Comput Syst Sci & Eng (2020) 6: 441–456

International Journal of
Computer Systems
Science & Engineering

Human Activity Recognition Based on Parallel Approximation Kernel K-Means Algorithm

Ahmed A. M. Jamel[1,*] **and Bahriye Akay**[2,†]

[1] *Erciyes University, Institute of Natural and Applied Sciences, Department of Computer Engineering, 38039, Melikgazi, Kayseri, Turkey*
[2] *Erciyes University, Engineering Faculty, Department of Computer Engineering, 38039, Melikgazi, Kayseri, Turkey*

Recently, owing to the capability of mobile and wearable devices to sense daily human activity, human activity recognition (HAR) datasets have become a large-scale data resource. Due to the heterogeneity and nonlinearly separable nature of the data recorded by these sensors, the datasets generated require special techniques to accurately predict human activity and mitigate the considerable heterogeneity. Consequently, classic clustering algorithms do not work well with these data. Hence, kernelization, which converts the data into a new feature vector representation, is performed on nonlinearly separable data. This study aims to present a robust method to perform HAR data clustering to mitigate heterogeneity in data with minimal resource consumption. Therefore, we propose a parallel approximated clustering approach to handle the computational cost of big data by addressing noise, heterogeneity, and nonlinearity in data using data reduction, filtering, and approximated clustering methods on parallel computing environments that have not been previously addressed. Our key contribution is to treat HAR as big data implemented by approximation kernel K-means approaches and fill the gap between the HAR clustering cost and parallel computing fields. We implemented our approach on Google cloud on a parallel spark cluster, which helped us to process large-scale HAR data across multiple machines of clusters. The normalized mutual information is used as validation metric to assess the quality of the clustering algorithm. Additionally, the precision, recall, f-score metrics values are obtained somehow to compare the results with a classification technique. The experimental results of our clustering approach prove its effectiveness compared with a classification technique and can efficiently detect physical activity and mitigate the heterogeneity of the datasets.

Keywords: Activity recognition, approximation approach, clustering, parallel computing, sampling

1. INTRODUCTION

With the growth of data in various fields from gigabytes to terabytes and even larger, new and promising data mining techniques and approaches are required to extract relevant information from these data. Because smart devices and wearable devices are integrated with various accelerometers, gyroscopes and GPS sensors, human activity data sets produced by multiple data sources have become large-scale. The data collected from the sensors are used to monitor a person's sports activities and health situations. Moreover, human activity recognition (HAR) uses the data to automatically recognize the daily activities of a person, such as sitting, walking, stair up, stair down, running, biking, and so on. Although the studies on HAR started in the 1980s [1], HAR only gained significant attention with the development of sensors, wearable devices, and smartphones. However, the differences in smart device models from numerous manufacturers results in a considerable heterogeneity in the collected data. This heterogeneity is due to several specific reasons, such as differences in sampling frequency, operating system, and central processing unit (CPU) load condition, thus it causes difficulty in processing the data.

For HAR purposes, time, frequency, and statistical domain features are used for dimensionality reduction and to easily manipulate data [2, 3]. Image-based methods in which the user is monitored with a camera in real-time have also been used for activity recognition [4, 5]. However, this method is not ideal due

*email: 4010941311@erciyes.edu.tr
†email: bahriye@erciyes.edu.tr

to portability and user privacy concerns. Simple sensors reduce the disadvantages of using cameras for activity recognition.

The HAR problem in which daily activities are categorized into groups, such as sitting, walking, stair up, stair down, running, biking, etc., can be considered as a clustering problem. Thus, clustering approaches, which have been successfully applied in pattern recognition, machine learning, image segmentation, text mining, computer graphics, and learning theory [6, 7] can be utilized to recognize human activity.

Many types of research focus on detect human activity with data mining and machine learning techniques. Classification algorithms such as Support vector machine (SVM), Decision tree, K nearest neighbor (KNN), Random forest and naïve Bayes have been widely used in HAR field [8–18]

Additionally, various versions of deep learning and neural network algorithms are also implemented in this field [19–21]. As can be seen, most of the research focuses on solving the HAR problem as a classification task where ground truth data (target class data) must exist in the classifier training data, which are not always available. Thus, there is a limited number of studies on HAR using clustering algorithms [2, 3, 22].

The clustering of HAR data is a challenging task for the following reasons [2, 8, 23–25].

- HAR data is noisy and requires digital signal processing operations before clustering
- HAR data is big and its clustering requires huge computational resources.
- HAR data suffer from heterogeneity, which makes linear cluster separation almost impossible. Nonlinear clustering techniques also require much more computational resources.

Further, implementing HAR in parallel manner is ignored in most research. Therefore, we noticed the technology gap between the two disciplines; big data clustering and HAR that needs to be filled. To that end, we proposed a parallel implementation of approximate kernel k-means to solve the problem of HAR data clustering with minimum computational resources and accuracy as high as the one obtained from classification methods.

We used a kernelization approach in this work. The approach involves kernel functions converting data into a new feature vector representation to exploit more information and accomplish better clustering in nonlinearly separable data. While these functions allow the data to be manipulated in a high-dimensional feature space, the high dimensionality of the kernel matrix still remains a significant bottleneck of the kernel K-means algorithm as it consumes high resources such as memory and CPU cores.

To overcome this problem, the algorithm of approximate kernel K-means is proposed, which uses only a small part of the kernel matrix for implementation. The data proportions are extracted systematically based on multistage sampling methods by a combination of stratified sampling and simple random sampling (SRS) as multistage approaches. Stratified sampling is a sampling method that divides data into groups that share the same properties. These groups are called stratas [26]. SRS selects samples of data randomly. First, we performed the first stage of stratified sampling before low-pass filtration is applied to filtering to reduce noise. Next, the second stage of stratified sampling is applied, and the data is subsequently normalized. The principal component analysis (PCA) is used [27–29] to avoid the correlation and overlap of the data. Further, the whole population of the dataset is divided into a small and big proportion with the SRS method. In the small proportion, we use kernelization to fit the nonlinearly separable dataset as the basic K-means algorithm does not work well with nonlinearly separable datasets [30]. The big proportion is approximated using the Euclidean distance and KNN classifier based on the small proportion's centroids.

It is not suitable to employ personal computers for implementing the algorithms specially designed for big data as they require a long time for processing. Therefore, parallel algorithms were developed to save time on massive, complex problems that are difficult to solve with limited computer memory. Parallel algorithms can exploit cluster resources on the Wide Area Network (WAN) or cloud, which have massive data storage and allow for quick data computations. Several studies have been published regarding the implementation of machine learning algorithms in parallel environments. In [31–37], parallel K-means clustering algorithms on multicore systems such as CPU and graphics processing unit (GPU) were implemented. In these studies, data was divided into small chunks in which each chunk was processed by one core simultaneously. Further, studies [38–43] accomplished parallel K-means algorithm on apache Hadoop, MapReduce, and Spark environments. The Spark standalone machine uses a simple first in first out (FIFO) scheduler for applications, in which each application uses all the available nodes in the cluster; thus, the number of nodes can be limited per application. Hence, in this work we clustered big HAR datasets on the cloud by Spark framework. We implemented our parallel approach on Google cloud [44] as cloud computing is advantageous for low cost, accessibility, flexibility, and reliability. Apache Spark is an open-source library that permits parallel processing on large-scale data across multiple machines of clusters and saves processing time.

To summarize, the contributions of this paper are shown as follows:

- Unlike most HAR problems that dealt with classification tasks, we attempt to identify human activities by using approximation kernel K-means algorithm which is not addressed with HAR before.
- Dealing with human activity recognition datasets as big data.
- Clustering human activity recognition data in a parallel environment by using Spark on google cloud.

The remaining part of this paper is organized as follows. Section 2 provides general information on the kernel K-means algorithm. Section 3 presents an overview of related works while Section 4 presents the methods that have been used to preprocess the data, sampling, and clustering approaches. Section 5 provides the experimental setup and dataset while Section 6 presents the results and data evaluation. Section 7 provides the discussions, and the last section presents the conclusion.

Table 1 Kernel functions and their formula

Kernel function	Equation
Polynomial	$K(x_i, x_j) = (x_i^T x_j + \gamma)^{\delta}$
RBF Gaussian	$K(x_i, x_j) = \exp(-x_i - x_j^2/2\sigma^2)$
Sigmoid	$K(x_i, x_j) = \tanh(\gamma(x_i^T x_j) + \theta)$

2. KERNEL K-MEAN

The kernel K-means algorithm depends on kernelization to achieve more accurate clustering capabilities for detecting nonlinearly separable clusters. In kernel K-means, the data points are projected from the input space into a higher-dimensional kernel space through a nonlinear transformation and then K-means is applied in the feature space [30, 45]. The computational complexity of kernel K-means is higher than K-means as kernel K-means computes the data as the $n \times n$ kernel matrix where n represents the number of records. The popular kernel functions are polynomial, Gaussian, RBF Gaussian, and Sigmoid functions (Table 1). This work focuses on the Gaussian kernel function.

Assuming the data points are to be clustered into C clusters, and each cluster has center w in the higher dimensionality space, as each data point is assigned to a cluster, the C cluster center is calculated using (Equation 1) [30, 45].

$$w = \frac{\sum_{i \in C} \phi x_i}{C} \tag{1}$$

The squared distance $Dis(x_i, w) = ||\phi(x_i) - w||^2$ between the vector x and w can be written as (Equation 2) [30, 45].

$$\begin{aligned} \|\phi(x_i) - w\|^2 &= \phi(x_i)\phi(x_i) - 2\frac{\sum_{j \in C} \phi(x_i)\phi(x_j)}{C} \\ &\quad + \frac{\sum_{i \in C}\sum_{j \in C} \phi(x_i)\phi(x_j)}{C^2} \\ &= k(x_i, x_i) - 2\frac{\sum_{j \in C} k(x_i, x_j)}{C} \\ &\quad + \frac{\sum_{j \in C}\sum_{j \in C} k(x_i, x_j)}{C^2} \end{aligned} \tag{2}$$

After measuring the distance between the data points and cluster centers, the data sample is reassigned with the minimum distance $Dis(x_i, w)$ to the cluster C. This is an iterative process in which the distances are calculated, and the cluster assignments are updated until the cluster entry assignments are no longer changed or the maximum number of iterations is reached.

3. RELATED WORK

HAR applications can be used in various fields such as health services, smart environments, and security. In the health sector, HAR is used for activity tracking and monitoring for aging people [46, 47]. To detect the activities, most studies related to HAR applications employed classification algorithms, and a few of the studies employed clustering algorithms, Hence, this present study employed the kernel K-means clustering algorithm to recognize six activities of nine users.

To monitor and track the activities of patients prescribed by doctors, whether the patients are committed, Zhao et al. [22] presented an embedded model (TransEMDT) in mobile phones that integrates classification (decision tree) and clustering (K-means) algorithms. TransEMDT aims to solve the problem of obtaining inaccurate results when the phone is used by a different user. The model utilizes unlabeled examples to adjust the activity recognition model and build a customized model for the new user.

Albert et al. [46] performed an experiment on eighteen healthy objects and eight patients diagnosed with Parkinson's disease (PD). They predicted the activities (standing, walking, sitting, holding, or not wearing the phone) by utilizing SVM and sparse multinomial logistic regression (SMLR) algorithms. They reported the accuracy measurement for both proposed algorithms. The SVM reached 96.1% for healthy subjects and 92.2% for PD patients. Regarding the SMLR, the results reached 89.7% for healthy subjects and 84.7% for PD patients.

HAR is utilized in several applications, such as those that estimate a falling accident and measure calorie consumption and energy utilization [48]. These applications require the accelerometer to work consistently so as to continuously perceive distinctive physical human activities. This results in superfluous power utilization by the sensor as well as computational overhead, which is viewed as a major problem considering the constrained power assets of smartphones. Luštrek and Kaluža [49] investigated fall detection and activity recognition approaches by employing machine learning algorithms. Three attributes were selected to describe a user's behavior: locations of body parts in the reference coordinate system, body coordinate system, and angles between adjacent body parts. Eight machine learning algorithms were applied and compared. The authors reported that the highest accuracy was achieved at approximately 95% with SVM. Regarding power consumption, Vo et al. [50] expected to reduce the energy utilization of the accelerometer and CPU by enhancing the HAR algorithm. The proposed method depended on the integration of SVM classification and k-medoids clustering algorithms as a personalization algorithm to choose confident samples. An 11% increase in accuracy was achieved compared to the non-personalization methods.

Another study field of HAR applications is the position and orientation problem of smartphones [51]. The authors designed an approach to overcome the problem. The approach consists of two phases. First, the position recognition was performed using gyroscope data. Second, the activity recognition was performed using accelerometer data. SVM algorithm was used in both phases to recognize positions and four activities (run, walk, down-stairs, up-stairs). The authors reported that the accurate measurement of the overall classification without position recognition was higher than the accurate measurement of overall classification with position recognition. However, the

proposed two-phase approach has stable performance in overall activities.

One application of activity recognition is fall detection. Generally, as a result of the rapid aging of people, efforts must be made to ensure that elderly people can live longer with minimal support. To follow elderly home activities, Xu et al. [13] proposed a two-stage method for recognizing elderly home activity based on random forest and activity similarity. In a similar study Kumar and Bhavani [18] performed activity recognition for elderly people and disabled patients based on a combination of SVM and KNN algorithms. The experimental results concluded that the proposed work achieved superior results than other classifier algorithms.

Borazio and Laerhoven [52] performed the activity detection process and classification of eleven activities on both accelerometer (sensor) and time-activity survey datasets using the SVM algorithm. In this study, according to accuracy measurement, 25%, 18%, and 6% were obtained for activities such as eating, socializing, and hobbies, respectively, which was obtained using sensor data only. However, the positive effect of using time-activity survey data varies with activity and has been shown to have a positive effect on accuracy only for some activities.

Li et al. [14] presented human motion and position recognitions based on Wi-Fi-signals. The signals generated from Wi-Fi devices by utilizing the Discrete Wavelet Transform (DWT) technique. Then the signals have been classified based on the SVM algorithm to identify five activities (bend, halve, squat, step, stretch leg, and jump). They reported that the proposed approach obtained robustness performance. In a similar study, Feng et al. [15] presented a three-phase system Wi-multi technique. They aimed to recognize multiple human activities in a wireless environment. After the feature extraction process, the activities recognized by using the SVM algorithm. The experimental results proved that the proposed approach achieved 91.6% accuracy.

Manzi et al. [53], reviewed various state of art methods and described each of them as a literature survey. They reviewed HAR based on machine learning techniques including supervised techniques like KNN, SVM, decision trees, and later presented deep neural network techniques like artificial neural networks, convolutional neural networks, and recurrent neural networks. In a similar survey, Liu et al. [54] presented a review study on human activity with wireless signals. In this study, the authors reviewed the studies on HAR based on classification, clustering, and neural network techniques. We noticed the limited number of studies reviewed regarding clustering techniques.

Some studies have focused on recognizing activities by employing the random forest classifier in various applications [55–57]. The random forest classifier generates a set of decision trees from a randomly selected subset of the training set. The final decision of the test object is then aggregated. Gjoreski and Gams [55] showed the importance of activity recognition data preparation and applied low and high pass filters. They performed an attribute finding process to describe the user's behavior. The random forest classification algorithm was carried out on the data, and an accuracy greater than 93% was achieved based on the f-score performance metric. Xu et al. [56] proposed an approach to avoid the privacy and light conditions of vision-based activity recognition by employing a single wearable device sensor. A random forest classification algorithm was applied to identify six activities (walking, standing, stair-down, stair-up, jumping, and running). They achieved an average accuracy of 90% for walking, stair-up, and standing; 80% for running and stair-down; and 75% for jumping recognitions. Further, Ar and Akgul [57] presented a novel approach of extracting motion and pose features from image sequences (videos). The two features were combined using statistical methods that describe the global motion and global pose information in the entire image sequence. The authors used two datasets called KTH and Weizmann. Thereafter, the random forest algorithm was implemented to recognize the activities. They reported that the combination approach and classification process achieved good accuracies.

Ignatov and Strijov [58] proposed a technique for human activity recognition by utilizing time series data gathered from a single triaxial accelerometer of a smartphone. Time series segmentation and noise reduction were performed as data preparation. Thereafter, the KNN classification algorithm was applied to classify the obtained segments. The technique accomplished above 96% accuracy.

Some studies have focused on HAR by employing neural network techniques. Wang et al. [19] accomplished a novel method for the identification of human activity based on attention to process the data of the weakly labeled behavior by utilizing the Convolutional Neural Network (CNN) technique. The experimental results showed a high performance in terms of labeling accuracy. In a similar study Zhu et al. [20] proposed an ensemble model based on CNN technique to recognize human activity using nine-axis motion signals of gyroscope, accelerometer, and the magnetometer in smartphones. They reported that their work achieved 96.11% in terms of accuracy.

In our study, we treated the kernel K-means on HAR as a clustering task, and studied the implementation of kernel K-means in other fields. It was seen that the kernel K-means has not been implemented on HAR before. Thus, we filled the gap between the two fields by implementing kernel K-means clustering on HAR data based on Google cloud in a parallel manner.

The most common problem in the K-means algorithm is cluster center initialization, and several studies have attempted to solve it. Tzortzis and Likas [59] proposed a method to solve the problem by avoiding the high sensitivity of K-means, and they discussed the usage of the kernel step in the K-means algorithm. Kernel K-means is a technique that uses a kernel trick to enhance the accuracy of the K-means algorithm for clustering nonlinearly separated data. Several works have used and discussed kernel techniques, and significant ideas are detailed in [60]. Several approaches have been employed to improve the accuracy and time complexly of kernel K-means. Particularly, Chitta et al. [25] accomplished an efficient approximate kernel K-means algorithm by reducing computational complexity and the memory of the full kernel matrix K-means by a randomized approach. To avoid computing a full kernel matrix, they restricted the solution for the cluster centers to a smaller subspace spanned by a set of randomly sampled data points. The authors compared between proposed work with full kernel matrix K-means and two-step kernel K-means. According to speedup,

Adjusted Rand Index (ARI), error reduction, and NMI metrics, the proposed algorithm showed its efficiency compared to full kernel K-means. Additionally, Tsapanos et al. [61] optimized kernel matrix computations for largescale data clustering. The fast computation was performed by the exceedingly optimized library of linear algebra named basic linear algebra subprograms (BLAS), which can be very fast on modern CPU architecture. Kong and Kong [62] proposed a fast and effective version of the kernel K-means clustering algorithm. They used a technique named conditionally positive definition (CPD) kernel in the state of the mercer kernel function. They performed experiments on artificial and real datasets and reported that the proposed method converged and was faster compared to K-means algorithm.

In another work of Tzortzis and Likas [63], they accomplished a fast-implementation version of kernel K-means, named global kernel K-means, by locating near-optimal solutions, which enables the recognition of nonlinearity separable clusters in the input space. Wang et al. [64] employed an approach of kernel K-means to avoid its local minima domination called conscience online learning (COLL). This approach chooses the wining paradigm based on the conscience mechanism. They used several datasets in a video clustering dataset and reported that the proposed work gives superior results compared to other kernel clustering methods.

Zhang and Rudnicky [65] presented an implementation of the kernel K-means algorithm on a large-scale data. To overcome the expensive computations and storage cost, they implemented a new clustering technique by changing the clustering order from a sequence of samples to the sequence of kernels, which is an efficient method of kernel matrix calculations utilizing a disk-based strategy. The authors reported that the running times of the proposed method were better than those of the K-means algorithm, which was implemented on both artificial and real data.

Regarding the parallelization kernel K-means algorithm on multicore CPU and GPU, Baydoun at el. [66] implemented parallel kernel K-means on multicore CPU and GPU by utilizing OpenMP, Cilk Plus, and Nvidia CUDA. They reported that kernel K-means is more accurate than the traditional K-means, and a GPU implementation is faster than a multicore CPU implementation. Baydoun et al. [30] optimized and extended to the previewed work. They implemented a parallel kernel K-means algorithm on CPU and GPU. Further, they optimized CUDA-based code and addressed the impact of pattern, number of clusters, and features on computational time. The method was tested on artificial and real datasets. The authors achieved the best results in the CUDA implementation.

Concerning the parallelization kernel K-means algorithm on distributed frameworks, Tsapanos et al. [45] proposed a technical implementation of the kernel K-means algorithm on large datasets. Different clustering frameworks were tested, such as trimmed kernel K-means (TKKM), efficient TKKM, and approximate kernel K-means (AKKM). The authors focused on reducing computation time and computing kernel matrix techniques and such as full/partial fast and slow kernel matrix techniques. The MapReduce framework is used based on Apache Spark. The authors reported that implementing partial fast kernel matrix plus AKKM is more suitable for large-scale datasets. However, where the running time is not an issue, full slow kernel matrix plus TKKM is preferred. Another work of the same authors [67] presented an implementation based on the Spark cluster computing MapReduce distributed framework for the particular version of kernel K-means and nearest neighbor algorithm in which kernel matrix is managed as a full graph.

In summary, according to the papers reviewed, we concluded that most of the research focuses on human activity recognition as a classification problem. Many classification techniques have been used like SVM, KNN, random forest, and decision tree. Besides, neural network techniques have been implemented in this filed. As can be seen, the papers worked with HAR as a classification task that has not addressed the clustering task. Additionally, they have not manipulated with HAR as big data. Further, the parallel implementation of HAR also has not addressed in recent papers. Moreover, we observed that the papers implemented kernel K-means algorithm have not addressed the HAR field. Therefore, we filled the gaps in the mentioned problems by combining all these fields. Thus, in this paper, we proposed a parallel implementation of HAR data based on the approximation kernel K-means algorithm.

4. PROPOSED METHOD: APPROXIMATE PARALLEL KERNEL K-MEANS

Due to the high heterogeneity and nonlinearity of HAR datasets caused by various sensors of the smartphones and smartwatches, our approach was to employ kernelization, which transforms the data into a high-dimensional feature space based on an $n \times n$ kernel matrix. However, kernelization increases the processing time quadratically. Hence, when it is applied to a large-scale dataset, converting data into high-dimensionally feature space becomes considerably difficult. Therefore, we propose an approximate kernel K-means to cluster big HAR datasets. The main idea of the approximation approach is to represent only a small part of the data on the kernel matrix. The block diagram of the proposed approximate method is shown in Figure 1. We first applied some preprocessing operations on the data and then performed stratified sampling based on device feature.

The stratified sampling, which is an unbiased sampling technique, represents a dataset by its proportion created based on common property, and samples are randomly gathered from each subgroup [68]. The population of size N is divided into k subpopulations of sizes $N_1, N_2, N_3, ...N_k$. When we have a sample of n elements from the initial population, a sample of n_i in which $n_1 + n_2 + n_3 + ... + n_k = n$ is selected. The most important two terms in a sample are sampling and elevation factors. Sampling factor is the ratio between the size of the selected sample and the whole population and is represented as $\frac{n}{N}$. To obtain the percentage of the population represented in the sample, we multiply the sampling factor by 100. The elevation factor is the ratio between the size of the entire population and the size of the selected sample and is represented as $\frac{N}{n}$. This ratio represents the number of elements existing in the entire population for each element of the selected sample.

In our approach, four stages of sampling techniques are performed. In the first and second stages, stratified sampling is performed based on devices and users, respectively. Additionally, SRS is performed to divide the whole dataset into 90%

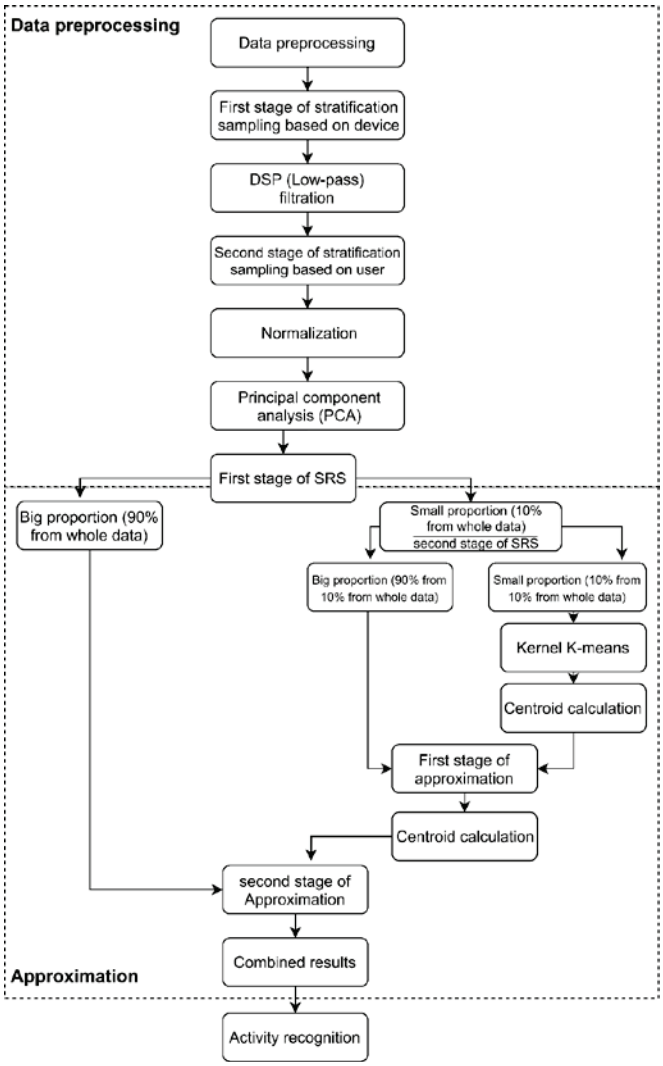


Figure 1 Proposed work flowchart.

and 10% proportions from data as the first stage of SRS. From here on out, the 90% and 10% proportions from data will be referred to as big and small proportions respectively. The small proportion from whole data is divided into another big 90% and small 10% proportions as the second stage of SRS. As can be seen in Figure 1, the proposed work starts with preprocessing steps involving cleaning, visualizing, getting data information. Then, the first stage of stratified sampling is applied based on devices. Thereafter, filtration, the second stage of the stratified sampling based on users, normalization, and PCA is applied sequentially. The small proportion of second stage of SRS is then clustered by kernel K-means. The clustering method lies in calculating the kernel matrix for each user separately. That is, each kernel matrix for each user is calculated once, where the distance between points will be called in the cluster assignment step. Hence, the calculation time is expected to decrease significantly. The kernel matrix calculation step is applied in parallel on a cloud platform. Subsequently, a new cluster centroid and cluster assignment is calculated for each cluster.

The approximation part of Figure 1 illustrates the approximation approaches used in this paper in (Figure 2 and Figure 3). The big proportion of the second stage of SRS is approximate based on Euclidean distance and KNN classifier. In the first approach of approximation with Euclidean distance. After the small proportion of the data being clustered, we calculate the cluster centers by taking the average of all data points of each cluster. Then, the big proportion of the data is approximated to be assigned to each cluster according to the shortest distance between the data points and cluster centers using Euclidean distance calculations (Figure 2).

In the second approximation approach with the KNN classifier, we were inspired by the idea presented in [69, 70, 71]. The applications related to data generated by sensors, especially HAR data, require costly labeling process. Therefore, the generation of clusters from a small amount of data by clustering algorithm presents a fast, simple, and accurate labeling process for the rest of the data by a classification algorithm, in which the classification model is built over the past labeled data with labels provided by the clustering algorithm. The KNN model is built by setting the small proportion as input and the clustered data by kernel K-means as output. Then, the big proportion of the data is approximated by the KNN model (Figure 3). This process continues until the first big proportion being approximated. Thus, the entire data points being clustered. The approximation approach enables the clustering of large-scale data in a shorter processing time and overcomes the kernel function calculations on all data.

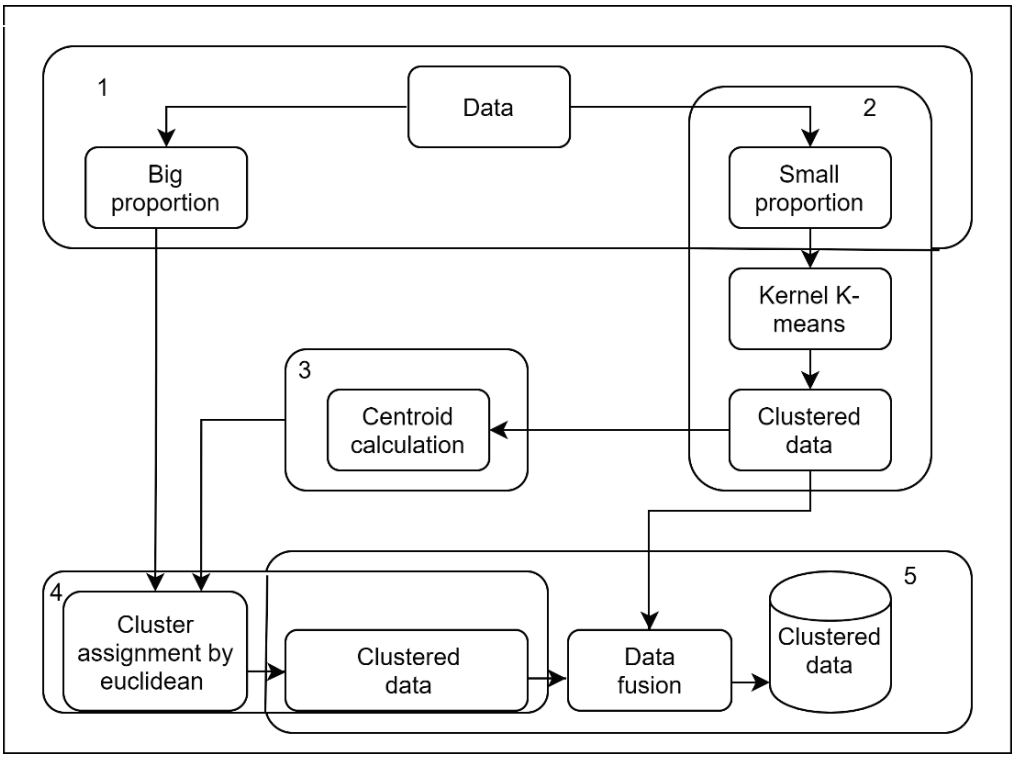


Figure 2 Approximation with Euclidean distance.

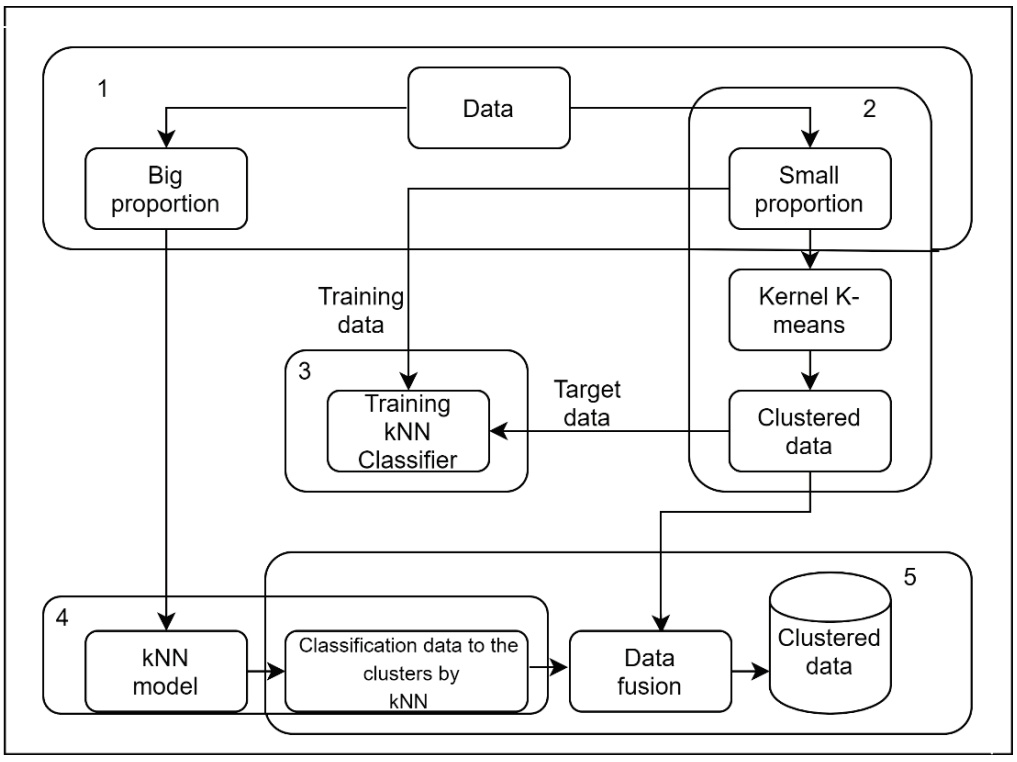


Figure 3 Approximation with KNN classifier.

5. EXPERIMENTAL SETUP AND DATASET

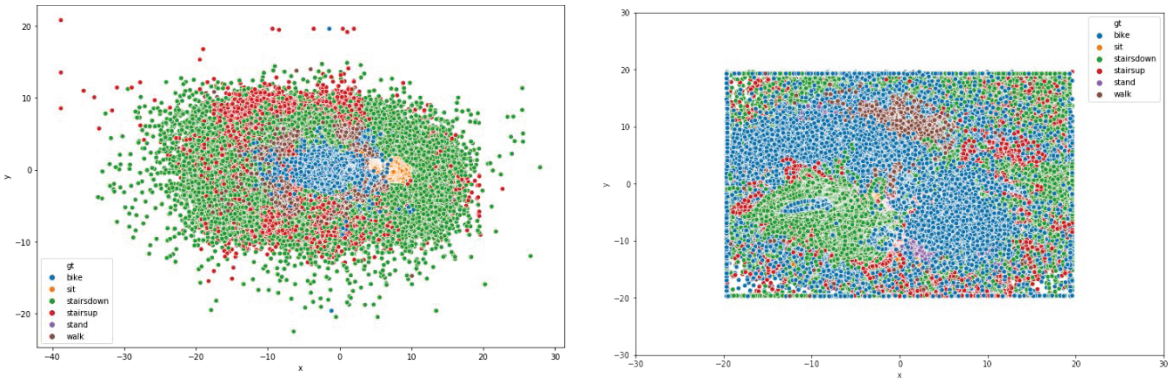
In this study, all the implementations were conducted using Python version 3.7.6, and Spark was used as the parallel programming environment. Pyspark API helped us to construct a relationship between Spark and Python kernel.

5.1 Datasets

In this study, we verified our work with two publicly available smartphone and smartwatch accelerometer datasets mentioned in the heterogeneity of human activity recognition study [72]. The raw data of phone and watch accelerometer datasets

Table 2 The devices and their corresponding sampling frequency.

Devices	Sampling frequencies (Hz)
Smartphones	
Nexus 4	200
Samsung Galaxy S3	150
Samsung Galaxy S3 mini	100
Samsung Galaxy Gold	50
Smartwatches	
LG watches	200
Samsung Gear	100



(a) (b)

Figure 4 The distribution of the datasets: (a) phone accelerometer dataset (b) watch accelerometer dataset.

contained 13,062,475 and 3,540,962 records of raw time-series tri-axil accelerometer data, respectively. The phone accelerometer dataset was recorded from four models with eight smartphones (2 × 4, 2 × Samsung Galaxy S3, 2 × Samsung Galaxy S3 Mini, and 2 × Samsung Galaxy Gold). The watch accelerometer dataset was recorded from two models with four smartwatches (2 × LG watches, and 2 × Samsung Gear). Each dataset was recorded by nine users, with each user performing a set of activities including walking, sitting, biking, standing, moving upstairs, and moving downstairs). Each activity was performed for five minutes. In this study, the six activities were considered as target clusters. The structure and distribution of both datasets were different (Figure 4). Each of the mentioned devices had different frequencies. Thus, each device had to be treated separately, especially in the filtration step (Table 2).

5.2 Data preprocessing

Due to the large-scale of the HAR data we used in this study, which was created from readings of various frequencies of smartphones and smartwatches, it contained corrupted and empty values. This affects the clustering accuracy. Therefore, to pass a pure dataset into the algorithm, some preprocessing steps were performed. After the cleaning steps (i.e., removing null values and records), the phone and watch datasets decreased to 11,279,275 and 3,020,605 records, respectively. After the cleaning, the sampling technique was applied based on the devices as the first stage of the stratified sampling. The filtration was then performed on the sampled data, and the second stage of the stratified sampling was applied successively. Afterward, the PCA was used for data visualization and feature reduction, in which the corrupted data could be shown clearly with the reduction of the data into two components. In the phone dataset, user "f" did not exist in Samsung Galaxy S3_1, and due to the lack of all users' data in the Samsung Galaxy S3 mini_2 device, it was considered as noise. As can be seen in Figure 5, each cluster (target clusters) is represented as colored line. It is observed that the figures do not contain all colored lines; hence, the values associated with the corrupted data were excluded. The excluded data represents 0.08% of all data. Therefore, it did not affect the structure of the dataset.

Regarding the watch dataset, Samsung Gear 1 and Samsung Gear 2 did not contain data from user "i". Additionally, users "f" and "h" of Samsung Gear 1, users "g" and "h" of LG Watch 1, and users "b", "d", and "h" of LG watch 2 were corrupted and did not contain all clusters. Therefore, they were excluded from the dataset (Figure 5).

The datasets contained nine features. Five representative features (accelerometer reading x, y, and z; creation time; and arrival time) were selected as input parameters, and PCA was used to reduce the five features into two components.

5.2.1 Filtration

Owing the heterogeneity of the datasets, which was caused by multisensory devices, after the first stage of the stratification sampling step, a frequency filtration was accomplished on the accelerometer data. As a nature of HAR data and the variation of smartphone model frequencies, the high frequencies are a resource of noise, which affect the clustering accuracy. To avoid and filter the high frequencies, a low-pass filtration was

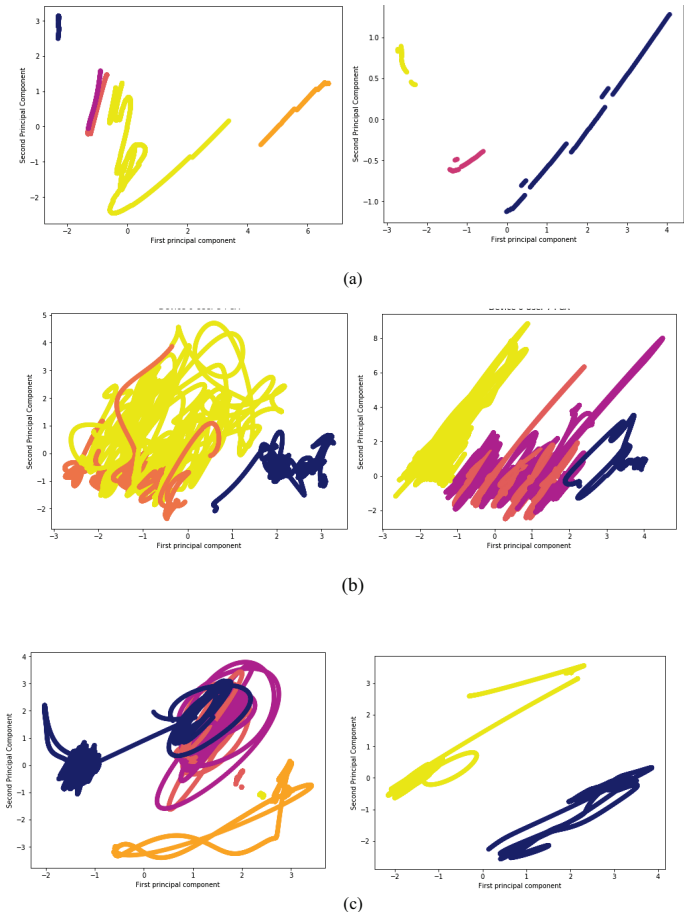


Figure 5 Corrupted data: (a) users "a" and "b" of Samsung S3 mini 2; (b) users "f" and "h" of Samsung Gear 1; and (c) users "h" and "d" of LG watch 1 and LG watch 2, respectively.

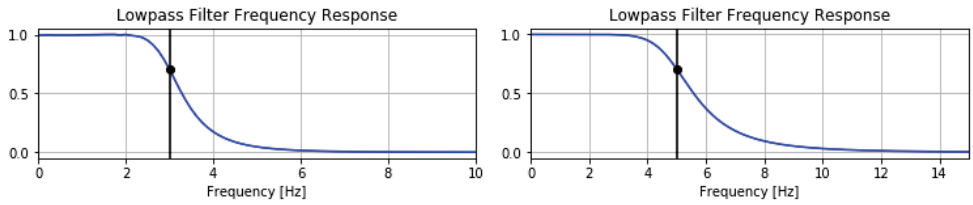


Figure 6 Order and cut-off frequency: (a) Phone accelerometer; (b) watch accelerometer.

applied on the raw accelerometer readings, which allowed low frequencies to be passed. Unlike the study in [2], the raw data were filtered directly on accelerometer features (x, y and z) separately, as addressed in [55]. Further, as mentioned in Section 4, stratified sampling was accomplished based on devices, then filtration was applied sequentially. Using stratified sampling, a reduction of the heterogeneity was observed because the data were separated into groups based on devices, and the correlation between data was reduced as well. Further, the heterogeneity was significantly reduced by implementing low-pass filtering directly without the need to implement frequency and time domain as in [2]. After several trials, a sixth-order and cut-off frequency of 3 Hz, and a fifth-order and cut-off frequency of 5 Hz for phone and watch datasets, respectively, were applied with a low-pass filter (Figure 6).

As addressed in [73, 74], this cut-off frequency was appropriate to filter the data without losing any information (Figure 7).

5.2.2 Feature reduction by principal component analysis

As mentioned in Subsection 5.2, the datasets consist of nine features. The most representative features used were (x, y, z, arrival time, and creation time). Because of the value variations of arrival and creation time features compared to the other features, a value normalization process was performed on the five features after the second stage of stratification. Further, due to the difficulty of data visualization and time-consumption of kernel matrix calculations, PCA was applied on the five features to represent them in two components. PCA is statistical feature extraction strategy in feature engineering in which a linear transformation is used to modify a set of possibly correlated observations into a number of uncorrelated variables called principal components, where the first component contains the highest variability of a data [75]. Additionally, it used for feature and dimensionality reduction [76]. PCA was useful in this study to plot the features easily with six activities, reduce the kernel

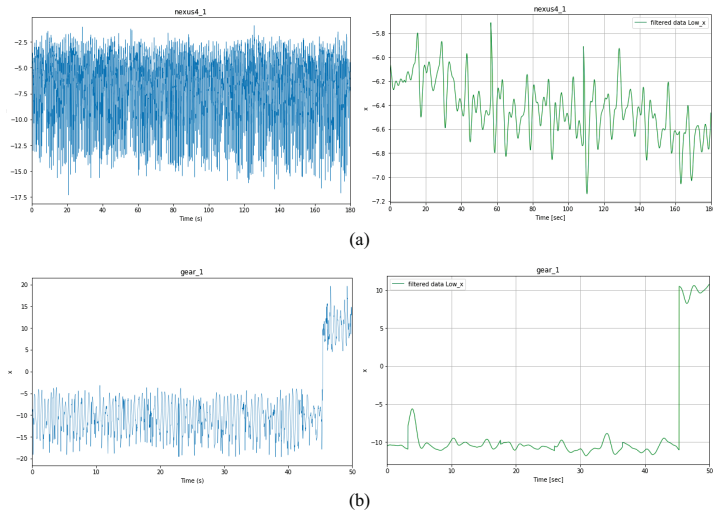


Figure 7 Duration of 180 s for phone and 50 s for watch data: (a) (left) raw x-axis of the accelerometer of Nexus4_1 and (right), filtered accelerometer data where the user was moving stairs-up; (b) (left) raw x-axis of the accelerometer of Samsung Gear 1, and (right) filtered accelerometer data where the user was moving stairs-up.

Table 3 Execution time of kernel K-means on the small proportion of the second SRS stage.

Time	Serial	Parallel
Total execution	11:17:27.541 hours	2:49:48.567 hours
Maximum iteration	9.230 seconds	4.158 seconds
Average iteration	3.682 seconds	1.655 seconds
Maximum kernel matrix	10:03.359 minutes	27.540 seconds
Average kernel matrix	04:54.668 minutes	12.054 seconds

matrix calculations, and avoid high correlations between the data points.

6. RESULTS

This section presents results that have been obtained from our approach to cluster the data using approximation kernel K-means approach. We focused on the Gaussian kernel function. The two smartphone and smartwatch datasets were utilized to verify our approach. The precision, recall, f-score, and (NMI) were used as validation metrics to assess the quality of the clustering algorithm. Although our approach is a clustering problem, we assessed our work with the classification metrics and compared them with the results in [72]. Additionally, NMI was used to assess the mutual dependencies of cluster points. We used the Google Cloud Platform GCP DataProc cluster and Google storage for storing the data. The google storage specialized for storing big data to be manipulated by the google cloud platform. The serial implementation carried out with single machine type n1-standard-4, which worked as a master node with 4 cores CPU and 15 GB of random access memory (RAM). The platform for parallel implementation was one master and six workers with the same properties above.

6.1 Results of phone dataset

As described in Section 4, regarding the sampling techniques we used in this study, the phone dataset was grouped into eight devices that represented eight strata based on the first stage of stratified sampling. The low-pass filter was applied in each stratum. As the second stage of stratified sampling, each stratum (device) was grouped into nine strata based on users. Further, the Samsung Galaxy S3 mini_2 device was excluded; thus, 62 strata were finally obtained.

Each stratum represented all the data of a single user from single devices. Furthermore, each stratum contained all six clusters (activities), and each of the 62 strata was divided into big and small proportions as the first SRS stage. Next, the small proportion was also divided into small and big proportions as the second SRS stage. Thereafter, the kernel K-means was applied on the small proportion in both serial and parallel manner. As seen in Table 3, the iteration time represents the duration of the algorithm to calculate the cluster assignment until it converges. These values change according to the data size. Regarding our approach, we calculated the kernel matrix and cluster assignment to the users' data separately. The maximum iteration time represents the longest iteration taken by the algorithm in the dataset. Moreover, the average iteration time represents the average time of all the iterations. Due to the kernel matrix calculated for each user data separately, the maximum kernel time represents the longest kernel matrix calculations for certain users. Further, the average kernel matrix time represents the average time of all kernel matrix calculations. The kernel matrix time is changed according to user data. The high size of data means the long duration of iteration time. These values are the main metrics for time assessment. Therefore, the total execution time of serial implementation is 11 hours and 17 minutes. In contrast, the time decreased significantly in the parallel implementation to 2 hours and 49 minutes. Additionally, it is important to emphasize that the maximum iteration time was reached in 9 seconds in serial and 4 seconds in parallel.

Table 4 Global metric values of the whole small proportion of the first SRS stage.

Metrics	Serial		Parallel	
	Approximation with Euclidean	Trained with KNN	Approximation with Euclidean	Trained with KNN
NMI	0.831	0.842	0.832	0.843
Precision	0.840	0.841	0.830	0.830
Recall	0.857	0.863	0.855	0.859
F-score	0.843	0.846	0.836	0.838

Table 5 Total execution time of approximation with Euclidean distance and KNN classifier on the small proportion of the second SRS stage.

Approximation with Euclidean	Trained with KNN
25.836 seconds	57:31.10 minutes

Table 6 Metrics and execution time of the approximation step on the big proportion of the first SRS stage.

Metrics	Values
NMI	0.840
Precision	0.903
Recall	0.902
F-score	0.902
Total execution time of approximation step on the big proportion of the first SRS stage	05:10.038 minutes

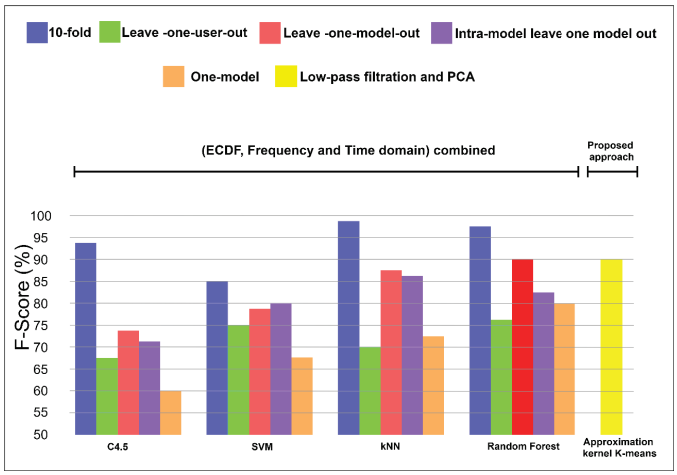


Figure 8 Comparison of HAR performances in [72] and our proposed approach on smartphone dataset.

The big proportion of the second SRS stage was approximated by both Euclidean distance and KNN classifier. Hence, the evaluation metrics of the whole small proportion of the first SRS stage is shown in Table 4.

As can be seen in Table 4, the evaluation metrics of both Euclidean distance and KNN classifier are almost the same, with only minor differences. However, due to a large number of KNN neighbors, in our approach, when we set the neighbors as 200, we noticed a large difference between both approaches in terms of execution time (Table 5). As can be seen, the total execution time of approximation with Euclidean decreased compared to approximation with KNN classifier.

After obtaining the whole small proportion of the first SRS stage clusters, the big proportion of the first SRS stage remains. A new centroid is calculated and the rest of the data (big proportion of first SRS stage) is approximated using both Euclidean distance and KNN classifier. As can be seen in Table 6, the evaluation metrics of the whole data clustering using the approximation approach significantly increased, especially the f-score measure. Additionally, due to the higher execution time of KNN compared to the Euclidean approach, the Euclidean distance execution time of 5 minutes was selected. In [72], the authors utilized four classification algorithms using three feature types (empirical cumulative distribution functions, frequency, and time domain) combined across various cross-validation and evaluation methods.

Our approach with only low-pass filtration and PCA feature selection obtained higher results than all the classification algorithms used in [72], except the 10-fold technique (Figure 8). Although the 10-fold technique in [72] obtained relatively higher results, the values are not realistic as addressed in the same paper. As the target data of clustering problems does not exist, the clustering problem results are usually less than the classification problem results. In our approach, we reached higher results than the classification problem, and this indicates that our approach is efficient. Furthermore, we also concluded that when the data size increased, the efficiency as an evaluation metric (i.e. F-score) also increased. It went from 0.84 in the whole small proposition to 0.902 in the whole dataset.

Table 7 Execution time of kernel K-means on the small proportion of the second SRS stage.

Kernel execution time	Duration time
Total execution	01:33:02.717 hours
Maximum iteration	5.184 seconds
Average iteration	1.796 seconds
Maximum kernel Matrix	41.553 seconds
Average kernel matrix	10.395 seconds

Table 8 Global metrics values of the whole small proportion of the first SRS stage of watch dataset.

	Parallel	
	Approximation with Euclidean	Trained with KNN
NMI	0.714	0.731
Precision	0.643	0.678
Recall	0.681	0.702
F-score	0.647	0.674

Table 9 Total execution time of approximation with Euclidean distance and KNN classifier on the small proportion of second SRS stage of watch dataset.

Approximation with Euclidean	Trained with KNN
7.412327 seconds	24.882297 seconds

Table 10 Global metrics values of the whole small proportion of the first SRS stage.

Metrics	Values
NMI	0.720
Precision	0.817
Recall	0.799
F-score	0.794
Total execution time of approximation step on the big proportion of the first SRS stage	0:01:11.084461 minutes

6.2 Results from watch dataset

The watch dataset has a different structure and data distribution. It was recorded by four devices; therefore, the record number and dataset size are less than the phone dataset. The watch dataset was grouped into 4 strata based on 4 devices as the first stage of stratified sampling, then the low-pass filtering was applied on each stratum. According to the users, each stratum (device) was grouped into nine strata as the second stage of stratification. After the exclusion of corrupted data as mentioned in Subsection 5.2, we obtained 30 strata. Each of the 30 strata was also divided into big and small proportions as the first SRS stage. The small proportion was also divided into small and big proportion as the second SRS stage. Afterward, the kernel K-means was applied on the small proportion in a parallel manner.

As summarized in Table 7, the execution time calculations of kernel K-means is 1 hour and 33 seconds, and the maximum iteration time is 5 seconds. As can be seen, although the size of the watch dataset is less than the phone dataset, the total execution time is slightly high, and the maximum iteration time is almost the same. This is because the LG watch_1 contained most data records and even more than one device compared with the phone dataset. Thus, the kernel function calculations are high. The increase in the kernel function calculations leads to an increase in the total execution time spent to realize the clustering, and it also leads to resource consumption.

Thereafter, the big proportion of the second SRS stage was approximated by both Euclidean distance and KNN classifier. We evaluated the results of the whole small proportion of the first SRS stage with the assessment metrics (Table 8).

The evaluation metrics of both Euclidean distance and KNN classifier are presented in Table 8; they are very similar, with minor differences, as observed in the phone dataset. As addressed in the previous section, the difference is only in terms of execution time. Due to the smaller dataset size relative to the phone dataset, we set the neighbors as 25. We also noticed a difference between the two approaches in terms of execution time. The short execution time in the approximation with the Euclidean distance compared to the KNN classifier is shown in Table 9. Here, the approximation with the Euclidean distance was implemented in 7 seconds and the KNN classifier in 24 seconds.

Here, we repeated the same steps as we performed in the phone dataset. After obtaining the whole small proportion of the first SRS stage, new centroids are calculated, and the big proportion of the first SRS stage is approximated. Due to the time-consumption of KNN, we utilized the Euclidean distance as an approximation approach.

As can be seen in Table 10, the evaluation metrics of the whole watch data clustering using the approximation approach significantly increased compared to the whole small proportion of the first SRS stage. Furthermore, the total execution time is 1 minute. From Table 10, we conclude that our metric results are higher compared to results in [72], in terms of the f-score (Figure 9). As we addressed in the phone dataset results in the previous subsection, the higher result in the mentioned paper is

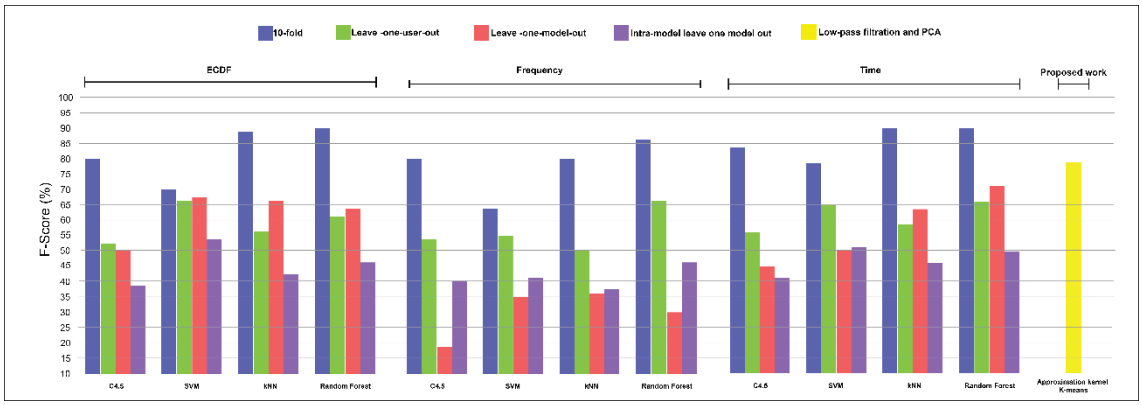


Figure 9 Comparison of HAR performances in [72] and our proposed approach on smartwatch dataset.

not realistic, thus indicating that our approach in both phone and watch datasets provided efficient clustering.

7. DISCUSSION

Nowadays, the smartphone and wearable devices have an enormous sensing ability including accelerometer, gyroscope, and GPS to record human daily activity. The data generated by the sensors let to have high heterogeneity. In other words, if the person performs more than one activity at the same time, the recognition task becomes difficult [8]. Owing to the heterogeneity issue, it is impracticable to pass the raw data to clustering techniques. Therefore, HAR data with high heterogeneity should be decreased somehow to be smooth to the recognition system. In addressing this challenge, we performed such analysis to decrease the heterogeneity like filtration and sampling, and feature reduction like PCA. Raw accelerometer data have been obtained from the Heterogeneity Human Activity Recognition Dataset available on UCI [72] contains datasets collected by nine users performed six activities.

During this work process, we faced several challenges. In the filtration phase, we attempted several trials until we set a proper cut-off frequency and order. As a result of the heterogeneity of the data it was difficult to show the outliers, thus by performing PCA we could reduce the features into two components in which be able to visualize the data and identify the outliers. Google cloud provides a perfect parallel environment to implement a project, however, still, there are several challenges. Because of the Spark written in Scala language it must utilize Pyspark Application Programing Interface (API) that provides interactions between Spark and Python language. Another challenge is the implementation of the kernel matrix calculations in a parallel manner. We employed one master and six workers. According to the Tables 3 and 7, it can be seen that the execution time of parallel implementation is decreased significantly.

Through the research reviewed regarding HAR recently. Most of the research concentrated on HAR datasets as a classification task [8–21]. However, they did not address HAR as a clustering problem. Additionally, the research reviewed regarding kernel K-means techniques did not implement on HAR. Thus, we filled the gap between them and merged the two fields. Therefore, we proposed a clustering technique to deal with HAR. Generally, due to HAR data generated by various types of sensors, it causes a heterogeneity to the data. Hence, it causes the nonlinearity of the data. Therefore, we employed the kernel K-means algorithm that works well with nonlinearity datasets [25] to recognize human activities. As a result of the kernelization technique consume more time and computational resources, we used the approximation technique. The idea behind approximation is that dividing data into subsets of data in which a kernelization technique works using only a small part of the kernel matrix [25, 60, 77] and approximate the remaining data based on Euclidean distance and KNN which have been proposed in this work.

After using the approximation technique in both serial and parallel manner, we observed a high reduction in parallel compared with serial implementation in terms of processing time. Further, the parallel implementation of HAR data is not addressed in the recent papers.

Usually, when any approach implemented to solve a problem, the work has to be assessed in terms of accuracy. To validate our clustering approach, we compared our approach with the works used the same data. As can be seen in the Tables 6 and 10, the obtained results are very promising and prove the validity and robustness of our approach. Our approach is related to Stisen et al. [72], who clustered based on devices that have recorded the data and treat measuring human activity as a classification problem and used several classification algorithms. Besides, Dobbins and Rawassizadeh [2] clustered the same datasets based on K-means, Hierarchal and DBSCAN algorithms by using feature engineering without parallel implementation. Unlike these works, we applied a clustering technique based on the approximation kernel K-means algorithm in a parallel manner. However, the obtained results should be improved more by using more feature engineering in which to get higher results in terms of accuracy. If a new dataset with the same properties is collected, then the same process would be tested, as these have confirmed to be the best technique for this type of data.

8. CONCLUSIONS

In this work, we presented and analyzed datasets obtained from accelerometers that can detect human physical activities. Due to the heterogeneities and high correlations of the datasets, we used low-pass filtration and data reduction using PCA, which reduced the features into two components. Subsequently, we clustered the data using the approximate kernel k-means approach. For

the approximation step, we utilized the Euclidean distance and KNN classifier methods. To select an unbiased sample for the approximation approaches, we used multistage sampling techniques based on stratification and SRS. The best results were obtained from phone and watch datasets, with 90% and 79% accuracies, respectively, in terms of the f-score metric. Further, we compared our results with the classification algorithms mentioned in [72]. Although our approach was performed as a clustering problem, it obtained better results than classification algorithms results mentioned in the same paper. Additionally, our approach was implemented in a parallel environment; thus, a significant improvement was observed in terms of execution time.

The previous studies fundamentally treat HAR data as a classification problem by using several classification techniques. A lack of a number of works focused on HAR as a clustering problem encouraged us to fill this gap and propose a new technique in HAR field. Another inference from previous works has not addressed the parallel implementation of HAR. Furthermore, the studies related to HAR have not addressed big data. Thus, we proved that our approach has not been implemented previously on HAR. Therefore, our study fills the gap between several fields by implementing a parallel approximation kernel k-means clustering algorithm on HAR as big data.

As a result of the nonlinearity and big size of data, we obligated to implement the kernel K-means algorithm in an approximation approach in parallel. Investigating a new approach to deal with all data without using approximation technique and optimizing the technique we used by using feature engineering to increase the accuracy will be our future work.

REFERENCES

1. Williamson J., Liu Q., Lu F., et al. (2015) Data sensing and analysis: Challenges for wearables. In: 20th Asia and South Pacific Design Automation Conference, ASP-DAC 2015. pp 136–141.
2. Dobbins C., Rawassizadeh R. (2018) Towards Clustering of Mobile and Smartwatch Accelerometer Data for Physical Activity Recognition. Informatics 5:29. doi: 10.3390/informatics5020029.
3. Machado I.P., Luísa Gomes A., Gamboa H., et al. (2015) Human activity data discovery from triaxial accelerometer sensor: Non-supervised learning sensitivity to feature extraction parametrization. Inf Process Manag 51:204–214. doi: 10.1016/j.ipm.2014.07.008.
4. Poppe R. (2010) A survey on vision-based human action recognition. Image Vis Comput 28:976–990. doi: 10.1016/j.imavis.2009.11.014.
5. S. Z., Z. W, J. N., et al. (2017) A Review on Human Activity Recognition Using Vision-Based Method. J Healthc Eng 2017:
6. Golghate A. A., Shende S. W. (2014) Parallel K-Means Clustering Based on Hadoop and Hama. IJCAT Int J Comput Technol 1: 33–37.
7. Aggarwal C. C., Reddy CKR (2014) DATA CLUSTERING Algorithms and Applications. In: An Introduction to Toxicogenomics. CRC Press, Boca Raton
8. Jobanputra C., Bavishi J., Doshi N. (2019) Human activity recognition: A survey. In: Procedia Computer Science. Elsevier B.V., pp 698–703
9. Kong Y., Fu Y. (2018) Human Action Recognition and Prediction: A Survey. arXiv 13:
10. Bragança H., Colonna J. G., Lima W. S., Souto E. (2020) A Smartphone Lightweight Method for Human Activity Recognition Based on Information Theory. Sensors 20:1856. doi:10.3390/s20071856.
11. Verma K. K., Singh B. M., Dixit A. (2019) A review of supervised and unsupervised machine learning techniques for suspicious behavior recognition in intelligent surveillance system. Int J Inf Technol. doi:10.1007/s41870-019-00364-0.
12. Zhuang Z., Xue Y. (2019) Sport-related human activity detection and recognition using a smartwatch. Sensors (Switzerland) 19:1–21. doi: 10.3390/s19225001.
13. Xu H., Pan Y., Li J., et al. (2019) Activity Recognition Method for Home-Based Elderly Care Service Based on Random Forest and Activity Similarity. IEEE Access 7:16217–16225. doi: 10.1109/ACCESS.2019.2894184.
14. Li H., He X., Chen X., et al. (2019) Wi-Motion: A Robust Human Activity Recognition Using WiFi Signals. IEEE Access 7:153287–153299. doi: 10.1109/ACCESS.2019.2948102.
15. Feng C., Arshad S., Zhou S., et al. (2019) Wi-Multi: A Three-Phase System for Multiple Human Activity Recognition with Commercial WiFi Devices. IEEE Internet Things J 6:7293–7304. doi: 10.1109/JIOT.2019.2915989.
16. Raja M., Ghaderi V., Sigg S. (2018) WiBot! In-vehicle behaviour and gesture recognition using wireless network edge. In: Proceedings - International Conference on Distributed Computing Systems. IEEE, pp 376–387.
17. Micucci D., Mobilio M., Napoletano P. (2017) UniMiB SHAR: A dataset for human activity recognition using acceleration data from smartphones. Appl Sci 7:. doi: 10.3390/app7101101.
18. Sanal Kumar K. P., Bhavani R. (2019) Human activity recognition in egocentric video using PNN, SVM, kNN and SVM+kNN classifiers. Cluster Comput 22:10577–10586. doi: 10.1007/s10586-017-1131-x.
19. Wang K, Jun H, Zhang L (2019) Attention-Based Convolutional Neural Network for With Wearable Sensors. IEEE Sens J 19:7598–7604.
20. Zhu R., Xiao Z., Li Y., et al. (2019) Efficient Human Activity Recognition Solving the Confusing Activities Via Deep Ensemble Learning. IEEE Access 7:75490–75499. doi: 10.1109/ACCESS.2019.2922104.
21. Qi W., Su H., Yang C., et al. (2019) A fast and robust deep convolutional neural networks for complex human activity recognition using smartphone. Sensors (Switzerland) 19: doi: 10.3390/s19173731.
22. Zhao Z., Chen Y., Liu J., et al. (2011) Cross-people mobile-phone based activity recognition. In: IJCAI International Joint Conference on Artificial Intelligence. pp 2545–2550.
23. Hooda D., Rani R. (2020) Ontology driven human activity recognition in heterogeneous sensor measurements. J Ambient Intell Humaniz Comput. doi: 10.1007/s12652-020-01835-0.
24. Wang S., Gittens A., Mahoney M. W. (2019) Scalable kernel K-means clustering with nystrom approximation: Relative-error bounds. J Mach Learn Res 20:1–49.
25. Chitta R., Jin R., Havens T. C., Jain A. K. (2011) Approximate kernel k-means: solution to large scale kernel clustering. In: Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining - KDD '11. pp 895–903.
26. Ye Y., Wu Q., Zhexue Huang J., et al. (2013) Stratified sampling for feature subspace selection in random forests for high dimensional data. Pattern Recognit 46:769–787. doi: 10.1016/j.patcog.2012.09.005.
27. Chen C., Yangzhou J. (2010) Essence of two-dimensional principal component analysis. In: Proceedings - 2010 International Conference on Computational Intelligence and Security, CIS 2010. pp 280–282.
28. Bingbing L. (2018) A principal component analysis approach to noise removal for speech denoising. In: Proceedings-2018

International Conference on Virtual Reality and Intelligent Systems, ICVRIS 2018. IEEE, pp 429–432.
29. Zhang X., Ren X. (2011) Two Dimensional Principal Component Analysis based Independent Component Analysis for face recognition. In: 2011 International Conference on Multimedia Technology, ICMT 2011. pp 934–936.
30. Baydoun M., Ghaziri H., Al-Husseini M. (2018) CPU and GPU parallelized kernel K-means. J Supercomput 74:3975–3998. doi: 10.1007/s11227-018-2405-7.
31. Hadian A., Shahrivari S. (2014) High performance parallel k-means clustering for disk-resident datasets on multi-core CPUs. J Supercomput 69:845–863. doi: 10.1007/s11227-014-1185-y.
32. Honggang W., Jide Z., Hongguang L., Jianguo W. (2008) Parallel clustering algorithms for image processing on multi-core CPUs. In: International Conference on Computer Science and Software Engineering, ICSSE. pp 450–453.
33. Naik D. S. B., Kumar S. D., Ramakrishna S. V. (2013) Parallel Processing Of Enhanced K-Means Using OpenMP. In: 2013 IEEE International Conference on Computational Intelligence and Computing Research. pp 1–4.
34. Wu R., Zhang B., Hsu M. (2009) Clustering billions of data points using GPUs. In: the combined workshops on UnConventional high performance computing workshop plus memory access workshop, ACM. pp 1–5.
35. Farivar R., Rebolledo D., Chan E., Campbell R. (2008) A parallel implementation of k-means clustering on GPUs. In: Pdpta. pp 212–312.
36. Kucukyilmaz T. (2014) Parallel K-Means Algorithm for Shared Memory Multiprocessors. J Comput Commun 2:15–23.
37. Bani Baker Q. B., Balhaf K. (2017) Exploiting GPUs to accelerate white blood cells segmentation in microscopic blood images. In: 2017 8th International Conference on Information and Communication Systems, ICICS 2017. pp 136–140.
38. Lv Z., Hu Y., Zhong H., Wu J. (2010) Parallel K-Means Clustering of Remote Sensing Images Based on MapReduce. WISM 2010, LNCS 6318 162–170.
39. Zhao W, Ma H, He Q (2009) Parallel K -Means Clustering Based on MapReduce. CloudCom 674–679.
40. Moertini V., Venica L. (2016) Enhancing Parallel k-Means Using Map Reduce for Discovering Knowledge from Big Data. In: IEEE International Conference on Cloud Compuing and Big Data Analysis. pp 81–87.
41. Abdouli A. E. L., Hassouni L., Anoun H. (2017) Mining Tweets of Moroccan Users using the Framework Hadoop, NLP, K-means and Basemap. In: Intelligent Systems and Computer Vision (ISCV).
42. Xia D., Wang B., Li Y., et al. (2015) An Efficient MapReduce-Based Parallel Clustering Algorithm for Distributed Traffic Subarea Division. Hindawi Publ Corp Discret Dyn Nat Soc.
43. Wang B., Yin J., Hua Q., et al. (2016) Parallelizing K-Means-Based Clustering on Spark. In: International Conference on Advanced Cloud and Big Data. pp 31–36.
44. Cloud G. https://cloud.google.com/
45. Tsapanos N., Tefas A., Nikolaidis N., Pitas I. (2018) Big Data Clustering with Kernel k -Means: Resources, Time and Performance. Int J Artif Intell Tools 27:1860006. doi: 10.1142/s0218213018600060.
46. Albert M. V., Toledo S., Shapiro M., Kording K. (2012) Using mobile phones for activity recognition in Parkinson's patients. Front Neurol NOV: 1–8. doi: 10.3389/fneur.2012.00158.
47. Álvarez de la Concepción MÁ, Soria Morillo LM, Álvarez García JA, González-Abril L (2017) Mobile activity recognition and fall detection system for elderly people using Ameva algorithm. Pervasive Mob Comput 34:3–13. doi: 10.1016/j.pmcj.2016.05.002.
48. Lee J, Kim J (2016) Energy-Efficient Real-Time Human Activity Recognition on Smart Mobile Devices. Mob Inf Syst 2016:. doi: 10.1155/2016/2316757.
49. Luštrek M., Kaluža B. (2009) Fall detection and activity recognition with machine learning. Inform 33:205–212.
50. Vo QV, Hoang MT, Choi D (2013) Personalization in mobile activity recognition system using K-medoids clustering algorithm. Int J Distrib Sens Networks 2013:. doi: 10.1155/2013/315841.
51. Hsu H. H., Chu C. T., Zhou Y., Cheng Z. (2015) Two-phase activity recognition with smartphone sensors. In: Proceedings - 2015 18th International Conference on Network-Based Information Systems, NBiS 2015. pp 611–615.
52. Borazio M., Van Laerhoven K. (2013) Using time use with mobile sensor data: A road to practical mobile activity recognition? In: Proceedings of the 12th International Conference on Mobile and Ubiquitous Multimedia, MUM 2013.
53. Manzi A., Dario P., Cavallo F. (2017) A human activity recognition system based on dynamic clustering of skeleton data. Sensors (Switzerland) 17:. doi: 10.3390/s17051100.
54. Liu J., Teng G., Hong F. (2020) Human activity sensing with wireless signals: A survey. Sensors (Switzerland) 20:. doi: 10.3390/s20041210.
55. Gjoreski H., Gams M. (2011) Accelerometer Data Preparation for Activity Recognition. In: International Multiconference Information Society.
56. Xu L., Yang W., Cao Y., Li Q. (2018) Human activity recognition based on random forests. In: ICNC-FSKD 2017 - 13th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery. pp 548–553.
57. Ar I., Akgul Y. S. (2013) Action recognition using random forest prediction with combined pose-based and motion-based features. In: 8th International Conference on Electrical and Electronics Engineering- ELECO 2013. pp 315–319.
58. Ignatov A. D., Strijov V. V. (2015) Human activity recognition using quasiperiodic time series collected from a single triaxial accelerometer. Multimed Tools Appl 75:7257–7270. doi: 10.1007/s11042-015-2643-0.
59. Tzortzis G., Likas A. (2008) The Global Kernel k-Means Clustering Algorithm. In: 2008 International Joint conference on Neural Networks (IJCNN 2008). pp 1977–1984.
60. Shawe-Taylor J., Cristianini N. (2004) Kernel methods for pattern analysis. Cambridge University Press, Cambridge.
61. Tsapanos N., Tefas A., Nikolaidis N., et al. (2015) Fast kernel matrix computation for big data clustering. Procedia Comput Sci 51:2445–2452. doi: 10.1016/j.procs.2015.05.352.
62. Kong D., Kong R. (2013) A fast and effective kernel-based K-means clustering algorithm. In: Proceedings of the 2013 3rd International Conference on Intelligent System Design and Engineering Applications, ISDEA 2013. pp 58–61.
63. Tzortzis G. F., Likas A. C. (2009) The Global Kernel K-Means Algorithm for Clustering in Feature Space. IEEE Trans Neural Networks 20:1181–1194. doi: 10.1109/TNN.2009.2019722.
64. Wang C., Lai J., Zhu J. (2010) A Conscience On-line Learning Approach for Kernel-Based Clustering. In: IEEE International Conference on Data Mining. pp 531–540.
65. Rong Zhang, Rudnicky A. I. (2002) A large scale clustering scheme for kernel K-Means. Object Recognit Support by user Interact Serv Robot IEEE 4:289–292. doi: 10.1109/icpr.2002.1047453.
66. Baydoun M., Dawi M., Ghaziri H. (2016) Parallel Kernel K-Means on the CPU and the GPU. In: Proceedings of the International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA). pp 117–122.
67. Tsapanos N., Tefas A., Nikolaidis N., Pitas I (2015) Distributed, mapreduce-based nearest neighbor and e-ball kernel k-means. Proc-2015 IEEE Symp Ser Comput Intell SSCI 2015 509–515. doi: 10.1109/SSCI.2015.81.
68. Barreiro P. L., Albandoz J. P. (2001) Population and sample. Sampling techniques.

69. Souza V. M. A., Silva D. F., Gama J., Batista GEAPA (2015) Data stream classification guided by clustering on nonstationary environments and extreme verification latency. In: SIAM International Conference on Data Mining 2015, SDM 2015. pp 873–881.
70. Zhao S., Li W., Cao J. (2018) A user-adaptive algorithm for activity recognition based on K-means clustering, local outlier factor, and multivariate gaussian distribution. Sensors (Switzerland) 18:. doi: 10.3390/s18061850.
71. Pius Owoh N., Mahinderjit Singh M, Zaaba ZF (2018) Automatic annotation of unlabeled data from smartphone-based motion and location sensors. Sensors (Switzerland) 18:. doi: 10.3390/s18072134
72. Stisen A., Blunck H., Bhattacharya S., et al. (2015) Smart Devices are Different: Assessing and Mitigating Mobile Sensing Heterogeneities for Activity Recognition. In: ACM SenSys. pp 127–140.
73. Zhang Y., Markovic S., Sapir I., et al. (2011) Continuous functional activity monitoring based on wearable tri-axial accelerometer and gyroscope. In: 2011 5th International Conference on Pervasive Computing Technologies for Healthcare and Workshops, PervasiveHealth 2011. pp 370–373.
74. Mayagoitia R. E., Nene A. V., Veltink P. H. (2002) Accelerometer and rate gyroscope measurement of kinematics: An inexpensive alternative to optical motion analysis systems. J Biomech 35:537–542. doi: 10.1016/S0021-9290(01)00231-7.
75. Mousa F. A., El-Khoribi R. A., Shoman M. E. (2016) A Novel Brain Computer Interface Based on Principle Component Analysis. In: Procedia Computer Science. Elsevier Masson SAS, pp 49–56.
76. Jiang C., Zhang F., Wang J., et al. (2019) Robust Principle Component Analysis with Intra-Block Correlation. Neurocomputing. doi: 10.1016/j.neucom.2019.12.092.
77. Ahmadvand H., Goudarzi M., Foroutan F. (2019) Gapprox: using Gallup approach for approximation in Big Data processing. J Big Data 6:. doi: 10.1186/s40537-019-0185-4.