Published in final edited form as:

Proc IEEE Int Conf Healthc Inform Imaging Syst Biol. 2012 September ; 2012: 23–30. doi:10.1109/HISB.2012.13.

Aggregated Indexing of Biomedical Time Series Data

Jonathan Woodbridge,
Computer Science Department, University of California, Los Angeles, California

Bobak Mortazavi,
Computer Science Department, University of California, Los Angeles, California

Majid Sarrafzadeh, and
Computer Science Department, University of California, Los Angeles, California

Alex A.T. Bui
Medical Imaging Informatics, University of California, Los Angeles, California

Jonathan Woodbridge: jwoodbri@cs.ucla.edu; Bobak Mortazavi: bobakm@cs.ucla.edu; Majid Sarrafzadeh: majid@cs.ucla.edu; Alex A.T. Bui: buia@mii.ucla.edu

Abstract

Remote and wearable medical sensing has the potential to create very large and high dimensional datasets. Medical time series databases must be able to efficiently store, index, and mine these datasets to enable medical professionals to effectively analyze data collected from their patients. Conventional high dimensional indexing methods are a two stage process. First, a superset of the true matches is efficiently extracted from the database. Second, supersets are pruned by comparing each of their objects to the query object and rejecting any objects falling outside a predetermined radius. This pruning stage heavily dominates the computational complexity of most conventional search algorithms. Therefore, indexing algorithms can be significantly improved by reducing the amount of pruning.

This paper presents an online algorithm to aggregate biomedical times series data to significantly reduce the search space (index size) without compromising the quality of search results. This algorithm is built on the observation that biomedical time series signals are composed of cyclical and often similar patterns. This algorithm takes in a stream of segments and groups them to highly concentrated collections. Locality Sensitive Hashing (LSH) is used to reduce the overall complexity of the algorithm, allowing it to run online. The output of this aggregation is used to populate an index. The proposed algorithm yields logarithmic growth of the index (with respect to the total number of objects) while keeping sensitivity and specificity simultaneously above 98%. Both memory and runtime complexities of time series search are improved when using aggregated indexes. In addition, data mining tasks, such as clustering, exhibit runtimes that are orders of magnitudes faster when run on aggregated indexes.

Keywords

Time series signals; Indexing; Data mining

I. Introduction

Remote and wearable medical sensing can improve the quality of care for those with various ailments including congestive heart failure (CHF) [1][2], arrhythmias [3], diabetes [4], and mental illness [5]. Such systems rely on continuous patient monitoring by various types of sensors, such as accelerometers for activity monitoring; electrocardiogram (ECG) for heart monitoring; and plethysmograph for organ volume monitoring. These devices have the potential to create massive amounts of data, making manual inspection infeasible. Therefore, medical time series databases must be able to store, index, and mine large datasets to enable proper analysis of a patient's health.

Difficulties in searching and mining time series data arise from the high dimensionality and the sheer size of the data. Techniques such as k-means clustering and Motif Discovery are not tractable due to high complexities ($O(n^{dk+1} \log n)$ and $O(dn^2)$ respectively, where d is the number of dimensions, k is the number of clusters, and n is the number of objects). Previous works have reduced the number of dimensions to help mitigate the inherent high runtime complexities of data mining algorithms [6]. However, certain domains, such as medical time series signals, are largely dominated by n ($n >> d$) [7] and therefore, reducing d does very little to improve the overall runtime. In addition, a significant decrease in d may not be possible as the underlying intrinsic dimensions may also be large. Attempts to decrease d to a smaller space than the intrinsic dimensions will impair the overall discriminatory power of any classifier.

Search and data mining tasks rely heavily on an optimal distance function. Finding an optimal distance function becomes more difficult with an increasing number of dimensions as distance norms have the property to converge with an increasing number of dimensions [8]. It has been shown that this convergence is true for any p-norm and is more related to the intrinsic dimensions than the embedding dimensions [9]. However, [10] suggests that L_p norms are effective when considering small neighborhoods. This finding was also shown experimentally in [11]. Therefore, a match can be defined as $\|u - v\|_p < R$ where R defines a tight radius (or small neighborhood). Clustering with such a distance measure will result in a large number of precise groupings.

Medical time series signals are unique from many other time series signals in their inherent redundancy. For example, ECG measurements of the heart produce a repetitive signal with little differences between subsequent beats. Therefore, a significant reduction in the size of a medical time series index can be observed when using such an aggregations. As shown later, the number of groupings grows logarithmically with increasing n for many real-world biomedical datasets.

This paper presents an online aggregation algorithm for constructing efficient time series data indexes. This algorithm takes in a stream of objects and groups them to highly concentrated collections. An aggregated index is composed of an object representative from each collection. Searches and data mining tasks are constrained to only the aggregated

index, thereby significantly improving performance with respect to memory and computation.

For the purpose of this paper, a time series $T = \{t_1, t_2, \ldots t_n\}$ is defined as an ordered set of points in the time domain. An object is defined as a segment $s = \{t_i, t_{i+1}, \ldots, t_{i+m-1}\}$ where s is an ordered subset of points from T and $m \leq n$. Similarity is defined by the L_2 norm (but any L_p norm can be used). Locality Sensitive Hashing (LSH) [12] is used to reduce the overall complexity of the algorithm, allowing it to run online.

The proposed algorithm runs similar to the Online Facility Location problem [13]. An input segment is hashed using LSH. If this hash collides with a facility (e.g., an existing grouping), the segment is assigned to that facility. If no collisions occur, a new facility (or grouping) is created and inserted into the global hash table. The aggregation algorithm is based on the assumption that classes, in terms of classification, are composed of multiple tight sub-groupings where L_p norms have high discriminatory power [10].

Similar indexes have been proposed in textual databases such that redundant information is indexed only once [14][15][16]. Such indexes are often used with versioned data such as indexing the Internet Archive (a collection of over 85 billion versioned web pages over the last decade), version control systems, Wikis, data backup solutions, etc. In general, documents are broken down into fragments. Fragments that occur in multiple documents (or versions) are indexed only once. Fragments are chosen in an optimal (aligned) manner such that the number of index fragments is minimized. These systems have been shown to significantly improve search performance by reducing the overall search space.

Real data are used in the analysis of this paper. The proposed algorithm yields logarithmic growth of groupings while keeping sensitivity and specificity above 98%. Search and clustering performance (in terms of computations) is improved by several orders of magnitude. This algorithm has a low computation and memory complexity, allowing it to run online.

The rest of this paper is organized as follows. Section II discusses related work and motivation. Section III describes the presented algorithm in detail. Sections IV and V present the experimental setup and results respectively. Conclusions are presented in VI.

II. Background

A. Subsequence Matching

Subsequence matching is synonymous with the R-NN problem in high dimensional space. Nearest neighbors are defined as all objects that fall within distance R to a query object. For the purpose of this paper, distances are defined by the L_2 norm (Euclidean distance) and objects are represented as points in $\Re^d$. Sequential search is a naïve approach to solving the R-NN problem. This, of course, has an $O(n)$ computational complexity that is far too slow for large databases. Hence, an optimal solution would guarantee sub-linear complexity.

Authors in [6] presented a framework for subsequence matching using spatial access methods (such as R^*-trees, iSAX [17], etc). to index objects (or segments). Indexing is accomplished by first reducing the dimensionality of each segment using a reduction algorithm that satisfies the property of minimum bounds. Each reduced segment is inserted into a spatial index for subsequent searches. A query is satisfied by first reducing the query segment. The reduced segment's corresponding bounding boxes in the spatial index are interrogated for any matching segments. The true distance (from the original non-reduced segments) between any potential matches and the query segment is calculated. Any match that falls outside a predefined distance is pruned and not included in the result set.

The property of minimum bounds ensures no false dismissals. The minimum bounds property guarantees that the distance between two reduced objects is less than or equal to the distance between the corresponding original (non-reduced) objects:

$$\mathrm{dist}(\hat{u}, \hat{v}) \leq \mathrm{dist}(u, v), \quad (1)$$

where $\hat{u}$ and $\hat{\upsilon}$ are the dimensionally reduced counterparts of vectors $u, \upsilon \in \Re^d$. Hence, the initial set of all search results is a superset of all true matches. Reduction algorithms satisfying the property of minimum bounds include Piecewise Aggregate Approximation (PAA) [18], Discrete Fourier Transform (DFT) [6], and Chebyshev polynomials [19].

PAA is perhaps the simplest and most studied of these reduction algorithms. A reduction of segment X of size m is accomplished by dividing X into M equal size bins. The reduction of X is represented by the average of each of these bins. More formally:

$$\hat{x}_i = \frac{M}{m} \sum_{j=\frac{m}{M}(i-1)+1}^{\frac{m}{M}i} x_j, [18]. \quad (2)$$

An example of the reduction of an ECG signal is given in Fig. 1 with a total of 512 data points reduced to 25 dimensions. Note that much of the detail of the signal is lost in the reduction. As biomedical signals are largely composed of similar repetitive patterns (such as heart beats), reductions can lead to a large number of false positives, resulting in an increase in the amount of pruning. This issue is exacerbated by the finding that spatial indexes have been shown both theoretically and experimentally to perform worse than sequential search for data with as little as 10 dimensions [20]. Hence, any method based on the framework proposed by [6] will have a theoretical bound that is no less than linear. Due to these drawbacks, experimentation for this paper was performed on indexes based on Locality Sensitive Hashing (LSH) [21].

LSH was introduced as an alternative to spatial indexing schemes. LSH is based on a family of hashing functions that are (r_1, r_2, p_1, p_2)-sensitive meaning that for any υ, $q \in S$:

- if $\upsilon \in B(q, r_1)$ then $\Pr_H[h(q) = h(\upsilon)] \geq p_1$

- if $\upsilon \notin B(q, r_2)$ then $\Pr_H[h(q) = h(\upsilon)] \leq p_1$

where $p_1 > p_2$ and $r_2 > r_1$. The gap between p_1 and p_2 is increased by combining several functions from the same (r_1, r_2, p_1, p_2)-sensitive family.

Spatial indexing methods and LSH require pruning as the initial result sets contain false positives. This pruning heavily dominates the computational complexity of both methods. Therefore, it is imperative to reduce the amount of pruning in order to build scalable time series databases.

The authors in [7] propose limiting the index to salient segments. Salient segments are defined as segments that are probabilistically unlikely to occur. Salient segmentation transforms a time series signal into a time series saliency function (TSF) such that each point in the TSF is defined as:

$$TSF_i = -\log \prod_{j=i-\lfloor m/2 \rfloor}^{i+\lfloor m/2 \rfloor} \Pr(T_j | T_{j-1} = t_{j-1}) \quad (3)$$

Each local maximum is treated as salient and is inserted into the index. Non-salient segments are excluded from the index, thereby limiting the search space. [7] showed up to 98% reduction in the size of the index, but with a trade off of recall, which reduced to approximately 80%. Authors in [22] showed that Salient Segmentation can significantly improve the performance of LSH searches. The amount of prunings were reduced by more than 75% with approximately 20% degradation in recall. Index sizes in [22] were reduced by approximately 95%.

There are two weaknesses of salient segmentation. First, salient segmentation is useful when the segmentation is unknown, such as accelerometer data in activity recognition. In contrast, segmentation of signals such as ECG is a well-solved problem. Therefore, a domain-specific segmentation can exhibit a near identical reduction in index sizes with no loss of recall. Second, a biomedical time series salient index often contains redundancy. Biomedical signals are cyclical in nature and are often composed of many near identical patterns. It thus serves little use to independently index each of these similar patterns.

B. Mining time series signals

Algorithms for mining time series signals are often bounded by quadratic (or greater) computation complexities. *k*-means clustering and motif discovery are two well-studied algorithms for mining time series signals and are computationally bounded by $O(n^{dk+1} \log n)$ and $O(dn^2)$ respectively, where d is the number of dimensions, k is the number of clusters, and n is the number of objects. There are a number of works that attempt to decrease the average computationally complexities for both algorithms. For example, authors in [23] and [24] attempted to improve *k*-means and motif discovery respectively by using properties of distance measures (such as the triangle inequality). However, only the average runtimes were reduced and not the overall complexities. Hence, this paper proposes a scalable reduction of

n to significantly improve the runtime performance of these and other data mining algorithms.

C. L_p norms

Medical signals are generally cyclical and stationary. ECG, for example, consist of extremely similar and repetitive patterns. Treating small variations of the same pattern as independent serves little use when mining medical time series signals. As shown later, the number of intrinsic classes (unique patterns) grows logarithmically with the total number of patterns. Therefore, the computational and memory complexities can be significantly reduced by only considering unique patterns.

The aggregation method proposed by this paper uses the L_2 norm to define similarity. Previous works have questioned the feasibility of L_p norms to distinguish high dimensional objects. Most notably, authors in [8] show that L_p norms have the property of convergence with an increasing number of dimensions. Convergence results in little to no differentiation between the distance to the closest object and the farthest object. This finding suggests that L_p norms are meaningless in high dimensional space. However, the finding in [8] assumed objects to be identical and independently distributed (i.i.d.); a false assumption for most datasets.

[10] shows that, in terms of classification, classes are composed of multiple clusters. In terms of L_p norms, clusters are assumed to be Gaussian; and classes a mixture of Gaussians. The distribution of these classes can be estimated using the expectation-maximization (EM) algorithm. This model exhibits high discriminative power for high dimensional objects. Authors in [11] show similar discriminative power of L_p norms for both synthetic and real data. However, this power decreases with an increasing radius. Therefore, objects can be grouped to tight clusters without the loss of precision (with each cluster being represented by its center).

D. Indexing Textual Databases with Redundancy

Several methods for consolidating redundant information in textual databases have been proposed [14][15][16]. In general, these methods break down documents into several fragments. Fragments are indexed in lieu of the entire documents. Identical fragments that occur across multiple documents (or multiple versions of a document) are inserted into an index only once thereby minimizing the size of the index.

The method in [15] is most similar to the work proposed here. [15] uses winnowing [25] to break down documents into fragments. Fragments (and their respective alignments) are chosen to maximize redundancy thereby reducing the overall size of the index. Performance of the winnowing algorithm is optimized using hashing such that the comparison of two fragments is based on their respective hashing. Index sizes are reduced by as much as 60% when removing redundancy. However, textual methods rely on exact matching to reduce redundancy. The probability of seeing exact matches in medical time series is extremely low due to several environmental (e.g., noise, sensor displacement/placement, etc.) and physiological (e.g., differences in heart rates, weight, age, etc.) factors.

III. Method

The proposed algorithm takes in a stream of objects from a biomedical time-series signal. Objects are defined by the segmentation. This paper assumes two types of segmentation: segmentation by events and segmentation by sliding window. Event-based segmentations are domain specific. Examples include heartbeats of an ECG signal, a cycle in the arterial blood pressure, etc. For sliding window segmentation, segments are extracted by sliding a window along the time dimension, indexing each possible segment. Each slide (or translation) of the window is of D data points where $D \geq 1$ (this paper sets $D = 1$ for all datasets). A fixed window size is assumed for both types of segmentation.

On an input object u, the following four procedures are run:

1. Find the hash of u (H_u) using LSH
2. Search the global hash table for collisions V
3. Calculate $d_\upsilon = \|u - \upsilon\|_p \forall \upsilon \in V$
   - **a.** If $d_\upsilon < R$, assign u to υ (on multiple matches, assign randomly)
   - **b.** Else, create a new grouping g with hash H_u, add u to g, and add g to the global hash table

The LSH implementation using p-stable distributions is used as the hashing algorithm [12]. This algorithm uses the following hash function:

$$h_{a,b}(v) = \left\lfloor \frac{a \cdot v + b}{r} \right\rfloor, \quad (4)$$

where a is a randomized vector following a Gaussian distribution, b is a uniformly randomized vector, and r is a predefined constant. Using the properties of the p-stable distribution, the authors show that the probability of collision is calculated as:

$$p(c) = \int_0^r \frac{1}{c} f_p\left(\frac{t}{c}\right)\left(1 - \frac{t}{r}\right)\mathrm{d}x, \quad (5)$$

with c being the distance between two vectors. As can be seen by Equation 5, the probability of collision decreases monotonically as c increases.

Each grouping is represented by its initial object. Upon completion of the aggregation, each grouping's representative is placed into the aggregated index. LSH indexing is used in the experimentation for this paper. However, this work could be applied to other indexing techniques such as spatial trees. LSH was chosen for analysis due to LSH's sub-linear theoretical complexity.

Searches are constrained to only the aggregated index. A mapping table is kept between a grouping's representative object and all its contents. When a matching representative is found, all objects from its respective grouping are returned.

A. Computational Complexity

This paper assumes an optimal hashing algorithm with a sufficiently large table. Therefore, the complexity of the algorithm is dominated by pruning (or distance computations). As shown in [26], the number of distance computations per object is bounded by $O(n^{\rho})$, where $\rho = \frac{\ln 1/p_1}{\ln 1/p2}$ and n is the total number of objects. This yields a total runtime complexity of $O(n^{2\rho})$. In practice, the overall runtime is much lower as the search space is significantly reduced (i.e., the search space grows sub-linearly with respect to the number of objects). Therefore, the actual the number of groupings ($\hat{n}$) at any point in the algorithm is much less than n ($\hat{n} << n$). Therefore, we get a runtime complexity bounded by $O(n\hat{n}^{2\rho})$ where, in general, $\hat{n} << n$.

Memory is bounded by $O(n + nL)$. Notice that k is not included in the memory computation, because LSH creates L different hashes, each composed of k sub-hashes. A match is defined by two segments that match for at least one of the L hashes. A matching of one of the L hash functions is defined as a match of all of the k sub-hashes. Therefore, each of the k sub-hashes can be combined and hashed to a single integer. Hence, the overall storage required to store L LSH hash results is L. As stated earlier, the number of groupings $\hat{n}$ is less than the total number of objects. In practice, the algorithm will be bounded by $O(n + \hat{n}L)$.

IV. Experimental Setup

Four datasets composed of various types of biomedical signals are used in the analysis of this paper. The datasets are as follows:

1. MIMIC Database [27][28]. This dataset contains multiple channel recordings taken from patients in intensive care units. Electrocardiogram (QRS), arterial blood pressure (ABP), and fingertip plethysmograph (PLE) were used in the analysis.
2. MIT-BIH Arrhythmia Database (ECG) [29][28]. This dataset contains several 30-minute segments of two-channel ambulatory ECG recordings. These sample included arrhythmias of varying significance.
3. Gait Dynamics in Neuro-Degenerative Disease Database [30][31][28]. This dataset contains data gathered from force sensors placed under the foot. Healthy subjects as well as those with Parkinson's disease, Huntington's disease, and amyotrophic lateral sclerosis (ALS) were asked to walk while the data was recorded. Data includes 5-minute segments for each subject.

4. WALK [7]. This dataset contains a series of annotated recordings from a tri-axial accelerometer worn in a subject's pants pocket. Data was recorded while subjects travelled through the interior of a building.

The proposed aggregation algorithm is assessed using five experiments. The first experiment aggregates times series signals and displays the growth of groupings with respect to the number of objects with varying values of R. This experiment is run using both event-based segmentation and sliding window segmentation. All three channels of the MIMIC dataset were used for testing event-based segmentation. Segmentation was based on the dataset's annotations. For sliding window segmentation, the MIMIC database along with the GAIT and WALK datasets were used.

The second experiment tested the sensitivity and specificity of the proposed aggregation algorithm. The MITDB database was used for this experiment. The MITDB database is composed of well-annotated two-channel ECG. Time series signals from the MITDB database were aggregated using the proposed algorithm with event based segmentation. Each segment is initially labelled using the dataset's annotations (ground truth labels). Each grouping is labelled using its representative's ground truth label. Sensitivity and specificity are calculated by comparing each segment's true label to that of its grouping label. The MITDB dataset was used used in lieu of other datasets as those datasets do not have annotated class labels for the segmented objects.

The third experiment assesses the improvements of clustering when limiting the clustered elements to only grouping representatives. The Fast k-means algorithm [23] is run on both the aggregated and non-aggregated data. Only event-based segmentation of the MIMIC dataset was used. Sliding window segmentation was not assessed as it has been shown to be ineffective for clustering time series signals due to redundancy [32]. The number of iterations is used to demonstrate the performance improvements when using an aggregated signal.

The fourth experiment compares the size of salient indexes [7] to that of aggregated indexes. The MITDB, GAIT, and WALK datasets were used (as these were the same datasets used in [7]).

The fifth, and final experiment, tests the improvement of subsequence search when using aggregated indexes. The index was populated with the MITDB dataset along with two additional datasets. The MIT-BIH Noise Stress Test Database (NSTDB) [28][33] and the MIT-BIH ST Change Database (STDB) [28] were added to increase the overall size of the database. Both a standard LSH index along with an aggregated LSH index were composed. 100 random searches were performed while measuring the respective memory usage, precision and recall.

Each segment is normalized using the standard score normalization function:

$$\frac{X-\mu}{\sigma}, \quad (6)$$

No filtering was used unless explicitly noted. In general, proper filtering improves the results of machine learning and data mining tasks. However, this paper forgoes filtering processes to avoid any effects of filtering (both positive and negative). Each dataset used a fixed segment size across all experiments. Each dataset's parametrization is listed in Table I. As it is assumed that each dataset is cyclical, segment sizes were chosen to encapsulate one cycle. Cycle sizes do vary, however, the proposed algorithm is fairly resilient to the choice of segment size (as shown by the results).

V. Results

A. Sub-linear growth of groupings

Fig. 2 and Fig. 3 displays the growth of groupings with respect to the total number of segments. Fig. 2 displays this growth with the MIMIC dataset (including QRS, PLE, and ABP). The left side of the graph displays the growth for event based segmentation and the right displays results for sliding window segmentation. Fig. 3 displays the growth of groupings for the GAIT and WALK dataset with only sliding window segmentation as there are no annotations for segmentation for these two datasets. Three values for the radius were tested: $R = 2$, 3, and 4.

All datasets show sub-linear growth with respect to the total number of objects. Growth is decreased with an increased radius. The rate of growth appears to be similar for both sliding window segmentation and event-based segmentation. Note that sliding window segmentation contains a much larger number of segments than that of its sliding window counterpart.

The effects of the radius R vary across datasets. For example, the WALK dataset shown in Fig. 3 reduces the total number of segments by about one third for a radius of $R = 2$. However, a radius of $R = 3$ or $R = 4$ shows excellent sub-linear growth. This is caused by two factors. First, the WALK dataset is much smaller than all other datasets. As more data is collected, there is a high probability of seeing a pattern that was previously seen. Therefore, smaller datasets may not experience as much reduction as larger datasets. Second, the WALK dataset is the most variable dataset tested by this paper. Accelerometer data is inherently noisy especially when the accelerometer is not affixed to the body (as with the WALK dataset). The results in Fig. 3 demonstrates the algorithm's susceptibility to noise. This noise can be improved by increasing R as well as adding filtering.

Fig. 4 shows the effect of filtering the data before aggregation for the WALK dataset. A basic high pass filter is used by assigning each time point as the mean of a surrounding window of size 10 (approximately 0.2 seconds). A significant decrease in the cluster growth is observed. The choice of filter can either improve or hurt the results (in terms of sensitivity and specificity). For brevity, a complete analysis of proper filtering techniques is left to future work.

A summary of the reduction of each type of data is given in Fig. 5. MITDB, NSTDB, and STDB are excluded as they are all ECG (same as QRS for the MIMIC dataset).

B. Sensitivity and Specificity

Increasing the radius R for the proposed algorithm decreases the growth of groupings. However, this increase in R has a cost. Increasing the radius for an acceptable match increases the probability of incorporating incorrect matches to a grouping. Fig. 6 shows the change to sensitivity and specificity with varying values of R. As expected, an increase in R degrades performance of the proposed algorithm. A significant drop off is observed for a radius of $R > 4$. Hence, all other experiments used a value of $R \leq 4$.

The effect of R on the sensitivity and specificity may change depending on the dataset. However, only the MITDB dataset was used in this set of experiments as no other datasets included in this paper have appropriate annotations for calculating performance measures (i.e., only MITDB contains class labels).

C. Clustering on aggregated indexes

Fig. 7 displays the number of iterations of the fast k-means algorithm in [23] for the MIMIC dataset. The number of iterations is shown as a function of the total number of segments. Only event-based segmentation was used in this experiment as its been shown that clustering using sliding window segmentation has little to no meaning [32].

As shown by Fig. 7, the number of cluster iterations for non-aggregated indexes exhibits growth that is greater than linear. When clustering on aggregated indexes, a seemingly linear (or sub-linear) growth is observed. Fig. 6 demonstrates that the aggregation results in only a small degradation in sensitivity and specificity with small values of R. Data mining on aggregated indexes will, therefore, make algorithms more tractable while ensuring a high level of accuracy.

D. Comparison of Salient and Aggregated Indexes

Table II displays the index sizes of Salient Segmentation versus those from the proposed aggregated indexing algorithm. Both algorithms achieve similar results. However, Salient Segmentation is lossy, meaning much of the time series signals are excluded from the database. When using aggregated indexing, only the groupings are added to the index (search space). However, any segment belonging to a grouping can be associated to the grouping through a mapping table. Hence, the database may still contain all segments from the original signal resulting in lossless storage of the signal.

E. Comparison of LSH on aggregated and non-aggregated indexes

Table III displays memory and precision for both an LSH index and an LSH aggregated index. Precision and recall are calculated by the annotation labelling. Precision for LSH alone is marginally better than LSH with an aggregated index. However, the average memory for each query is approximately 15 times less for the aggregated index. As the

growth of groupings is less than linear with respect to the number objects, memory improvements will increase with larger databases.

The aggregated index has a recall that is slightly better than LSH alone. An improvement in recall is largely due to distant matches (those that have a distance of approximately R to the query segment) that are less likely to have colliding hashes with the query segment. For example, if a query segment has two distant matches, where both distant matches are extremely close to each other, then it is possible that the query segment will have a matching hash with just one of the distant matches. In addition, the two distant matches will probably have several matching hashes resulting in the two matches being aggregated to the same grouping. Therefore, LSH alone will miss one of the distant matches, while LSH on an aggregated index will likely match to the grouping and retrieve both distant matches.

Precision is slightly lower for the aggregated index as any distant groupings (distance of approximately R to the query segment) may contain objects that are greater than R from the query object. However, the effects of such groupings appear to be minimal.

VI. Conclusion

This paper presented an algorithm for constructing an aggregated medical time series index. The algorithm is based on the observation that medical time series signals are often composed of similar and repetitive cycles. Therefore, the size of the index can be significantly reduced by only including unique patterns. For the purpose of this paper, two patterns are considered to be the same if the corresponding Euclidean distance is less than R. Euclidean distance has been shown to have high discriminatory power for small values of R [10]. Hence, small neighborhoods with radius R tend to consist of a single class.

The proposed algorithm runs similar to the Online Facility Location problem [13]. An input segment is hashed using LSH. If this hash collides with a facility (e.g., an existing grouping), the segment is assigned to that facility. If no collisions occur, a new facility (or grouping) is created and inserted into the global hash table. Each grouping is represented by its first resident and only the representative is added to the index.

This paper showed that the total number of groupings created by the algorithm grows sub-linearly with respect to the total number of objects for many medical time series signals. Aggregated indexes are shown to significantly improve performance of searching and mining medical time series with little degradation in the quality of results.

Acknowledgments

The authors would like to thank Alexandr Andoni et. al. for providing their implementation of LSH presented in [12]. This publication was partially supported by Grant Number T15 LM07356 from the NIH/National Library of Medicine Medical Informatics Training Program.

References

1. Suh M, Chen C, Woodbridge J, Tu M, Kim J, Nahapetian A, Evangelista L, Sarrafzadeh M. A remote patient monitoring system for congestive heart failure. Journal of medical systems. 2011:1–15. [PubMed: 20703590]
2. Suh, M.; Evangelista, L.; Chen, V.; Hong, W.; Macbeth, J.; Nahapetian, A.; Figueras, F.; Sarrafzadeh, M. World of Wireless Mobile and Multimedia Networks (WoWMoM), 2010 IEEE International Symposium on a. IEEE; 2010. Wanda b.: Weight and activity with blood pressure monitoring system for heart failure patients; p. 1-6.
3. Liszka K, Mackin M, Lichter M, York D, Pillai D, Rosenbaum D. Keeping a beat on the heart. Pervasive Computing, IEEE. 2004; 3(4):42–49.
4. Agarwal S, Lau C. Remote health monitoring using mobile phones and web services. Telemedicine and e-Health. 2010; 16(5):603–607. [PubMed: 20575728]
5. Varshney, U. Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2009). IEEE; 2009. A framework for wireless monitoring of mental health conditions; p. 5219-5222.
6. Faloutsos, C.; Ranganathan, M.; Manolopoulos, Y. Fast subsequence matching in time-series databases. Proceedings of the 1994 ACM SIGMOD international conference on Management of data; ACM; New York, NY, USA. 1994. p. 419-429.ser. SIGMOD '94.
7. Woodbridge, J.; Lan, M.; Sarrafzadeh, M.; Bui, A. Healthcare Informatics, Imaging and Systems Biology (HISB), 2011 First IEEE International Conference on. IEEE; 2011. Salient segmentation of medical time series signals; p. 1-8.
8. Beyer K, Goldstein J, Ramakrishnan R, Shaft U. When is "nearest neighbor" meaningful? Database TheoryICDT99. 1999:217–235.
9. Francois D, Wertz V, Verleysen M. The concentration of fractional distances. Knowledge and Data Engineering, IEEE Transactions on. 2007; 19(7):873–886.
10. Bennett, K.; Fayyad, U.; Geiger, D. Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining. ACM; 1999. Density-based indexing for approximate nearest-neighbor queries; p. 233-243.
11. Houle, M.; Kriegel, H.; Kröger, P.; Schubert, E.; Zimek, A. Scientific and Statistical Database Management. Springer; 2010. Can shared-neighbor distances defeat the curse of dimensionality?; p. 482-500.
12. Datar, M.; Immorlica, N.; Indyk, P.; Mirrokni, V. Proceedings of the twentieth annual symposium on Computational geometry. ACM; 2004. Locality-sensitive hashing scheme based on p-stable distributions; p. 253-262.
13. Meyerson, A. Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on. IEEE; 2001. Online facility location; p. 426-431.
14. Herscovici M, Lempel R, Yogev S. Efficient indexing of versioned document sequences. Advances in Information Retrieval. 2007:76–87.
15. Zhang, J.; Suel, T. Proceedings of the 16th international conference on World Wide Web. ACM; 2007. Efficient search in large textual collections with redundancy; p. 411-420.
16. Broder A, Eiron N, Fontoura M, Herscovici M, Lempel R, McPherson J, Qi R, Shekita E. Indexing shared content in information retrieval systems. Advances in Database Technology-EDBT 2006. 2006:313–330.
17. Shieh J, Keogh E. isax: disk-aware mining and indexing of massive time series datasets. Data Mining and Knowledge Discovery. 2009; 19(1):24–57.
18. Keogh E, Chakrabarti K, Pazzani M, Mehrotra S. Dimensionality reduction for fast similarity search in large time series databases. Knowledge and information Systems. 2001; 3(3):263–286.
19. Cai, Y.; Ng, R. Proceedings of the 2004 ACM SIGMOD international conference on Management of data. ACM; 2004. Indexing spatio-temporal trajectories with chebyshev polynomials; p. 599-610.

20. Weber, R.; Schek, H.; Blott, S. Proceedings of the International Conference on Very Large Data Bases. IEEE; 1998. A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces; p. 194-205.
21. Gionis, A.; Indyk, P.; Motwani, R. Similarity search in high dimensions via hashing; Proceedings of the International Conference on Very Large Data Bases; 1999. p. 518-529.
22. Woodbridge, J.; Mortazavi, B.; Bui, A.; Sarrafzadeh, M. Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC 2012). IEEE; 2012. High performance biomedical time series indexes using salient segmentation. p. To Appear.
23. Elkan C. Using the triangle inequality to accelerate k-means. Proc. 20th Intl Conf. Machine Learning (ICML 03). 2003; 20(1):147.
24. Mueen, A.; Keogh, E.; Zhu, Q.; Cash, S.; Westover, B. Exact discovery of time series motifs; Proc. of 2009 SIAM International Conference on Data Mining: SDM; 2009. p. 1-12.
25. Schleimer, S.; Wilkerson, D.; Aiken, A. Proceedings of the 2003 ACM SIGMOD international conference on Management of data. ACM; 2003. Winnowing: local algorithms for document fingerprinting; p. 76-85.
26. Indyk, P.; Motwani, R. Proceedings of the thirtieth annual ACM symposium on Theory of computing. ACM; 1998. Approximate nearest neighbors: towards removing the curse of dimensionality; p. 604-613.
27. Moody, G.; Mark, R. Computers in Cardiology 1996. IEEE; 1996. A database to support development and evaluation of intelligent intensive care monitoring; p. 657-660.
28. Goldberger A, Amaral L, Glass L, Hausdorff J, Ivanov P, Mark R, Mietus J, Moody G, Peng C, Stanley H. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. Circulation. 2000; 101(23):e215–e220. [PubMed: 10851218]
29. Mark R, Moody G. Mit-bih arrhythmia database directory. Cambridge: Massachusetts Institute of Technology. 1988
30. Hausdorff J, Mitchell S, Firtion R, Peng C, Cudkowicz M, Wei J, Goldberger A. Altered fractal dynamics of gait: reduced stride-interval correlations with aging and huntington's disease. Journal of Applied Physiology. 1997; 82(1):262. [PubMed: 9029225]
31. Hausdorff J, Lertratanakul A, Cudkowicz M, Peterson A, Kaliton D, Goldberger A. Dynamic markers of altered gait rhythm in amyotrophic lateral sclerosis. Journal of applied physiology. 2000; 88(6):2045–2053. [PubMed: 10846017]
32. Keogh E, Lin J. Clustering of time-series subsequences is meaningless: implications for previous and future research. Knowledge and Information Systems. 2005; 8(2):154–177.
33. Moody G, Muldrow W, Mark R. A noise stress test for arrhythmia detectors. Computers in Cardiology. 1984; 11(3):381–384.

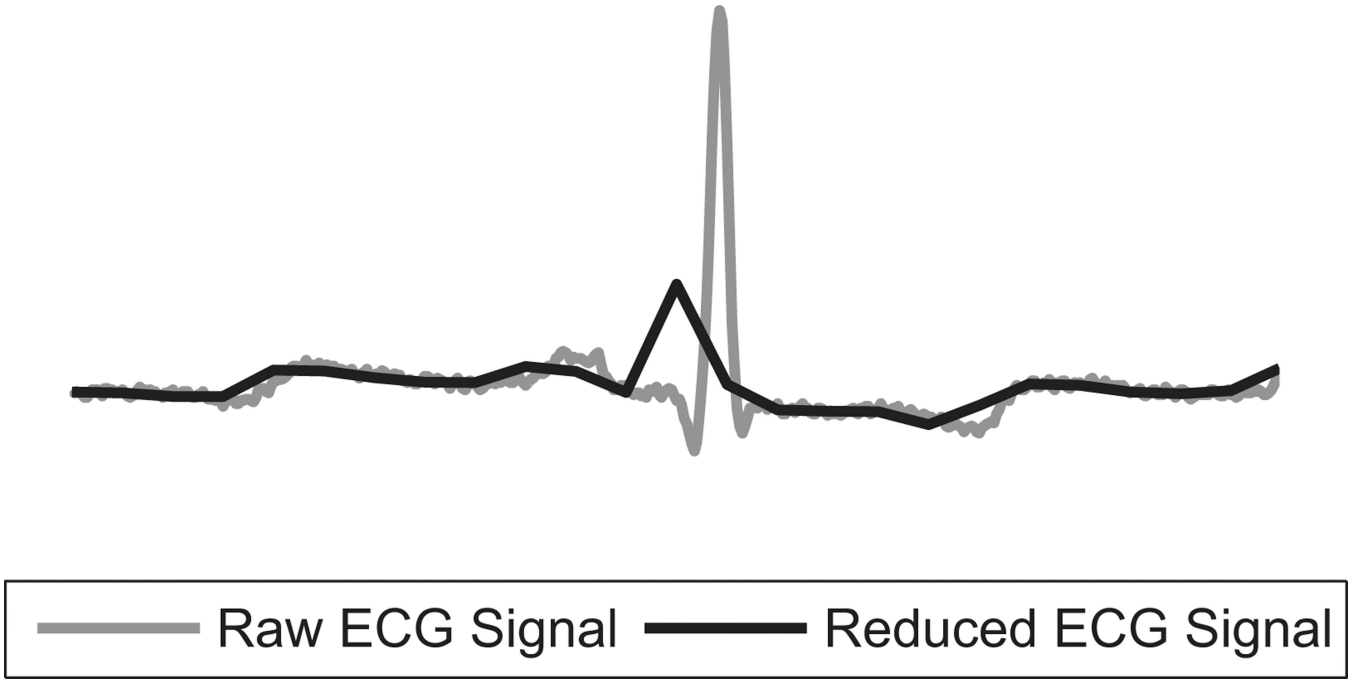

Fig. 1.
Shows the Piecewise Aggregate Approximation (PAA) representation of an ECG signal composed of 512 data points reduced to 25 dimensions.

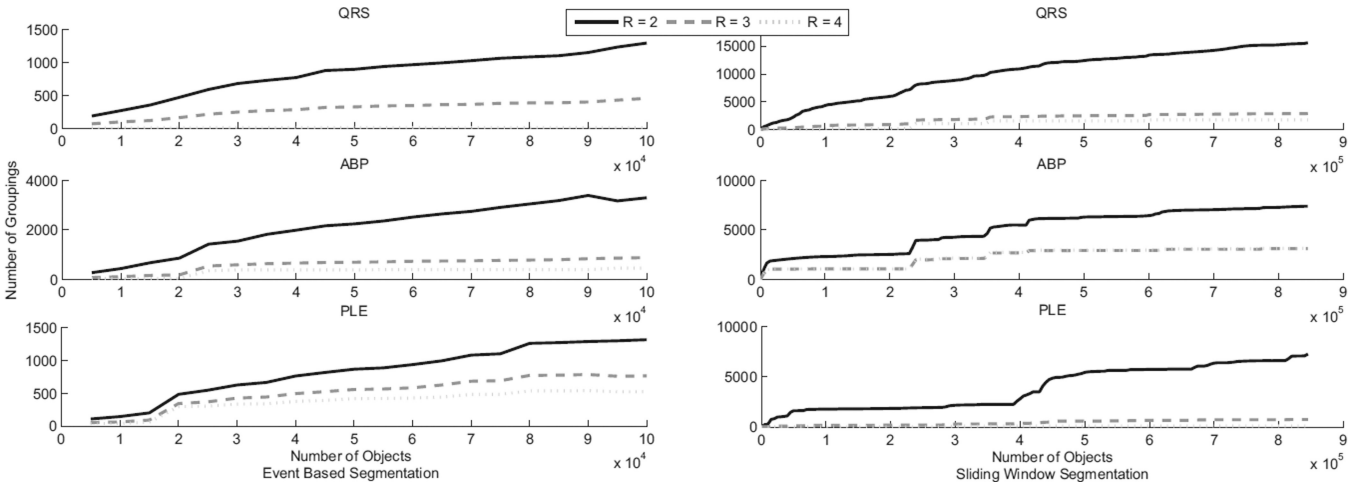

Fig. 2.

Displays the growth of clusters with increasing number of objects for the MIMIC dataset (including QRS, PLE, and ABP). Both event-based segmentation (left) and sliding window segmentation are shown (right). Each type of data shows sub-linear growth with a decreasing rate of growth with increasing *R*.

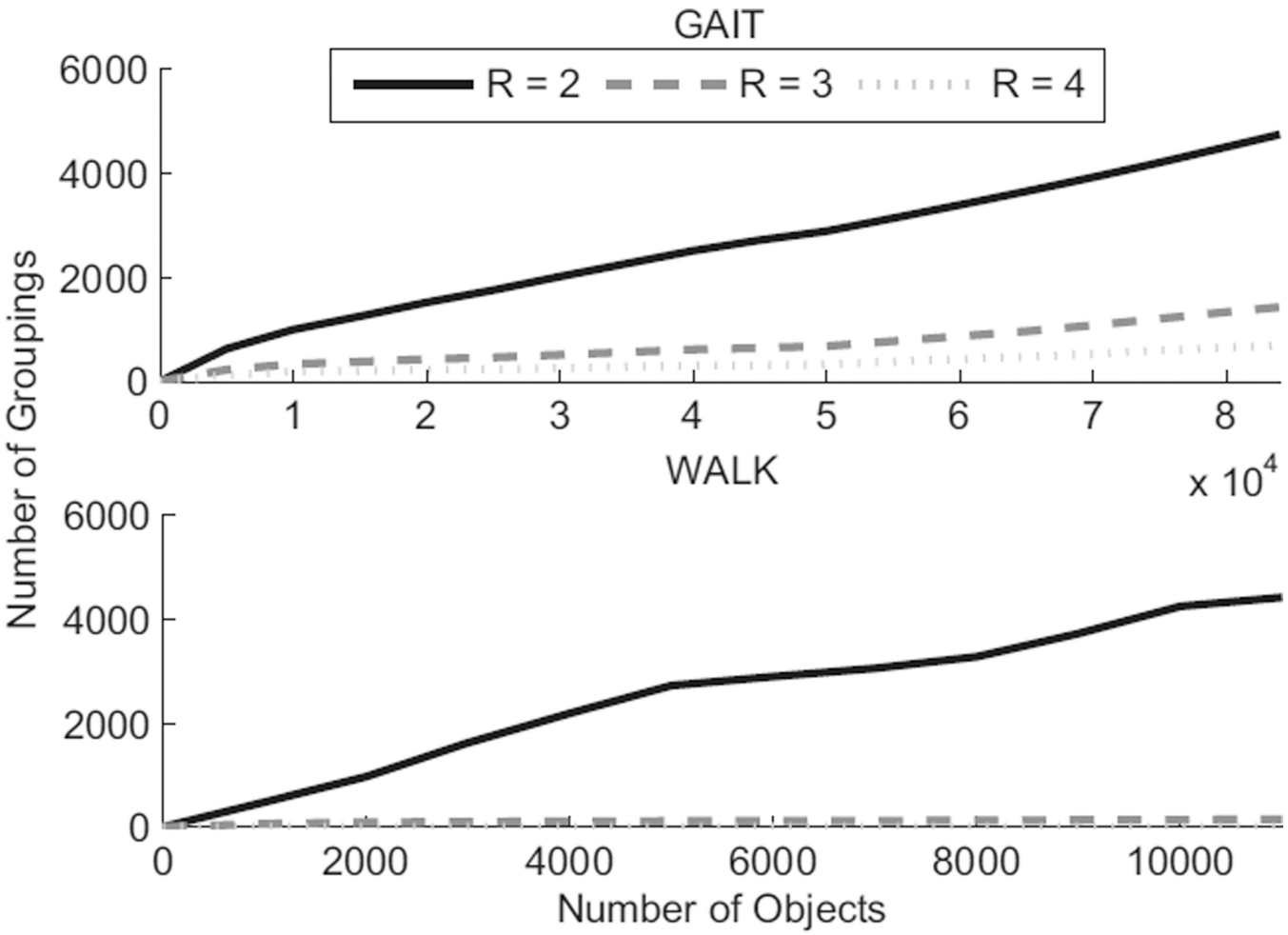

Fig. 3.

Displays the growth of clusters with increasing number of objects for the GAIT and WALK datasets. Only sliding window segmentation is used. Both datasets show sub-linear growth with a decreasing rate of growth with increasing *R*.

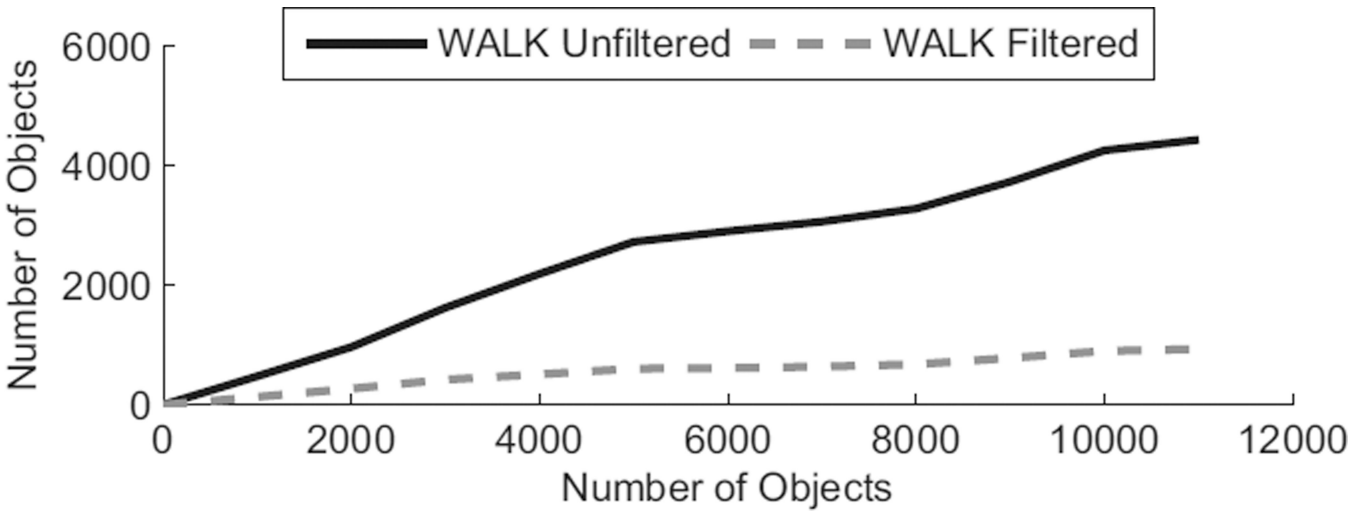

Fig. 4.

Displays the growth of clusters with increasing number of objects for the WALK dataset with a fixed $R = 2$. Results are shown with the WALK dataset filtered and unfiltered. Filtering is shown to slow object growth.

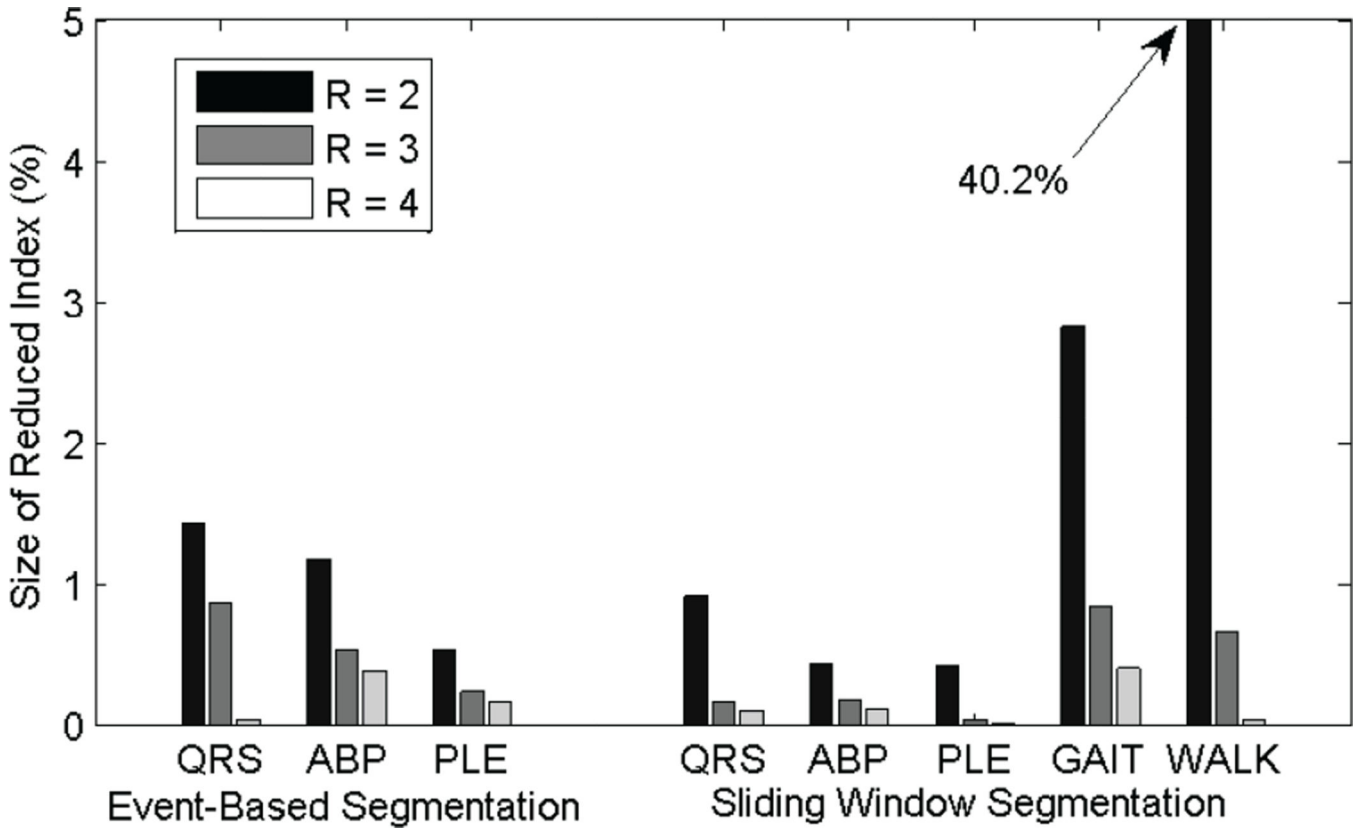

Fig. 5.

Displays the aggregated index sizes for all types of data with $R = 2$, 3 and 4 as compared to original index size.

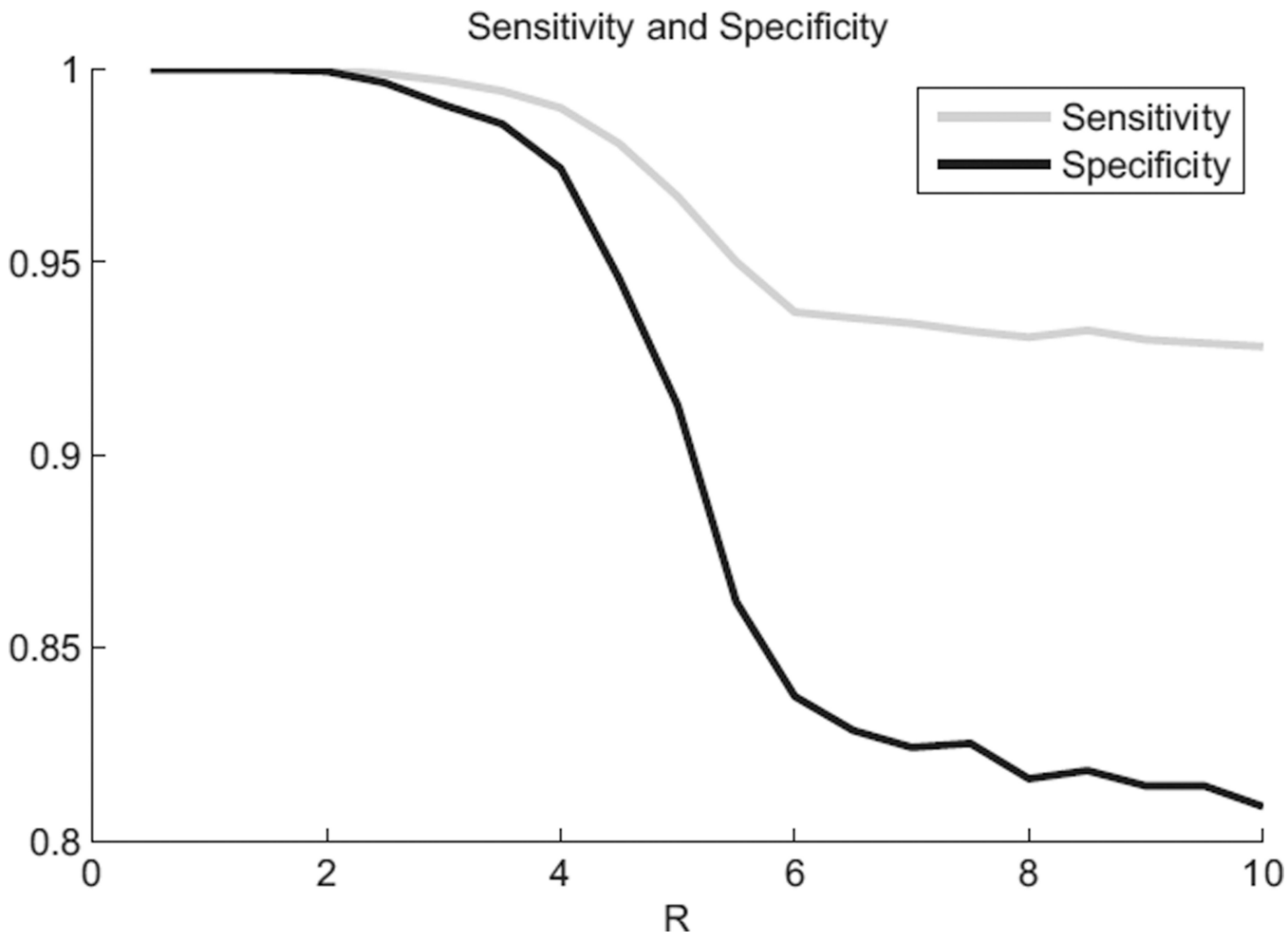

Fig. 6.

Sensitivity and Specificity with varying radius (R) for the MITDB dataset. Both specificity and sensitivity drop significantly with $R > 4$.

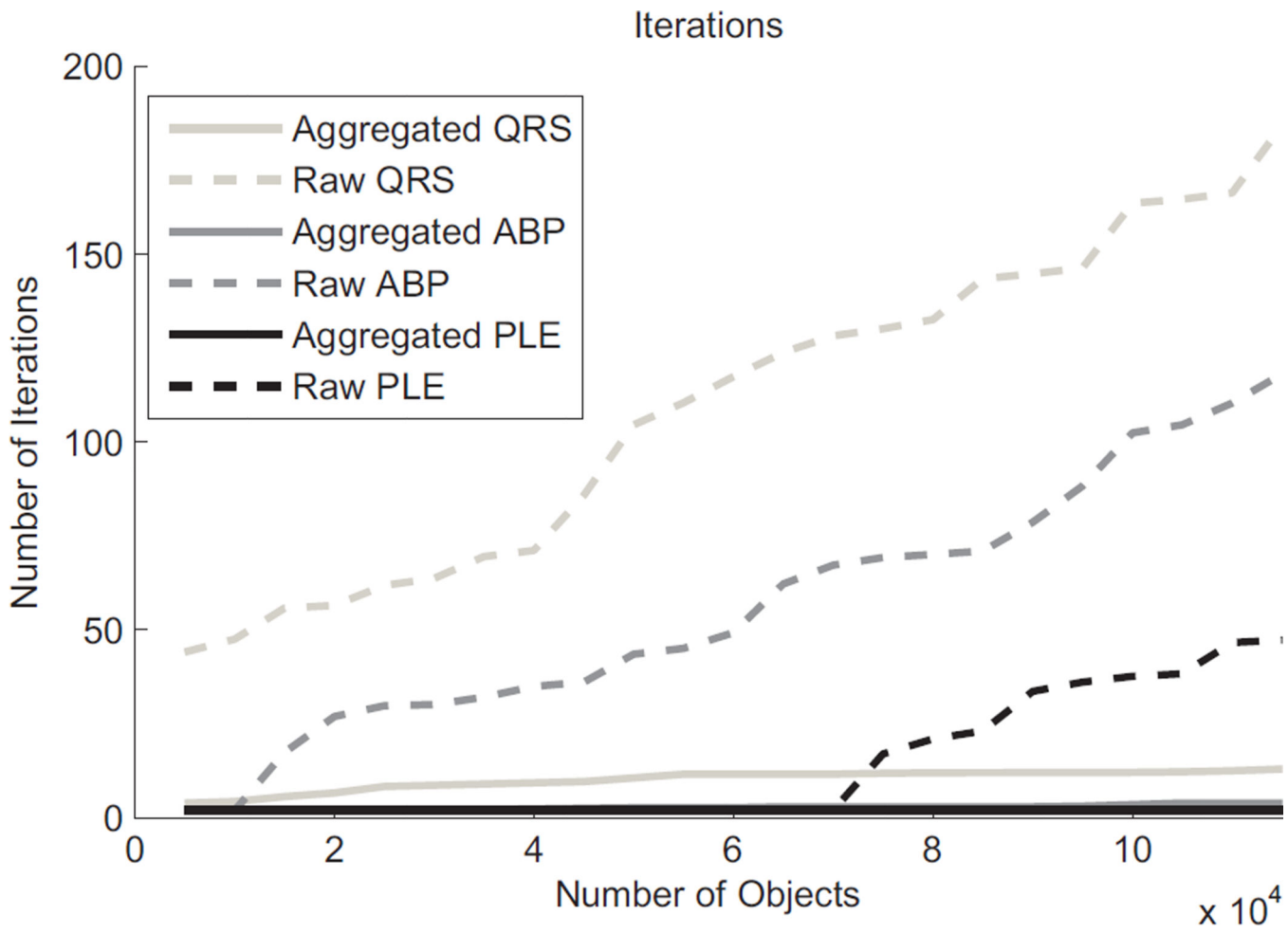

Fig. 7.

Displays the number of iterations when clustering using the fast k-means algorithms [23]. Results for both aggregated and raw data is shown for the MIMIC dataset.

TABLE I

Dataset Parameters

Dataset	Segment Size	Sampling Rate
MIMIC	128	125Hz
MITDB	512	360Hz
GAIT	512	300Hz
WALK	60	50Hz
NSTDB	512	360Hz
STDB	512	360Hz

TABLE II

Sizes of Salient and Aggregated Indexes

Dataset	Salient Segmentation	Aggregated Index		
		R **= 2**	***R*** **= 3**	***R*** **= 4**
MITDB	.4%	11.1%	1.8%	1.0%
GAIT	1.8%	5.6%	1.7%	.82%
WALK	1.6%	40.2%	1.3%	1.0%

TABLE III

Memory, Precision and Recall for Aggregated Index as compared to LSH alone

Non-Aggregated Index		Aggregated Index		
Precision	**Memory**	**Precision**	**Memory**	**Recall**
.9523	9424kB	.9428	691kB	+5%