilvitskii, k-means++: the advantages of careful seeding, in: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2007, pp. 1027–1035.
- [20] D.G. Mraquez, A. Otero, C.A. Garcia P. Felix, A novel and simple strategy for evolving prototype based clustering, *Pattern Recognit.* 82 (2018) 16–30.
- [21] H. Ismikhani, l-k-means++: An iterative clustering algorithm based on an enhanced version of the k-means, *Pattern Recognit.* 79 (2018) 402–413.
- [22] Internet address for downloading the faces dataset: <http://cswww.essex.ac.uk/mv/allfaces/faces94.html>.
- [23] I.S. Dhillon, Y. Guan, B. Kulis, in: *Proceedings of the 10th ACM SIGKDD*, 2004, pp. 551–556.
- [24] I.A. Maraziotis, A semi-supervised fuzzy clustering applied to gene expression data, *Pattern Recognit.* 45 (1) (2012) 637–648.
- [25] I.S. Dhillon, D.S. Modha, Concept decompositions for large sparse text data using clustering, *Mach. Learn.* 42 (1–2) (2001) 143–175.

- [26] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner, Gradient based learning applied to document recognition, *Proc. IEEE* 86 (11) (1998) 2278–2324.
- [27] Y. LeCun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, Backpropagation applied to handwritten zip code recognition, *Neural Comput.* 1 (4) (1989) 541–551 December 1989.
- [28] F. Shang, L.C. Jiao, J. Shi, F. Wang, M. Gong, Fast affinity propagation: A multi-level approach, *Pattern Recognit.* 45 (1) (2012) 474–486.
- [29] S.A. Shah, V. Koltun, Robust continuous clustering, *PNAS* 114 (37) (2017) 9814–9819.
- [30] D. Brawant, The evolution of gene expression levels in mammalian organs, *Nature* 478 (2011) 343–348.
- [31] S. Belongie, J. Malik, J. Puzicha, Shape matching and object recognition using shape contexts, *IEEE Trans. Pattern Anal. Mach. Intell.* 24 (4) (2002) 509–522.
- [32] A. Andoni, A. Naor, A. Nikolov, I. Razenshteyn, E. Waingarten, Holder homeomorphisms and approximate nearest neighbors, *IEEE 59th Annual Symposium on Foundations of Computer Science*, 2018.
- [33] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Susstrunk, SLIC superpixels compared to state-of-the-art superpixel methods, *PAMI* 34 (11) (2012) 2274–2281.
- [34] I.A. Maraziotis, K. Dimitrakopoulou, A. Bezerianos, Growing functional modules from a seed protein via integration of protein interaction and gene expression data, *BMC Bioinformatics* 8 (2007) 408.
- [35] C. Elkan, Using the triangle inequality to accelerate K-means, in: *Proceedings of the Twentieth International Conference on International Conference on Machine Learning*, 2003, pp. 147–153.
- [36] Y. Chen, S. Tang, N. Bouguila, C. Wang, J. Du, H. Li, A fast clustering algorithm based on pruning unnecessary distance computations in DBSCAN for high-dimensional data, *Pattern Recognit.* (83) (2018) 375–387.
- [37] X. Shen, W. Liu, I. Tsang, F. Shen, Q.S. Sun, Compressed K-means for large-scale clustering, in: *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)*, 2017.