Implementation of SFA in the MESSI framework

Bachelor's Thesis Expose

Jakob Brand
Humboldt-Universität zu Berlin

1 Introduction

In modern machine learning applications, the analysis of ordered sequences of real-valued data points (referred to as time series) and especially similarity search across large sets of time series are common tasks. As the amount of data gathered by applications, e.g. as scientific data in biology or financial data in economics [7] tends to increase further, index structures which are able to work on huge datasets are needed. To enable interactive exploration for similarity search, low response times in the order of milliseconds are necessary. In order to achieve this, Peng et al. introduced MESSI [1] as the first index for time series which is designed to work with in-memory operations on modern hardware, e.g. SIMD operations and multi-threading. MESSI uses the symbolic representation SAX [3] as a compressed representation of time series prior to indexing, but there are also other symbolic time series representations, as for example SFA [4]. Both representations guarantee the retrieval of queries with no false dismissals, but SFA provides a tighter approximation of the data than SAX. The goal of this work is to implement the SFA representation in the MESSI framework. While both symbolic representations in the MESSI framework should give the same results for similarity search queries, their response time and their index creation time should be compared.

2 Background & Related Work

2.1 Symbolic Representations

When performing similarity search on a high dimensional search space, the performance of the search using spatial index structures (e.g. R-Trees) becomes worse than a linear scan over the data already starting from around 20 dimensions, which is called the Curse of Dimensionality [4].
This is why, when working with large amounts of high-dimensional data, a dimensionality reduction can be applied to the time series. A similarity search query can then be solved approximately with an index in the reduced space. With the approximate solution, the actual nearest neighbours can be efficiently found in the original space. A symbolic representation of a time series approximates and discretizes the data points of the series into a sequence over a fixed alphabet which results in a word representation. Such a representation can further lower the memory consumption, so that the index over the reduced time series can fit into main memory, enhancing the response time for a similarity search query [4].
Two common approaches which will be further presented are SAX (Symbolic Aggregate approXimation) and SFA (Symbolic Fourier Approximation). SAX is based on approximation via PAA (Piecewise Approximate Aggregation) and a data-independent discretization. SFA is based on approximation with discrete Fourier transformation (DFT) and a data-dependent discretization called MCB (Multiple Coefficient Binning).

SAX/iSAX: As the SAX representation is based on PAA, a time series is split into w equally sized time intervals and the mean value for each interval is calculated. These

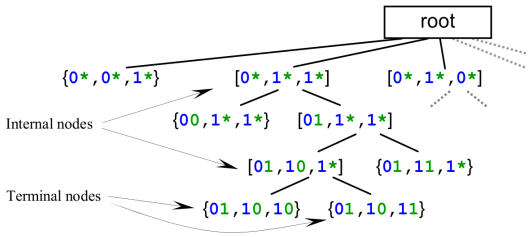

Figure 1: iSAX index [3] with a binary representation of alphabet sizes

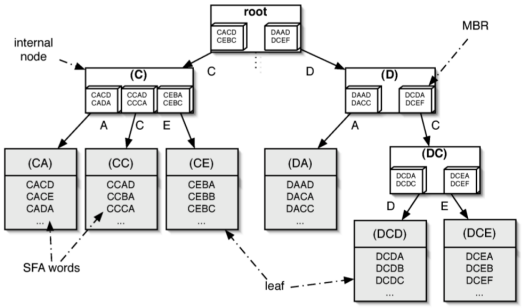

Figure 2: SFA trie [4]

mean values are then discretized by regions on the y-axis. The y-axis is divided by precomputed SAX breakpoints, which are calculated from equi-depth splitting a Gaussian Distribution over the number of desired symbols [2]. As an enhancement to SAX, the iSAX representation (indexable SAX) has been introduced to index large amounts of data while still enabling a fast exact and approximate search. Both SAX and iSAX provide a lower bounding distance measure (defined in Section 2.2) in the reduced space for the Euclidean Distance in the original space [3] to guarantee similarity search without false dismissals.
In the iSAX representation, different alphabet sizes are used for the SAX words. The words are constructed as binary numbers with alphabet sizes as power of two. Without any recomputation, words with a higher alphabet cardinality can be obtained from words with a lower alphabet cardinality for measuring lower bounding distances between them. With these different alphabet cardinalities, the iSAX index (Figure 1) can be built from nodes with low cardinalities. If the number of time series contained in a terminal node exceeds an upper bound *th*, the SAX space is split. An internal node marks this split and the cardinality of one interval in the terminal nodes is doubled. This results in an index structure where the SAX space is subdivided by internal nodes until each terminal node contains at most *th* time series [3].

SFA/indexable SFA: For the SFA representation, a DFT is applied on the time series which yields the Fourier coefficients. As part of preprocessing, the discretization intervals for each real and imaginary part of the coefficients are calculated using multiple Equi-Depth-Binnings (MCB). With these discretizations, each time series can be represented by keeping the first w Fourier coefficients discretized by the MCB intervals. In order to build an indexable data structure called SFA trie (Figure 2), the internal nodes contain prefixes of the SFA words and, when splitting nodes, because the threshold *th* of SFA words contained in the node has been reached, the prefix length gets increased [4]. The major difference between indexable SFA and iSAX is that SFA works with a fixed alphabet size and an increasing approximation length, whereas iSAX does the opposite.

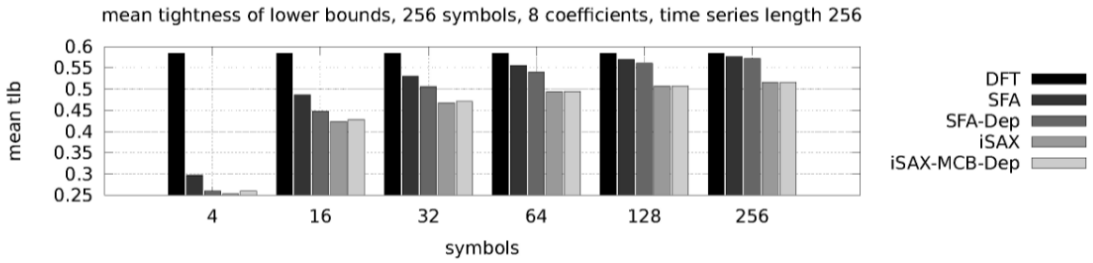

Figure 3: Tightness of lower bounds [4]

2.2 The Lower Bounding Property

An important property for symbolic representations to be used in the MESSI index is the so-called Lower Bounding Property. The distance of two time series S and T of length n is usually measured by the squared Euclidean Distance: $D_E^2(S,T) = \sum_{i=1}^{n}(s_i - t_i)^2$. For a symbolic representation *symb*, a distance measure *dist* is a lower bound to the Euclidean Distance, iff the distance between the symbolic representations of two time series is always smaller or equal to the Euclidean Distance of the two time series: $(dist(symb(S), symb(T)) \leq D_E(S,T))$[5]. Both iSAX and SFA provide MinDist functions which satisfy the lower bounding property. The result of an approximate search in the reduced space with such a distance measure is a superset of the result set in the original space with the Euclidean Distance. Therefore, there are no false dismissals when using the reduced space for a similarity search [5], so no results get lost by using the symbolic representation. It is possible however that the result set returned in the reduced dimensionality contains more elements (false-positive elements) which will be filtered out when answering a query.
Another important thing to mention when comparing SAX and SFA regarding their MinDist is that SFA has shown on datasets to have a tighter lower bound on average than SAX/iSAX [4] (Figure 3). The reasons for SFA's higher tightness of lower bounds are the use of the discretization scheme MCB and of DFT rather than iSAX's PAA for approximation and fixed intervals for discretization. In terms of query answering, this should result in less false-positive values as search results and more effective pruning of candidates when traversing the index, resulting in better response times [6].

2.3 The MESSI Approach

The in-MEmory data SerieS Index (MESSI) is an index for time series with improved performance at index construction and query answering compared to other state-of-the-art index approaches. This improvement is obtained by utilizing modern hardware techniques such as parallelization and SIMD instructions [1]. The index construction is based on the iSAX representation: a number of threads (index workers) compute the iSAX summaries (words), each for a part of the input data, and fill them into iSAX buffers. These buffers are filled in a way that each buffer contains the iSAX summaries

which are stored in one root subtree of the tree index. After all iSAX summaries are computed, these subtrees can be constructed. As every index worker thread gets an iSAX buffer assigned, the subtrees for each buffer are constructed independent from each other.

For answering a query, the index is traversed to the most promising leaf node which stores pointers to the real time series. The minimal true distance to a full (non-approximated) data series of the leaf node is saved as Best-So-Far (BSF). Then, the index workers traverse the subtrees using the lower bounding distance (MinDist) on the iSAX words, either pruning subtrees or placing their leaf nodes‘ time series in priority queues which are then searched for elements with a Euclidean Distance lower than BSF. When finding such an element, BSF gets updated and will contain the 1-NN-answer after all priority queues are processed.

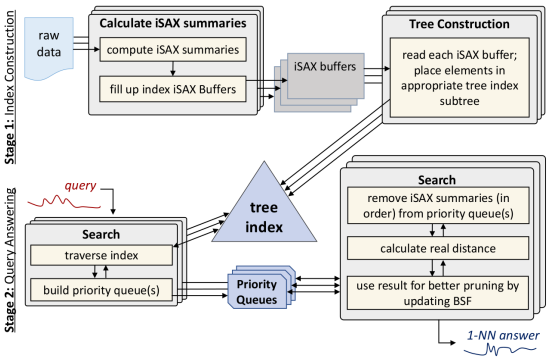

Figure 4: The MESSI workflow [1]

3 Objective & Motivation

The objective of this bachelor thesis is to implement the symbolic representation SFA into the MESSI index and compare its performance on datasets to the ”standard”-MESSI solution which utilizes the SAX representation. The breakpoints for the SAX discretization are fixed, whereas SFA needs a training of the MCB discretization values on the actual data prior to index construction. Therefore, the SFA approach is likely to be slower in the index building phase. On the other hand, the higher tightness of lower bound of SFA could return less false-positive results in the approximate search and be more effective for pruning subtrees [4]. This could potentially result in a lower query answering time for the SFA approach. These factors might have the potential to improve the overall runtime of MESSI, which shall be investigated in this bachelor thesis.
Furthermore, a factor which might affect the SFA-approach in a negative way is the amount of data given for training. If only a subset of the dataset is given, the training of the MCB values only on this subset is faster than the training on the whole set. On the other hand, these values might not be a good generalization for the entire dataset and could potentially slow down the runtime of the search.

4 Implementation & Methods

In order to implement SFA into the MESSI index, following steps have to be considered and implemented:

1. Training of the MCB breakpoints as part of preprocessing the time series (this step is not needed for SAX) on the entire dataset or a subset
2. Building SFA words with DFT of the time series and discretization with the MCB breakpoints instead of SAX words
3. Building the MESSI index on SFA words
4. Updating the MinDist-Calculation according to SFA

The code of MESSI is given in C by the authors including SIMD operations. The SFA operations are given as pseudocode and in JAVA. Additionally, the Lernaean Hydra Archive contains a C implementation of SFA [8]. The non-SIMD- and the SIMD-variants of the functions from the MESSI-framework with iSAX will be compared to the newly implemented non-SIMD-functions with SFA.
To compare the SAX- and the SFA-approach, mainly their query response time on different datasets and the parameter configurations (alphabet size and word length) for 1-NN queries will be relevant. The authors of the MESSI paper used several synthetic datasets with sizes between 50GB-200GB and two real datasets, Seismic and SALD [1]. The data series have a size of 256 points, except for SALD with 128 points. Therefore, the approaches should also be compared on these real and similar synthetic datasets. Furthermore, additional real seismological datasets will be used.
But also other parameters might be interesting to assess advantages and disadvantages of the approaches, e.g. the amount of false-positive values returned by the approximate search or the amount of nodes which are pruned during the search.

5 Outlook

There are a few points which would be interesting to further investigate concerning the use of SFA in the MESSI framework. On the one hand, SFA approximates using DFT, but it could also be modified to use any other approximation, e.g. the principal component analysis (PCA), which might have an influence on the response time. Another possible extension would be, instead of just replacing the SAX-words in the framework by SFA-words, to also modify the index construction to build SFA tries as used by the indexable SFA structure [4]. However, the latter modification would require lots of changes in the MESSI framework and is not trivial.

6 References

[1] Botao Peng, Panagiota Fatourou and Themis Palpanas. MESSI: In-Memory Data Series Indexing. April 2020. DOI: 10.1109/ICDE48307.2020.00036. http://helios.mi.parisdescartes.fr/~themisp/publications/icde20-messi.pdf

[2] Jessica Lin, Eamonn Keogh, Li Wei and Stefano Lonardi. Experiencing SAX: a novel symbolic representationof time series. April 2007. DOI: 10.1007/s10618-007-0064-z. https://link.springer.com/content/pdf/10.1007/s10618-007-0064-z.pdf

[3] Jin Shieh and Eamonn Keogh. iSAX: Indexing and Mining Terabyte Sized Time Series. August 2008. DOI: 10.1145/1401890.1401966. https://www.cs.ucr.edu/~eamonn/iSAX.pdf

[4] Patrick Schäfer and Mikael Högqvist. SFA: A Symbolic Fourier Approximation and Index for Similarity Search in High Dimensional Datasets. March 2012. DOI: 10.1145/2247596.2247656. https://www2.informatik.hu-berlin.de/~schaefpa/sfa_trie.pdf

[5] Christos Faloutsos, M. Ranganathan and Yannis Manolopoulos. Fast subsequence matching in time-series databases. May 1994. DOI: 10.1145/191843.191925. http://www.cs.cmu.edu/~christos/PUBLICATIONS.OLDER/sigmod94.pdf

[6] Kaushik Chakrabarti, Eamonn Keogh, Sharad Mehrotra and Michael Pazzani. Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases. May 2001. DOI: 10.1145/376284.375680. https://www.cs.rutgers.edu/~pazzani/Publications/locally.pdf

[7] Themis Palpanas. Data series management: The road to big sequence analytics. August 2015. DOI: 10.1145/2814710.2814719. http://helios.mi.parisdescartes.fr/~themisp/publications/sigrec15-bisemvision.pdf

[8] Karima Echihabi, Kostas Zoumpatianos, Themis Palpanas, Houda Benbrahim. The Lernaean Hydra of Data Series Similarity Search. https://github.com/karimaechihabi/lernaean-hydra