Truly Scalable Data Series Similarity Search

Karima Echihabi
Supervised by: Prof. Themis Palpanas and Prof. Houda Benbrahim
IRDA, Rabat IT Center,
ENSIAS, Mohammed V Univ.
karima.echihabi@gmail.com

ABSTRACT

Data series are widely used in numerous domains. Analyzing this data is important for a variety of real-world applications and has been extensively studied over the past 25 years. At the core of the analysis task lies a classic algorithm called similarity search. A number of approaches have been proposed to support similarity search over massive data series collections. The results of two comprehensive data series experimental evaluations form the foundations of our future work, which will lead to the development of a novel index that can efficiently support both exact and approximate data series similarity search, as well as progressive query answering with bound guarantees. Based on the insights gained from the exhaustive study of the related work, the new index will surpass the state-the-art approximate and exact techniques both in performance and accuracy.

1. INTRODUCTION

A data series is a sequence of ordered real values[1]. Data series are omnipresent in various domains from science and engineering to business and medicine [35]. The proliferation of IoT technologies is also heavily contributing to the explosive growth of data series collections to the order of terabytes [38]. A common subroutine to data series analytical tasks is a classic algorithm called similarity search. Therefore the research community has extensively studied the development of efficient similarity search algorithms for data series. The similarity search algorithm for data series returns the set of candidate data series in a collection that is similar to a given query series. This algorithm is often reduced to the nearest neighbor problem where data series are represented as data points in multidimensional space and their (dis)similarity is evaluated using a distance function.

[1]The order attribute can be angle, mass, frequency, position, or time [39]. When the order is based on time, the sequence is called a *time series*. The terms *data series*, *time series* and *sequence* are used interchangeably.

Proceedings of the VLDB 2019 PhD Workshop, August 26th, 2019. Los Angeles, California.

Although a data series can be represented as a vector in high dimensional space, conventional vector-based approaches are not adapted for two reasons: (a) they cannot scale to thousands of dimensions; and (b) they do not exploit the correlation among dimensions typical for data series.

Similarity search methods can either return exact or approximate answers. Exact methods are costly while approximate methods offer better efficiency at the expense of losing some accuracy. We call approximate the methods that do not provide any guarantees on the results *ng*-approximate, and those that provide guarantees on the approximation error, δ-ϵ-approximate methods, where ϵ is the approximation error and δ, the probability that ϵ will not be exceeded.

A plethora of similarity search methods have been published by the community including techniques designed for generic vectors [24, 23, 7, 14, 46, 20, 25, 44, 5, 36] and those specific to data series [3, 41, 27, 42, 47, 34, 33, 40, 15, 9, 43, 45, 10, 49, 30].

This work aims to propose a novel index that will support progressive query answering with probabilistic guarantees. We describe the related work, succinctly report the results of our extensive experimental evaluation of *exact* methods [17], and give a glimpse of some very interesting results from an ongoing experimental study focused on *approximate* methods [18]. Finally, we present our future work directions.

In our work, we focus on the problem of *whole matching similarity search in collections with a very large number of data series*, i.e., similarity search that produces approximate or exact results, by calculating distances on the whole (not a sub-) sequence. This is a very popular problem that lies at the core of several other algorithms, and is important for many applications in various domains in the real world [17].

2. RELATED WORK

Similarity search involves finding a set of data series in a collection that are similar to a query according to some definition of sameness. A common abstraction is to consider the query and candidate data series as points in a metric space and evaluating the sameness (or difference) using the euclidean distance.

To develop efficient similarity search algorithms on massive datasets, two major costs need to be optimized: the cost of accessing data on disk (I/O) and the cost of comparing the query to candidates (CPU cost). Typically, the first cost is reduced by using summarization techniques that map the high-dimensional data to a lower-dimensional space, while the second cost is optimized with sophisticated data structures and search algorithms. Several similarity search meth-

ods have been proposed in the literature supporting either exact search [23, 7, 46, 20, 27, 42], approximate search [24, 25, 44, 5, 36], or both [14, 43, 45, 10, 49, 47, 30, 34].

In the following section, we provide a succinct description of the state-of-the-art similarity search methods and the summarizations techniques on which they are based.

1. Summarization Techniques. The Discrete Fourier Transform (DFT) [19] decomposes a data series into frequency coefficients, a subset of which represents a summarization of the data series.

The Discrete Haar Wavelet Transform (DHWT) [12] transforms a data series using Haar wavelets into a hierarchical representation.

Random projections map the raw high dimensional data into a lower dimensional space using a random matrix while preserving pairwise distances within a distortion threshold [26].

The Piecewise Aggregate Approximation (PAA) [28] and *Adaptive Piecewise Constant Approximation* (APCA) [11] techniques divide a data series into segments (of equal and arbitrary length, respectively) and approximate each segment with the mean of the points that belong to it. The *Extended Adaptive Piecewise Approximation* (EAPCA) [45] method enhance APCA by approximating each segment with the standard deviation in addition to the mean.

Symbolic Aggregate Approximation (SAX) [32] first approximates data series using PAA, then discretizes the PAA values into a compact binary representation.

Symbolic Fourier Approximation (SFA) [43] first transforms a data series into DFT coefficients, which are then approximated using a succinct symbolic approximation.

Optimized Product Quantization (OPQ) [21] applies a linear transformation on the data series to decorrelate it, then applies on it a product quantizer [25].

2. Exact Similarity Search Methods. Below, we briefly describe algorithms that produce exact results.

The R*-tree [7] is a height-balanced spatial index that organizes data into a hierarchy of nested overlapping rectangles. Search returns all entries in leaves whose rectangle contains the query.

The M-tree [14] is a multidimensional index for metric space which partitions the data using hyper-spheres based on their relative distances. The search algorithm uses the triangular inequality to prune data.

The VA+file [20] is an improvement of the VA-file [46]. It creates a filter file containing summarizations of the raw data. Search uses the filter file to prune candidates based on a lower bounding distance. For efficiency reasons, we modified the VA+file to use the DFT transform instead of the Karhunen–Loève transform (KLT).

Stepwise [27] represents the data space in a multi-level hierarchical representation using DHWT summarizations. Search transforms a query into DHWT and filters out candidate based on upper and lower bounding distances.

The SFA method [43] builds a trie with SFA summarizations of the data. During query answering, a lower bounding distance is used to prune out candidates.

The UCR Suite [42] is an optimized sequential scan algorithm for exact matching that we consider as a baseline for performance comparisons.

The DSTree [45] index dynamically segments data using EAPCA. During search, it uses a lower bounding distance to prune the search space.

iSAX2+ [10] is a bulk-loading index based on SAX. During search, a query is represented with SAX symbols and the candidates contained in non-pruned leaves are further refined using the euclidean distance. Pruning uses a lower bounding distance.

ADS+ [49] is the first adaptive data series index based on SAX. It starts with a minimal tree structure containing only summarizations, and then adds in the raw data to leaves during query answering. It supports a number of search options, in particular SIMS, a skip-sequential algorithm.

In addition to exact search, the MTree, SFA trie, DSTree, iSAX2+ and ADS+ also support ng-approximate search.

3. Approximate Similarity Search Methods. We now present the three most popular approximate methods.

SRS [44] belongs to the family of LSH methods, inspired by the randomized algorithm in [24]. It uses random projections to build a scalable index and supports approximate search with theoretical guarantees.

IMI [21, 5] is an inverted index based on OPQ. A query is answered by returning all points corresponding to the corresponding entries in the inverted index. Returned results are ng-approximate.

HNSW [36] is an in-memory neighborhood graph exploiting the Voronoi Diagram and the Delaunay Triangulation. A greedy search returns the best candidates with high empirical accuracy but no formal theoretical guarantees.

3. PROPOSED WORK

We propose a novel data series index that can answer progressive similarity search queries with strong probability guarantees. In addition to this unique functionality, it also leverages the strengths and addresses the weaknesses of the state-of-the-art approximate and exact techniques.

Completed Work. We outline the main contributions in [17]: (i) we provide a formal problem definition for data series similarity search, unifying conflicting terminology from different research communities; (ii) we present a survey of the state-of-the-art data series similarity search techniques (Table 1 summarizes each technique according to our definitions); and (iii) we conduct an extensive experimental evaluation for the efficiency of data series *exact* similarity search.

The study assessed ten state-of-the-art methods under the same experimental framework. To guard against implementation bias, we used a large number of comparison criteria including implementation-independent ones, and we reimplemented from scratch four methods that were not available in C/C++. Our implementations largely outperform the original ones both in time and space, thus enriching the landscape of data series similarity search methods.

In an effort to make our results reproducible and support future research in the area, we share with the community a public archive containing all source codes, datasets, queries, results, and plots [1].

Based on the results of our study, we draw up recommendations to help users decide the best approach for their problem. Figure 1 shows a decision matrix given a typical hardware and query workload. The VA+file is particularly well-suited for long series in-memory while for shorter series, the DSTree is the best contender on disk, and iSAX2+ is the winner in-memory.

Table 1: Similarity search methods [17]

		Matching Accuracy				Matching Type		Representation		Implementation	
		exact	ng-appr.	ϵ-appr.	δ-ϵ-appr.	Whole	Subseq.	Raw	Reduced	Original	New
Indexes	ADS+	[49]	[49]			✓			iSAX	C	
	DSTree	[45]	[45]			✓			EAPCA	Java	C
	iSAX2+	[10]	[10]			✓			iSAX	C#	C
	M-tree	[14]		[13]	[13]	✓		✓		C++	
	R*-tree	[7]				✓			PAA	C++	
	SFA trie	[43]	[43]			✓	✓		SFA	Java	C
	VA+file	[20]				✓			DFT	MATLAB	C
Other	UCR Suite	[42]					✓	✓		C	
	MASS	[48]					✓		DFT	C	
	Stepwise	[27]				✓			DHWT	C	

We present an elaborate discussion based on the deep insights gained about the data series similarity search problem. The main points are the following:

1. Unexpected results confirmed the importance of careful parameter tuning, hardware setup, implementation framework, and workload selection. In particular, Stepwise and ADS+ performed below our expectations while our optimized implementations of the DSTree and the VA+file helped bring them back to the spotlight. Moreover, our carefully crafted experiments identified optimal parameters that were different than the ones published in the original papers. Another important finding was that, unlike what was originally believed [29], the tightness of the lower bound of a given method does not alone predict its performance. In fact, it is of paramount importance to consider other factors such as the hardware platform and the the clustering quality of a an index.

2. A better understanding of current approaches helped pinpoint interesting avenues for improvement, in particular identifying the methods that would most benefit from modern hardware. For instance: (i) the DSTree is a very good candidate for parallelization as its index building is over 85% CPU cost; (ii) the performance of Stepwise can be significantly improved with a redesign of the physical storage and the use of modern hardware as the total cost of query answering is 50%-98% CPU; (iii) ADS+ can be enhanced with the use of asynchronous I/O to overcome the expensive random I/O incurred with each skip.

3. Although index building with iSAX-based indexes is much faster than with the DSTree, the latter achieves a better clustering as it adapts to the data distribution.

4. Choosing between an index scan and a serial scan is not a trivial decision. In fact, access path selection is an optimization problem that depends on a variety of factors including hardware, query pruning ratio, data characteristics, the accuracy of a summarization and the efficacy of the clustering provided by an index.

Work In Progress. Our current work involves an experimental study that evaluates data series *approximate* similarity search, both in-memory and on-disk [18]. It differs from other experimental studies which focused on the efficiency of exact search [17], the accuracy of dimensionality reduction techniques and similarity measures for classification tasks [29, 16, 6], or in-memory data [31, 4].

Our results show that some strikingly simple modifications to existing exact methods enable them to answer δ-ϵ-approximate queries and have excellent empirical performance. In fact, extensive experiments on large synthetic and real datasets, including the two largest real datasets publicly

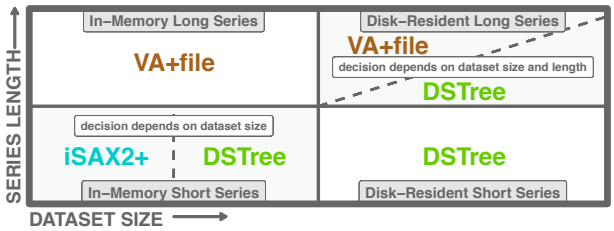
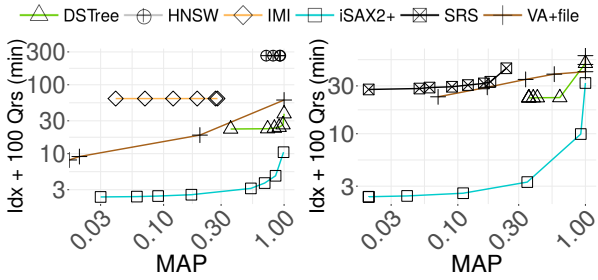


Figure 1: Recommendations [17] (Indexing and answering 10K exact synthetic queries on a hard-drive machine)



(a) 100-NN (ng) **(b) 100-NN ($\delta\epsilon$)**

Figure 2: Efficiency vs. accuracy of approximate search [18] (Dataset = Sift25GB, Data series length = 128)

available, demonstrate that the extended techniques compete in memory and outperform on disk the state-of-the-art approximate techniques from the vector indexing community both in accuracy and efficiency. Figure 2 summarizes results for 100-NN queries on an in-memory 25GB data collection extracted from the Sift dataset [2]. We observe that our modifications to iSAX2+ enable it to outperform the popular ng-approximate search approaches HNSW and IMI (Faiss implementation) and the state-of-the-art LSH method SRS, in terms of combined indexing and query answering costs. We measure accuracy using MAP, a popular metric in the information retrieval literature [37, 8].

Future Work. Inspired by the insights gained from our two experimental studies on the the inner workings of the different indexing approaches and the effectiveness of their design choices, the key future direction for our work is the design and development of a novel data series index that will outperform the state-of-the-art approximate and exact

techniques. The new index will also support progressive query answering [22] with probability guarantees, so as to further enable interactive exploration tasks on very large data series collections.

4. CONCLUSIONS

The goal of this thesis is to develop a new index that will support approximate and exact search, and progressive query answering with probability guarantees. The first step in this direction was to thoroughly assess the state-of-the-art [17, 18]. We describe the lessons learned from two extensive experimental evaluations and outline our future work.

References

[1] Lernaean Hydra Archive. `http://www.mi.parisdescartes.fr/~themisp/dsseval/`, 2018.

[2] TEXMEX Datasets for Approximate Nearest Neighbor Search. `http://corpus-texmex.irisa.fr/`, 2018.

[3] R. Agrawal, C. Faloutsos, and A. Swami. Efficient similarity search in sequence databases. pages 69–84, 1993.

[4] M. Aumüller, E. Bernhardsson, and A. Faithfull. Ann-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms. In *SISAP*, pages 34–49, 2017.

[5] A. Babenko and V. Lempitsky. The inverted multi-index. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(6):1247–1260, June 2015.

[6] A. J. Bagnall, J. Lines, A. Bostrom, J. Large, and E. J. Keogh. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Min. Knowl. Discov.*, 31(3):606–660, 2017.

[7] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger. The R*-tree: an efficient and robust access method for points and rectangles. In *ICMD*, pages 322–331. ACM, 1990.

[8] C. Buckley and E. M. Voorhees. Evaluating evaluation measure stability. In *SIGIR*, pages 33–40. ACM, 2000.

[9] A. Camerra, T. Palpanas, J. Shieh, and E. J. Keogh. iSAX 2.0: Indexing and Mining One Billion Time Series. In *ICDM*, pages 58–67. IEEE Computer Society, 2010.

[10] A. Camerra, J. Shieh, T. Palpanas, T. Rakthanmanon, and E. J. Keogh. Beyond one billion time series: indexing and mining very large time series collections with iSAX2+. *Knowl. Inf. Syst.*, 39(1):123–151, 2014.

[11] K. Chakrabarti, E. Keogh, S. Mehrotra, and M. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. *ACM TODS*, 27(2):188–228, June 2002.

[12] K.-P. Chan and A. W.-C. Fu. Efficient time series matching by wavelets. In *ICDE*, pages 126–133, 1999.

[13] P. Ciaccia and M. Patella. PAC Nearest Neighbor Queries: Approximate and Controlled Search in High-Dimensional and Metric Spaces. In *ICDE*, pages 244–255, 2000.

[14] P. Ciaccia, M. Patella, and P. Zezula. M-tree: An Efficient Access Method for Similarity Search in Metric Spaces. In *VLDB*, pages 426–435, 1997.

[15] R. Cole, D. E. Shasha, and X. Zhao. Fast window correlations over uncooperative time series. In *SIGKDD*, pages 743–749, 2005.

[16] H. Ding, G. Trajcevski, P. Scheuermann, X. Wang, and E. Keogh. Querying and mining of time series data: experimental comparison of representations and distance measures. *PVLDB*, 1(2):1542–1552, 2008.

[17] K. Echihabi, K. Zoumpatianos, T. Palpanas, and H. Benbrahim. The Lernaean Hydra of Data Series Similarity Search: An Experimental Evaluation of the State of the Art. *PVLDB*, 12(2):112–127, 2018.

[18] K. Echihabi, K. Zoumpatianos, T. Palpanas, and H. Benbrahim. The Return of the Lernaean Hydra: An Experimental Evaluation of Data Series Approximate Similarity Search. *Under Submission*, 2019.

[19] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos. Fast subsequence matching in time-series databases. In *SIGMOD*, pages 419–429, New York, NY, USA, 1994. ACM.

[20] H. Ferhatosmanoglu, E. Tuncel, D. Agrawal, and A. El Abbadi. Vector Approximation Based Indexing for Non-uniform High Dimensional Data Sets. In *CIKM*, pages 202–209, 2000.

[21] T. Ge, K. He, Q. Ke, and J. Sun. Optimized product quantization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 36(4):744–755, Apr. 2014.

[22] A. Gogolou, T. Tsandilas, T. Palpanas, and A. Bezerianos. Progressive similarity search on time series data. In *BigVis, in conjunction with EDBT/ICDT*, 2019.

[23] A. Guttman. R-Trees: A Dynamic Index Structure for Spatial Searching. In *SIGMOD*, pages 47–57, 1984.

[24] P. Indyk and R. Motwani. Approximate nearest neighbors: Towards removing the curse of dimensionality. In *STOC*, pages 604–613, 1998.

[25] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(1):117–128, Jan 2011.

[26] W. Johnson and J. Lindenstrauss. Extensions of Lipschitz mappings into a Hilbert space. In *CONM*, volume 26 of *Contemporary Mathematics*, pages 189–206. 1984.

[27] S. Kashyap and P. Karras. Scalable knn search on vertically stored time series. In C. Apt, J. Ghosh, and P. Smyth, editors, *KDD*, pages 1334–1342. ACM, 2011.

[28] E. Keogh, K. Chakrabarti, M. Pazzani, and S. Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *KAIS*, 3(3):263–286, 2001.

[29] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: A survey and empirical demonstration. *Data Min. Knowl. Discov.*, 7(4):349–371, Oct. 2003.

[30] H. Kondylakis, N. Dayan, K. Zoumpatianos, and T. Palpanas. Coconut: A scalable bottom-up approach for building data series indexes. *PVLDB (11)6*, pages 677–690, 2018.

[31] W. Li, Y. Zhang, Y. Sun, W. Wang, W. Zhang, and X. Lin. Approximate Nearest Neighbor Search on High Dimensional Data - Experiments, Analyses, and Improvement (v1.0). *CoRR*, abs/1610.02455, 2016.

[32] J. Lin, E. J. Keogh, S. Lonardi, and B. Y. Chiu. A symbolic representation of time series, with implications for streaming algorithms. In *SIGMOD*, pages 2–11, 2003.

[33] M. Linardi and T. Palpanas. Scalable, variable-length similarity search in data series: The ulisse approach. *PVLDB*, 11(13):2236–2248, 2018.

[34] M. Linardi and T. Palpanas. ULISSE: ULtra compact Index for Variable-Length Similarity SEarch in Data Series. In *ICDE*, pages 1356–1359, 2018.

[35] M. Linardi, Y. Zhu, T. Palpanas, and E. Keogh. Matrix profile x: Valmod - scalable discovery of variable-length motifs in data series. In *SIGMOD*, pages 1053–1066, 2018.

[36] Y. A. Malkov and D. A. Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *CoRR*, abs/1603.09320, 2016.

[37] C. D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. Cambridge University Press, New York, NY, USA, 2008.

[38] T. Palpanas. Data series management: The road to big sequence analytics. *SIGMOD Record*, 44(2):47–52, 2015.

[39] T. Palpanas. Big sequence management: A glimpse of the past, the present, and the future. In *SOFSEM*, volume 9587 of *LNCS*, pages 63–80, 2016.

[40] B. Peng, T. Palpanas, and P. Fatourou. ParIS: The Next Destination for Fast Data Series Indexing and Query Answering. *IEEE BigData*, pages 791–800, 2018.

[41] D. Rafiei. On similarity-based queries for time series data. In *ICDE*, pages 410–417, 1999.

[42] T. Rakthanmanon, B. J. L. Campana, A. Mueen, G. E. A. P. A. Batista, M. B. Westover, Q. Zhu, J. Zakaria, and E. J. Keogh. Searching and mining trillions of time series subsequences under dynamic time warping. In *KDD*, pages 262–270, 2012.

[43] P. Schäfer and M. Högqvist. Sfa: A symbolic fourier approximation and index for similarity search in high dimensional datasets. In *EDBT*, pages 516–527, 2012.

[44] Y. Sun, W. Wang, J. Qin, Y. Zhang, and X. Lin. SRS: Solving c-approximate Nearest Neighbor Queries in High Dimensional Euclidean Space with a Tiny Index. *PVLDB*, 8(1):1–12, 2014.

[45] Y. Wang, P. Wang, J. Pei, W. Wang, and S. Huang. A data-adaptive and dynamic segmentation index for whole matching on time series. *PVLDB*, 6(10):793–804, 2013.

[46] R. Weber, H.-J. Schek, and S. Blott. A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces. In *VLDB*, pages 194–205, 1998.

[47] D.-E. Yagoubi, R. Akbarinia, F. Masseglia, and T. Palpanas. DPiSAX: Massively Distributed Partitioned iSAX. pages 1135–1140, 2017.

[48] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, Z. Zimmerman, D. F. Silva, A. Mueen, and E. Keogh. Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile. *DAMI*, pages 1–41, 2017.

[49] K. Zoumpatianos, S. Idreos, and T. Palpanas. ADS: the adaptive data series index. *VLDB J.*, 25(6):843–866, 2016.