A General Approach For Supporting Time Series Matching using Multiple-Warped Distances

Article *in* IEEE Transactions on Knowledge and Data Engineering · May 2020

DOI: 10.1109/TKDE.2020.2998002

6 authors, including:

Rodica Neamtu

Worcester Polytechnic Institute

31 PUBLICATIONS **107** CITATIONS

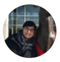
SEE PROFILE

Ramoza Ahsan

Worcester Polytechnic Institute

19 PUBLICATIONS **81** CITATIONS

SEE PROFILE



Cuong Nguyen

Worcester Polytechnic Institute

5 PUBLICATIONS **34** CITATIONS

SEE PROFILE

A General Approach For Supporting Time Series Matching using Multiple-Warped Distances

Rodica Neamtu *, Ramoza Ahsan *, Cuong Nguyen *,
Charles Lovering #, Elke A. Rundensteiner *, Gabor Sarkozy *
* Worcester Polytechnic Institute, Worcester MA | # Brown University, Providence RI
* rneamtu | * rashan | * ctnguyendinh | * rundenst | * gsarkozy@wpi.edu, # cjlovering@brown.edu,

***Abstract*—Time series are generated at an unprecedented rate in domains ranging from finance, medicine to education. Collections composed of heterogeneous, variable-length and misaligned times series are best explored using a plethora of dynamic time warping distances. However, the computational costs of using such elastic distances result in unacceptable response times. We thus design the first practical solution for the efficient GENeral EXploration of time series leveraging multiple warped distances. GENEX pre-processes time series data in metric point-wise distance spaces, while providing bounds for the accuracy of corresponding analytics derived in non-metric warped distance spaces. Our empirical evaluation on 66 benchmark datasets provides a comparative study of the accuracy and response times of diverse warped distances. We show that GENEX is a versatile yet highly efficient solution for processing expensive-to-compute warped distances over large datasets, with response times 3 to 5 orders of magnitude faster than state-of-art systems.**

***Index Terms*—Time Series Mining, Dynamic Time Warping, Similarity Exploration**

I. Introduction

A. Background and Motivation

Time series are prevalent in many scientific and commercial applications from weather, medicine, finance to energy forecasting [1], [2]. Finding similarities between time series by computing their distance is a core functionality of many data mining applications. It has been shown that computing the similarity among time series using a specific distance often misses insights that could have been revealed if another distance had been utilized [3]. Thus different applications rely on specific interpretations of similarity expressed through the use of diverse domain-specific distance metrics. For example, similarity in financial data analysis and market prediction [4], [5] is interpreted differently than in weather forecasting or medicine [1] reflected in the choice of distances used to express their analytics queries.

It has been repeatedly shown that warped distances are better suited than point-wise distances to explore sequences with different lengths and alignments [6], [7]. Thus, GDTW methodology [3] was designed to extend the capability of warping to a variety of point-wise distances in a unified manner. Using diverse warped distances for time series mining guarantees highly accurate results due to their ability to capture temporal misalignments and to compare sequences of different lengths. [3] showed experimentally that distances warped by this methodology improve the accuracy of certain data mining tasks such as classification, clustering and similarity retrieval by enabling flexible comparisons between unaligned sequences. This helps to reveal insights into datasets that would otherwise be missed.

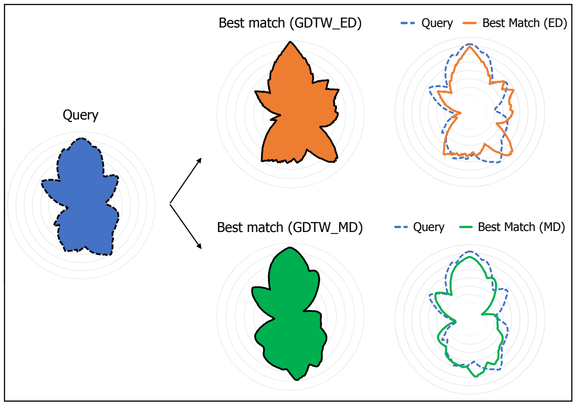


Fig. 1: Motivating example displays the best matches to a sample leaf retrieved using diverse warped distances (i.e. warped Euclidean and warped Manhattan.)

Fig. 1 displays a classification example applying two warped distances, namely warped Euclidean (commonly called DTW, here referred to as $GDTW_{ED}$) and warped Manhattan (here $GDTW_{MD}$) [3] respectively, to classify shapes of leaves in the OSULeaf dataset [8]. As the figure shows, DTW did not correctly classify the target leaf (blue), while the warped Manhattan found the matching species (green). Thus, analysts using a system based on only one distance, say the common DTW, would work with an incorrect classification. When identifying leaves that could induce severe allergic reactions in people, incorrect results could lead to dramatic consequences. This is only one example of how beneficial it is for analysts to have multiple warped distances at their finger tips for their data mining tasks. If they could compare with ease the results within the same system, they could decide which warped distance is best suited for their specific dataset.

Unfortunately, the benefits of using multiple warped distances are overshadowed by the quadratic complexity of their computation (prohibitive for large data sets) as illustrated below. Further, the lack of proven triangle inequalities for elastic distances hinders their usage in practice [3]. Some applications such as astronomy may be able to function with slower response times, while others such as medicine or the stock market require quick results for making good decisions in a timely manner.

Let us consider the limitations imposed by the lack of adequate performance for exploring large datasets using robust alignment tools. For example, the response time for finding one best match in a large dataset such as "Computers" from the UCR archive [8] using warped distances takes approximately 43 minutes per each sample sequence for each warped distance using the only existing system that incorporates many warped distances, namely GDTW [3]. We will show that the technology we propose instead finds the best match in about 0.2 seconds for each distance. Often we are interested in several, say 10 or 15, similar matches instead of just a single one. In this case, this would take almost 11 hours for each warped distance by GDTW for the Computers dataset, while our proposed GENEX technology can retrieve 15 matches in only 3 seconds as shown in Sec. V-B. GENEX thus offers a practical turn-around time 5 orders of magnitude faster.

In summary, there is a need for exploratory systems that support multiple warped distances within one integrated platform, guaranteeing quick response times and highly accurate results.

B. Limitations of State-of-the-Art

We summarize key challenges in solving the above problems of efficient exploration of time series datasets:

1.Lack of Performance for comparisons between sequences with different temporal alignments and/or lengths. The ubiquitous Euclidean Distance (ED) is used by many applications for fast distance computation [9], [10], [11]. However, ED and point-wise distances in general are brittle in comparing sequences with temporal misalignments or with different lengths. Unlike point-wise distances, time warped distances [3], including DTW [12], overcome this challenge, but their performance is impractical. That is, due to the high complexity of their computation and their non-metric nature reflected in the lack of proven triangle inequalities, exploring datasets using warped distances requires finding all pairwise similarity relationships. Thus it does not scale well to large datasets. Fortunately, as our results in Sec. V-B show, our proposed GENEX technology can be used to explore large datasets within seconds.

2. High data cardinality leading to a compromise between increased responsiveness and higher accuracy. Time series datasets such as the ones used to store energy consumption habits of millions of customers [13] tend to be huge. Thus performing all necessary pair-wise distance-based similarity comparisons is impractical. This leads state-of-the art techniques to focus on either increased responsiveness or increased accuracy. For DTW some systems provide exact or highly accurate solutions [4], [14] at the expense of increased response times. Others offer fast response times but with decreased accuracy [9], [10]. Yet clearly we need both. The system that incorporates other warped distances besides DTW into one single integrated platform, namely, GDTW [3], offers no optimizations beyond the LB_{Keogh} lower bound. It is thus impractical to use on large datasets due to its slow response times.

3. Supporting multi-distance driven similarity exploration. Most systems use one single distance [14], [15], [16]. Yet, as motivated above, exploratory results change based on the distance metric used. While GDTW [3] corresponds to a logical approach for warping a large variety of point-wise distances, its response times are impractical for large datasets. In this light, efficiently supporting a generalized similarity model that incorporates many distances is imperative.

C. Our GENEX Approach

In this work, we design a novel exploratory methodology called GENEX that empowers analysts to gain unique insights into time series datasets by performing similarity exploration instantiated by multiple time warped distances. Based on the general theoretical foundation underlying GENEX, analysts can with ease incorporate new distances and have them thereafter be efficiently supported by the system. Although other works [16], [17] "combine" ED and DTW, generalizing the pairing of point-wise and warped counterpart distances is far from trivial. The novelty of this work rests on establishing bounds that "extend" time series similarity from the metric space of point-wise distances to the non-metric space of warped distances in a general way, regardless of the distance used and without having to compromise between accuracy and response times. Mitigating this problem for a large array of distances at the theoretical rather than empirical level opens the door for increased versatility by allowing the incorporation of new distances, while guaranteeing accurate similarity exploration results with response times up to 5 orders of magnitude faster than existing systems. Our GENEX architecture described in Sec. IV-A supports the above functionality through three modules with specific functions: (1) enable analysts to choose specific point-wise distances and warp them for similarity exploration; (2) pre-process time series by creating similarity clusters and representatives based on their chosen distance; (3) perform efficient similarity searches by examining a much reduced number of sequences, namely finding the best candidate representatives and only explore the sequences that they represent.

Contributions:

1. GENEX offers a versatile framework by supporting the extension with a plethora of new distances, while guaranteeing the results in their usage for fast and accurate time series mining. (Sec. III-A)

2. As theoretical foundation of GENEX, we establish and prove a generalized triangle inequality between pairs of point-wise distances and their warped counterparts. This allows us to efficiently "extend" the similarity of sequences from the metric space of point-wise distances to the non-metric space of their counterpart warped distances. (Sec. III-B).

3. GENEX encodes similarity relationships between sequences into compact similarity clusters constructed using simple-to-compute point-wise distances and compresses them into representatives. Efficiency of processing is achieved by applying diverse time-warping distances to these representatives, instead of the raw data. This processing is supported by several GENEX optimization strategies including indexing for diverse distances to mitigate scalability issues. (Sec. IV-B).

4. Our experimental evaluation over 66 datasets in the UCR archive [8] depicts the changes in the similarity panorama revealed by the use of multiple warped distances. GENEX is up to 5 orders of magnitude faster than state-of-the-art systems. (Sec. V-B).

II. Key Concepts

A. Generalized Dynamic Time Warping

We summarize a large array of point-wise distances that now can be "warped" by a novel methodology [3] based on generalizing the classic DTW methodology [18], [15].

For two variable-length time series $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_m)$, with $n \geq m$, an $n \times m$ grid G is constructed. Similarly to the classic DTW algorithm, a *warping path* P is defined as a sequence of elements that forms a contiguous path from $(1, 1)$ to (n, m). The t^{th} element of P denoted as $p_t = (i_t, j_t)$ refers to the indices (i_t, j_t) of the element (x_{i_t}, y_{j_t}) in the path.

Hence, a path P is $P = (p_1, p_2, \ldots, p_t, \ldots, p_T)$, where $n \leq T \leq 2n - 1$, $p_1 = (1, 1)$ and $p_T = (n, m)$ and $n \geq m$. By "decoding" this general warping path and extracting the values for x_{i_k} and y_{j_k} at every position on the path, we conceptually construct the following two equal-length vectors: $X_P = (x_{i_1}, x_{i_2}, ..., x_{i_T})$ and $Y_P = (y_{j_1}, y_{j_2}, ..., y_{j_T})$, where some of the x_{i_k} and y_{j_k} are repeated while advancing on the path.

Considering an arbitrary point-wise distance d, the weight of the warping path P is then defined as the distance between X_P and Y_P computed using d. That is, $w(P) = d(X_P, Y_P)$. We note that the case of $d = ED$ defaults to the classic DTW.

Definition 1: The **Generalized Dynamic Time Warping Distance** corresponding to a distance d, denoted by $GDTW_d$, is the weight of the path P with the minimum weight, namely:

$$GDTW_d(X, Y) = \min_P(d(X_P, Y_P)).$$

There is an exponential number of warping paths satisfying these conditions [12]. Thus finding the minimum weight warping path is prohibitively expensive. Similar to the efficient computation of the classic DTW warping path using dynamic programming [19], the key idea in [3] is to construct the distance function recursively by incorporating the n^{th} coordinates based on the previous n-1 coordinates.

Definition 2: The distance d in Definition 1 must satisfy the following recursive condition: There exists a 3-variable function $f_d : \mathbf{R}^+ \times \mathbf{R} \times \mathbf{R} \rightarrow \mathbf{R}^+$ where $\mathbf{R}$ denotes the set of real numbers and $\mathbf{R}^+$ denotes the set of non-negative real numbers with respect to a distance d such that for vectors $X_P = (x_1, x_2, ..., x_n)$ and $Y_P = (y_1, y_2, ..., y_n)$ $(n \geq 2)$, we have:

$$\begin{aligned} d(X_P, Y_P) &= d((x_1, \ldots, x_n), (y_1, \ldots, y_n)) \\ &= f_d\left(d((x_1, \ldots, x_{n-1}), (y_1, \ldots, y_{n-1})), x_n, y_n\right). \end{aligned}$$

The f_d function tells us, given the distance measure on the first $n - 1$ coordinates $(x_1, ...x_{n-1}, y_1...y_{n-1})$ how to incorporate the n^{th} coordinates (x_n, y_n). This function is used to compute the $GDTW_d$ path recursively using dynamic programming.

Definition 3: The **general recursive expression** amendable for dynamic programming for warping a point-wise distance d is:

$$\gamma(i, j) = \min \begin{cases} f_d(\gamma(i-1, j-1), x_i, y_j), \\ f_d(\gamma(i-1, j), x_i, y_j), \\ f_d(\gamma(i, j-1), x_i, y_j). \end{cases} \tag{1}$$

with $\gamma(1, 1) = d(x_1, y_1)$.

Definition 4: Using Eqn. 1, the "warped" version of a distance d returns a **general dynamic warping distance** defined as:

$$GDTW_d(X, Y) = \gamma(n, m) \tag{2}$$

For the specific case of $d = ED$, this defaults to the known dynamic programming recursive expression for DTW[12]:

$$\gamma(i, j) = ED^2(x_i, y_j) + min(\gamma(i-1, j-1), \gamma(i-1, j), \gamma(i, j-1))$$

The complexity of the generalized warping (GDTW) process is the same as in the classical DTW algorithm [3], namely quadratic. Thus, the use of warped distances faces the same open challenges first revealed by the use of DTW, making it imperative to find viable, general efficient solutions, especially for exploring large datasets.

B. Key Concepts in Similarity

We introduce time series and sequences, then we define their similarity in the context of our generalized model instantiated by multiple warped distances. A time series $X = (x_1, x_2, ..., x_n)$ is an ordered set of n real values. A dataset $D = \{X_1, X_2, ..., X_N\}$ is a collection of N such time series.

There are many distances and similarity measures for exploring time series similarity [20]. Since the similarity measures can be expressed in terms of distances, for the remaining of this work we will not make the distinction between the two categories and will refer to them as *"distances" or "similarity distances"*.

Definition 5: A **sequence of a time series** X_p, denoted $(X_p)^i_j$, is a time series of length i starting at position j where $1 \leq i \leq n$ and $1 \leq j \leq n - i + 1$.

Definition 6: We define the **normalized distance** $\overline{d}$ between two sequences of the same length n, $X = (x_1, x_2, ..., x_n)$ and $Y = (y_1, y_2, ..., y_n)$ as:

$$\overline{d}(X, Y) = \frac{d(X, Y)}{g(n)},$$

where $d(X, Y)$ is a point-wise distance and $g(n)$ is specific for each distance and generally dependent on the length of the sequence.

Table I shows similarity distances and their normalized counterparts used in this work. For brevity, we denote Euclidean as ED, Manhattan as MD, Minkowski as Mink, and $GDTW_d$ as the warped variant of a general point-wise distance d. We chose these distances because their use for similarity exploration is documented [3] and well-known to the research community.

TABLE I: Popular similarity distances

	Definition	Normalized distance
ED	$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$	$\overline{ED}(X,Y) = \frac{ED(X,Y)}{\sqrt{n}}$
MD	$\sum_{i=1}^{n} \lvert x_i - y_i \rvert$	$\overline{MD}(X,Y) = \frac{MD(X,Y)}{n}$
Mink	$\max_{i=1}^{n} \lvert x_i - y_i \rvert$	$\overline{Mink}(X,Y) = Mink(X,Y)$
$GDTW_d$	$GDTW_d$	$\overline{GDTW}_d(X,Y) = \frac{GDTW_d(X,Y)}{2n}$

Definition 7: **Similar Sequences.** In the context of this work, we consider two sequences of the same length n, namely X and Y to be similar if the chosen normalized distance $\overline{d}$ between them is within a user specified similarity threshold ST, that is $\overline{d}(X,Y) \leq ST$.

III. GENEX Theoretical Foundation

A. Generalized Similarity Model

The key idea of our model is to first group together sequences of equal length that are similar according to Def. 7 into clusters. The clusters encode similarity relationships between sequences by imposing specific requirements that, as we prove later, insure that these clusters can be explored through their representatives instead of the raw data.

By construction, the representative R_k^i of a cluster C_k^i of sequences of equal length i, is a sequence from the cluster chosen such that the distance between this representative and any other sequence in the cluster is within half of the similarity threshold. In other words, $d(R_k^i, (X_p)_j^i) \leq ST/2$ for all $(X_p)_j^i$ in C_k^i.

Definition 8: Given the set T of all possible sequences $(X_p)_j^i$ of dataset D, a partition is created. That is, the sequences $(X_p)_j^i \in$ T are clustered into similarity clusters denoted by C_k^i based on a given distance d with their respective representatives R_k^i, such that all sequences $(X_p)_j^i \in$ T are in one and only one cluster C_k^i. These similarity clusters are said to be **GENEX similarity clusters**, denoted by C_k^i, if the following two properties hold:
(1) all sequences $(X_p)_j^i$ in a cluster C_k^i have the same length,
(2) each cluster C_k^i has one representative R_k^i such that $\overline{d}$ between any sequence $(X_p)_j^i$ in C_k^i and the representative R_k^i of this cluster C_k^i is smaller than half of the similarity threshold ST, that is
$\overline{d}((X_p)_j^i, R_k^i)) \leq ST/2, \forall i \in [1,n]$, $\forall j \in [1, n-i+1]$, and $\forall p \in [1,N]$.
In summary, the key requirements for placing sequences of equal length into the same similarity cluster are: (1) $\overline{d}$ between the sequences and the representative of the cluster must be the smallest compared to the $\overline{d}$ to any other representative at the time the sequence is examined for placement, and (2) $\overline{d}$ is also smaller than $ST/2$.

We refer to the similarity clusters and their representatives as **GENEX Bases**. These requirements entail that all sequences that belong to the same similarity cluster are similar to each other, meaning that the $\overline{d}$ between any two sequences in the cluster is smaller than ST.

Intra-Cluster Similarity Property: *For any two sequences of equal length i, namely X and Y belonging to the same cluster C_k^i, with C_k^i defined in Def. 8, $\overline{d}$(X,Y) defined in Def. 6 is within the threshold ST, that is, $\overline{d}(X,Y) \leq ST$, for all X, $Y \in C_k^i$.* This property is intrinsically based on proving a triangle inequality for the general distance d. Thus from this point forward we assume that our GENEX model only works with such distances. Proofs for specific distances such as MD and Mink are trivial, based on their own triangle inequalities. Since they are used in this paper, we give the proofs for MD and Mink along with our additional material [21], while the proof for ED can be found in [16]. All "metric" distances can work with our generalized similarity model.

Although the results of the grouping algorithm are specific to each point-wise distance, it is important to note that the difference in results has no further impact. That is, conceptually the exploration follows the same algorithm that, as we show in Sec. III-B, uses the counterpart warped version of each specific point-wise distance and leads to guaranteed results.

B. Expanding Similarity Exploration from Metric Point-Wise Distances Space to Non-Metric Warped Distances Space

Based on the above property that there exists a triangle inequality for distance d, our GENEX time-warped exploration framework is based on proving a customized triangle inequality between a general point-wise distance $\overline{d}$ and its warped counterpart $\overline{GDTW_d}$. This allows us to create compact GENEX clusters using the point-wise distance $\overline{d}$, yet explore these clusters through their representatives using the more powerful warped counterpart, namely $\overline{GDTW_d}$. We prove that the similarity between a sample sequence seq provided by the user and the representative of a GENEX similarity cluster as defined in Def. 8 "extends" to all sequences in that cluster. This empowers GENEX to *perform time warped comparisons of the sample sequence over the representatives instead of the entire dataset* D.

More specifically, for a general distance d, if $\overline{GDTW}_d$ between a sample sequence Q and the representative R_k^i is smaller than some value s, then we can guarantee that all sequences in that cluster C_k^i are similar to this sequence Q. More precisely, $\overline{GDTW}_d$ between Q and any of these sequences is smaller than $s + ST/2$. We prove that this important property holds for any general distance d that is "GENEX-compliant" as defined below. The value of s is chosen by the analyst, and the smaller this value is, the more similar the sequences are (the distance d between the sample and the sequence is smaller).

Definition 9: A general distance d is said to be "GENEX-compliant" if the following conditions are true for any sequences of equal length X, Y and Z:

1. d is **symmetric in the coordinates**, i.e., if we swap some coordinates in X and we make the same swaps in Y, then the value of $d(X,Y)$ does not change.
2. d satisfies the **triangle inequality**, i.e., $d(X,Z) \leq d(X,Y) + d(Y,Z)$.
3. d is **monotonic increasing** in the following sense: Let us pick a subsequence X' of X (we keep some of the coordinates from X) and let Y' be the respective

subsequence from Y (we keep the same coordinates). Let $\bar{X} = (X, X')$ (so we get $\bar{X}$ from X by repeating the coordinates in X') and let similarly $\bar{Y} = (Y, Y')$. Then we have the following:

$$d(X, Y) \leq d(\bar{X}, \bar{Y}) \leq d(X, Y) + d(X', Y').$$

These are natural assumptions. First, without the triangle inequality, a distance d would not even be a metric. The monotonicity condition is also satisfied by many distances such as the ones based on sum or max of base distances. Examples of GENEX-compatible distances include the $Lp - norms$, Inner Product, Intersection, Gower, Canberra, Wave Hedges, Pearson Coefficient and many other distances based on sums and respectively maximums as defined in [20]. While there are possibly other distances that can work with our framework, outside of the ones based on sums and maximums – we are only showcasing the ones for which a general proof exists.

When searching for the top-k most similar sequences to a given sample, sometimes we might have to explore more than one cluster, namely as many clusters as needed to contain at least k sequences combined, where k is the number provided by the analyst. When k is large, for some of these clusters the warped distance between their representatives and the sample is within ST/2, but for others, the warped distance between the representatives and the sample has some value s, close to ST/2. We can guarantee that all sequences in such clusters are similar to the sample, having a warped distance between the sample and any of these sequences within $s + ST/2$.

Lemma 1: Given $Y = (y_1, \dots, y_n)$ an arbitrary sequence of length n in any cluster as per Def. 8, with the representative of the cluster $R = (r_1, \dots, r_n)$ and a sample sequence $Q = (q_1, \dots, q_m)$, then the following is true:
If $d(R, Y) \leq ST/2$ and $\overline{GDTW}_d(Q, R) \leq s$, then we have $\overline{GDTW}_d(Q, Y) \leq s + ST/2$.

This allows us, for small values of s, to guarantee the results of exploring our similarity clusters using $GDTW_d$.

Proof: (Case: sequences of the same length). From the assumptions of Lemma 1 we have:

$$d(R, Y) \leq ST/2 \tag{3}$$

Furthermore, from the definition of $GDTW_d$, $\overline{GDTW_d}$, and the assumptions of Lemma 1 we know that there is a warping path P between Q and R from $(1,1)$ to (n,n) with the $GDTW_d$ weight at most $2ns$. More precisely, P is a contiguous path in the $n \times n$ grid from $(1,1)$ to (n,n). The t^{th} element of P is $p_t = (i_t, j_t)$. Thus $P = (p_1, p_2, \dots, p_t, \dots, p_T)$, where $n \leq T \leq 2n-1$, $p_1 = (1,1)$ and $p_T = (n,n)$. By "decoding" this path and extracting the values x_{i_k} and r_{j_k} at every position on the path, we construct the two equal-length vectors: $Q_P = (q_{i_1}, q_{i_2}, ..., q_{i_T})$ and $R_P = (r_{j_1}, r_{j_2}, ..., r_{j_T})$, where some of the q_i and r_j are repeated while advancing on the path. Then for this path P we have

$$GDTW_d(Q, R) = d(Q_P, R_P) \leq 2ns. \tag{4}$$

We now have to show that there is a warping path from $(1,1)$ to (n,n) between Q and Y with GDTW weight at most $2nST$. In fact we will show that the same warping path P will be good, i.e., we need to prove that:

$$GDTW_d(Q, Y) \leq d(Q_P, Y_P) \leq 2n(s + ST/2) \leq 2ns + nST. \tag{5}$$

From the triangle inequality, we know that:

$$d(Q_P, Y_P) \leq d(Q_P, R_P) + d(R_P, Y_P).$$

From (4) we know for the first term that

$$d(Q_P, R_P) \leq 2ns.$$

Thus in order to prove (5), all we need is to prove for the second term that below holds:

$$d(R_P, Y_P) \leq nST. \tag{6}$$

We get R_P (resp. Y_P) by repeating some coordinates in R (resp. Y), where each coordinate is repeated at most $(n-1)$ times. Using the monotonicity condition we get an upper bound if we repeat every coordinate in R (respectively Y) *exactly* n times. Thus we get the following upper bound using (3) and the fact that the distance is symmetric and monotonic increasing:

$$d(R_P, Y_P) \leq d\left((R, \dots, R), (Y, \dots, Y)\right) \tag{7}$$

$$\leq nd(R, Y) \leq n\frac{ST}{2} \leq nST \tag{8}$$

This proves (6).

Proof sketch (Case: sequences of different lengths.) Let R and Y' be sequences of length n where R is the representative of the cluster, Y' an arbitrary sequence in the cluster and X a query sequence of length m, with $m \leq n$. Without loss of generality we consider here the case of $m \leq n$ but the proof is very similar for $n \leq m$. In $\overline{GDTW_d}$ defined in Table I we divide by $2n$ because the warping path may have length up to $m + n \leq 2n$. The matrix $M(X, Y')$ is an $m \times n$ matrix and the warping path connects $(1,1)$ to (m,n). Other than this, the proofs for sequences of different lengths and for sequences of the same length are the same.

We note that for the special case when $s = ST/2$ the Lemma 1 guarantees that exploring clusters that are within ST/2 of the sample sequence will lead to sequences that are similar to this sample within ST.

Lemma 2: Let d be a general distance satisfying Def. 9. Given $Y = (y_1, \dots, y_n)$ an arbitrary sequence of length n in any cluster as per Def. 8, with the representative of the cluster $R = (r_1, \dots, r_n)$ and a sample sequence $Q = (q_1, \dots, q_m)$, then the following is true: If $d(R, Y) \leq ST/2$ and $\overline{GDTW}_d(Q, R) \leq ST/2$, then we have $\overline{GDTW}_d(Q, Y) \leq ST$.

The proof for this specific case is very similar to the proof for Lemma 1, just making the following changes: 1) replacing in (4) s with ST/2 which leads to the right term to be n ST; and 2) changing the right term of (5) to be $2nST/2 = nST$. The triangle inequality for d remains the same, so the only difference between Lemma 2 and its generalized form Lemma 1 is that now that the second term in (5) is 2 n ST/2 = n ST. Other than this, the rest of the proof is the same.

In addition, analysts can prove these lemmas for other specific distances on an individual basis. We give such examples of proofs for the distances used in this paper, MD and Mink in our additional material [21], while the proof for Lemma 1 for ED can be found at [17].

IV. GENEX Framework

A. GENEX Overview

Our GENEX provides fast and accurate insights into time series datasets by using the theoretical foundation in Sec. III. As depicted in Fig. 2, GENEX facilitates time series exploration with multiple distances using the following modules:

OperationManager: enables analysts (Fig. 2-a) to perform similarity exploratory operations listed in Sec. IV-C based on the efficient processing strategies described in Sec. IV-D.

DistanceManager: provides a repository of warped distances. Analysts can add new point-wise distances similar to [3]. Both point-wise distances and their warped counterparts are accessible to the rest of GENEX (Fig. 2-b).

BaseManager: pre-processes time series datasets using the point-wise distance chosen by the analyst, and constructs GENEX similarity clusters (Fig. 2-c).

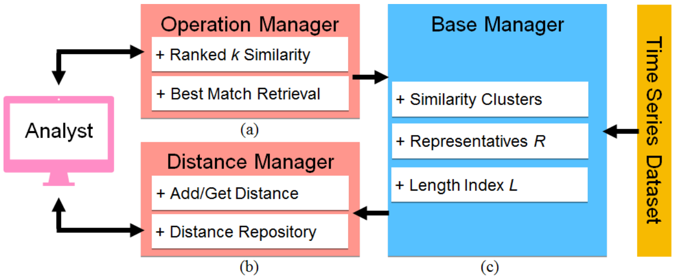


Fig. 2: GENEX Overview

B. Base Manager Construction

The algorithm for constructing similarity clusters is independent of the distance chosen by the analyst. As indicated in Sec. III, our aim is to construct clusters with a diameter smaller or equal to ST, such that any sequence in a cluster is similar to the representative of the cluster within ST/2. As shown in Sec. III-A, this guarantees that all sequences in the cluster are similar to each other.

There are many strategies to build such similarity clusters. Similar to [16], GENEX uses an algorithm that is empirically robust where clusters are incrementally constructed by adding the given sequences to the existing cluster whose representative has the minimum distance to the sequence and that is also within ST/2 of the sequence. If no such similarity cluster exists, a new cluster is constructed with the current sequence becoming the representative of this new cluster. This process is performed for all sequences in the dataset and it is parallelized across different lengths with concurrent threads.

The complexity of constructing the GENEX Base for each distance d is in the worst case $\mathcal{O}(nl^3g)$ where l is the number of distinct lengths that each time series is decomposed into, g the number of groups and n the number of time series in the dataset. The l^3 term is due to the $\mathcal{O}(l^2)$ sequences and the $\mathcal{O}(l)$ the cost of computing d, assuming a linear complexity of computing d for any two sequences of length l. It has been shown probabilistically that the expected number of groups is $\sqrt{nl}$ [16]. However in the worst case each item could become its own group, i.e., $g = \mathcal{O}(nl)$. For the general case where $l \ll n$, we treat l as constant with respect to n, so the expected complexity is $\mathcal{O}(n^{\frac{3}{2}})$.

C. Similarity Exploratory Operations

The Operation Manager allows the analyst to choose a specific distance for similarity exploration and a sample sequence seq.

Similarity search allows analysts to perform two subclasses of operations expressed in the following syntax:

```
Q OUTPUT set of $X_p$
   FROM  D WHERE Sim <= min| ST, seq = q
   MATCH = Exact(L)|Any
   d in {ED,MD, Mink, or other
   distances in the Repository}
   k=provided by user
```

Ranked top K similarity search returns the top k most similar sequences to a user-supplied sample seq. The distance is chosen by the analyst and returned sequences have minimum or within ST distance with the provided sample seq. If MATCH=Exact, the returned sequences have the same length L as sample seq, otherwise all length sequences are explored. *Use Case*: A financial analyst may want to retrieve the top 10 stocks whose fluctuations are similar to that of the Apple Stock over a specific time period. This illustrates the case when the sample sequence is a sequence present in the dataset. Alternatively, an analyst can "design" a desired stock fluctuation and search the datasets for the top 10 stocks similar to this desired sequence. Such sequence is likely not to exist in the dataset, in which case the closest matches are retrieved.

Best match retrieval. As a special case of the similarity search class for k= 1, this subclass returns the best match to the sample sequence.
Use Case: An analyst might want to retrieve the stock having the closest selling price with that of Google stock over one year. Or a doctor might want to find the most similar shape to the ECG of a patient from an annotated collection of ECGs to help diagnose specific heart conditions.

D. Exploratory Processing Strategies

Based on its formal foundation (Sec. III-B), our GENEX Operations Manager applies time-warped strategies on the compact GENEX bases. In this section we describe the processing strategies that handle the similarity search operations described in Sec. IV-C.

To optimize the similarity exploration we construct a LengthIndex L which indexes the set of representatives of each length. As shown below, we explore these representatives first; then only the corresponding sequences in the similarity clusters that we are interested in are explored, instead of the entire raw data. To find the most similar k sequences to a sample seq, the OperationManager selects a set of candidate representatives. Then it computes the distance to the sample from all the represented time series, selecting the k most similar sequences. The selection of candidate representatives is optimized as shown in Fig. 3-a and explained below. For finding the best match sequence we only select one candidate representative and then explore the sequences that belong to its cluster.

Similarity Search Operations involve both retrieval of the k most similar sequences and the best match to a given sample. We discuss below the strategy for retrieving the top k most similar sequences to a given sample, while the retrieval of the best match becomes the specific case of k=1. We first find the set of representative candidates whose similarity clusters are most likely to contain the k most similar sequences. Then we explore the sequences in these clusters and rank them based on their similarity to the given sample, thus selecting the top k most similar sequences. These strategies are the same regardless of the chosen distance.

Ranked top K similarity search: We denote the desired number of similar sequences chosen by the analyst as k. We denote the minimum number of subsequences that the candidate representatives must represent to insure 100% accuracy as k_e. GENEX retrieves k most similar sequences similar to [17] by first finding the representatives having the least distance with the query sequence and that have at least $k_e \geq k$ members combined. In the next step, the pairwise warped distances of at least k_e sequences to the sample query are computed and the top k sequences with the smallest distances are returned.

In order to find the candidate set of k_e sequences, we first retrieve the representatives of each specific length by using the LengthIndex L (Fig. 3-a). Then we compute the $\overline{GDTW_d}$ between each representative and the sample sequence (Fig. 3-b), selecting those whose distance is the smallest and within ST/2. A max-heap maintains the most similar representatives H_r (Fig. 3-c). Before H_r contains at least k_e sequences, any representative with a warped distance to the sample smaller than $\frac{ST}{2}$ is added to the heap. This is *heapified* based on the distance from the representatives to the sample. From here on, H_r maintains the current worst candidate R^*, enabling early abandonment techniques. R^* is evicted when a new better candidate is added to H_r. This results in a max-heap populated with the set of candidate representatives most similar to the sample (Fig. 3-d). The similarity clusters of these representatives now contain the most similar k sequences to the sample. To retrieve them we apply the same methodology, but instead of the representatives we are now exploring the k_e sequences. The max-heap H_s has a capacity of k instead of k_e, resulting in the k most similar time series (Fig. 3-e).

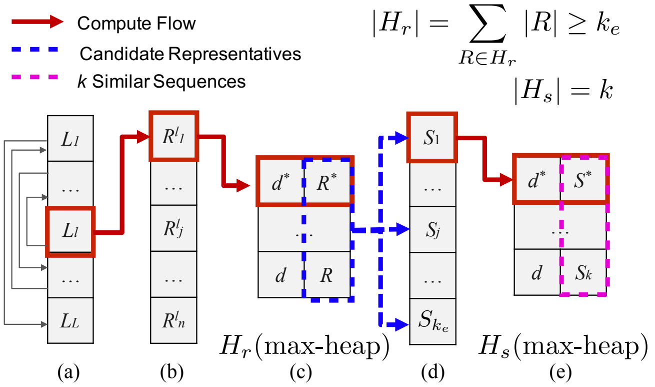


Fig. 3: Operation Manager Internals

The complexity of the top k similarity operation is composed of the complexity of selecting the candidate representatives and that of retrieving k top ranked sequences. The complexity of selecting representatives is $\mathcal{O}(|G|\log(|G'|)l^2)$, where $|G|$ is the number of examined clusters, $|G'|$ is the number of clusters similar to the sample which in the worst case, when each cluster has only one member, is k_e. The complexity of the warped distance computation for sequences of length l is l^2. The complexity of retrieving k sequences in G' is $\mathcal{O}(k_e \log(k)l^2)$. In practice, the retrieval requires processing at least one cluster. So k_e is at least the size of the best candidate cluster. The overall complexity is $\mathcal{O}(|G|\log(k_e)l^2 + k_e\log(k)l^2)$. However, k and k_e are constants and additionally $\log(k) \ll k_e$ and $|G'| \leq k_e \ll |G|$, so we summarize the overall complexity for the top k similarity as $\mathcal{O}(|G|l^2)$. It is important to note that our k is generally a very small number, thus the difference in complexity and performance of using other methods related to the k-selection problem is not significantly impacting the real response time. For $k = 1$ the complexity is $\mathcal{O}(G + m)$, where G is the number of clusters explored and m is the number of sequences in the best match cluster.

E. Optimizations for Exploratory Operations

We devise general strategies to work with any distance d and efficiently retrieve the k most similar sequences to a given sample seq, by optimizing the retrieval of the best candidate clusters and of the top k similar sequences within these clusters as described in Sec. IV-D. Additionally, existing distance-specific optimizations can be incorporated into our framework.

General distance optimization: For a given sample sequence of length $\mathcal{L}$, we start the search for candidate clusters with the ones of the same length as the query, as items with similar lengths are more likely to be similar [22]. This allows us to better leverage early abandonment techniques.

Distance specific optimization: For ED we use the LB_{Keogh} [15] lower bound to build envelopes around the representatives. These envelopes are computed during the preprocessing step, allowing us to "prune" many unpromising representative candidates. Similar techniques can be developed for MD, Mink and other monotonic increasing distances to optimize the construction of similarity clusters.

V. Experimental Evaluation

A. Experimental Setup

Our GENEX framework can incorporate a large array of distances. *Thus, instead of highlighting the merits of individual distances we focus on showcasing the accuracy and efficiency of our method compared to state-of-the-art systems.* For this, we implement a select subset of warped distances known to the research community, namely, $GDTW_{ED}$ (DTW),

$GDTW_{Mink}$ (warped Minkowski), and $GDTW_{MD}$ (warped Manhattan).

GENEX is implemented in C++11 and experiments are conducted on a Linux machine with a 3.30 GHz Intel Xeon CPU and 64GB of memory. All our experiments are reproducible and the detailed results are available at [21] while our code is publicly available at [23].

Alternate state-of-the-art methods. Although there are many previous approaches that have indexed DTW, we focus on all three warped distances equally, thus we use for comparison systems that implement all three of them: Generalized Dynamic Time Warping (GDTW) [3] and Piecewise Aggregation Approximation (PAA) [14]. To avoid confusion between the GDTW as the system and the generalized warped distances denoted with $GDTW_d$ and its variants, we refer to the original GDTW system as introduced in [3] as GDTWSys for this evaluation section. The GDTWSys implementation finds the exact solution by computing all pairwise distances from the sample to each subsequence in a dataset. Thus we use its results as ground truth for assessing the accuracy of other methods. PAA is a well-known data-reduction method that finds an approximate solution by averaging consecutive pieces of equal length in each sequence. Using simple heuristics we decided to average every 3 data points of a time series to obtain a data point of the PAA time series. PAA's ability to use multiple warped distances makes it appropriate for comparison to GENEX. Our preprocessing is a one-time step whose results can be re-used repeatedly thereafter during analysis. The expense of this step pays off leading to increased online time responsiveness. Similar with [24], which uses a preprocessing step, we thus don't include preprocessing phase costs as part of on-line retrieval costs. We instead report the online performance which reflects the analyst‘s experience. Both competitors take advantage of the well-known lower bound LB_{keogh} optimization in their implementations.

Next, we explain why other methods, in particular, iSax [25] while also mining time series, are not suitable for experimental comparison in our context. iSax focusses on a data structure to scale to large time series datasets that cannot fit in main memory. For this, they support bulk loading strategies of data from a disk structure. GENEX instead focuses on fast and accurate similarity search using in-memory structures. In addition, as memory becomes increasingly affordable at larger capacity, the proposed structures of GENEX supporting interactive exploration experience for increasingly larger datasets can fit into modern main memory. In a nutshell, [25] targets disk-access bound indexing for a cheap to compute metric, while our objective is to speed up a rich variety of CPU bound time series similarity queries in memory with an expensive to compute measure.

Datasets. We run experiments on 66 datasets from the benchmark UCR time-series collection [8]. These datasets were selected in increasing order of their size computed as $size = n * (m * (m-1)/2)$, where n is the number of time series and m is the length of each time series. We do not run experiments on the remaining datasets in the archive due to the extremely long time necessary for the competitor systems to run. We normalize each sequence $X = (x_1 ... x_n)$ based on the maximum (max) and minimum (min) values in each dataset [15] by computing the normalized values for each point x_i as $\frac{x_i - min}{max - min}$.

Experimental methodology.

We perform three classes of experiments:

1. Experiment on similarity search.

1.1 **Best match retrieval.** We first evaluate the accuracy and speed of our system in retrieving the best match sequence to a given sample using each of the three warped distances. We compare our accuracy and response time with the two benchmark methods: Generalized Dynamic Time Warping (GDTWSys) and Piecewise Aggregation Approximation (PAA) to show that GENEX has comparable accuracy, while being orders of magnitude faster than both competitors.

1.3 **Trade-off evaluation.** We find the best similarity threshold ST for each specific dataset, the one that leads to the lowest error rate and fastest response time. These results can assist analysts in establishing the best similarity setting for exploring specific datasets.

2. Evaluating GENEX bases. We create GENEX bases for $GDTW_{ED}$, $GDTW_{MD}$, $GDTW_{Mink}$ for 66 datasets in the UCR collection. For each distance, we evaluate the GENEX bases by measuring the **compression rate** and the **construction time** of our preprocessed clusters when varying similarity threshold ranges across datasets. This results in a "similarity panorama" that helps analysts better understand their specific datasets. We present these results with the sole purpose of showing that our data compression strategy allows us to process fairly large datasets into memory.

3. Case Study: Using GENEX for botanical applications. To demonstrate the advantages of a multi-distance system, we conduct a classification experiment on the OSULeaf dataset [8]. We show that other distances can be better than the classic DTW for specific data mining tasks, which reinforces the need to have multiple warped distances integrated within the same system.

B. Experimental Results

1) Experiment on Similarity Search: Each dataset in the UCR archive has a Test set and respectively a Training set. To streamline this experiment we use the Test set to search for similar sequences. Thus, we name this set DATA. We want to experiment with sequences both inside and outside the dataset, so we organize our search as follows: when we want to experiment with samples outside the dataset we use the Training set to select our sample sequences, so we name this set the QUERY set. When we want to experiment with sequences inside the dataset, we select them from the Data set. For each specific distance, we run the similarity search experiment using the following methodology:

First, we generate 30 different samples of arbitrary length for each dataset by randomly selecting fifteen subsequences from the DATA set and fifteen subsequences from the QUERY set. This selection scheme covers samples both present in the dataset and not present in the dataset. Next, we find the best match and respectively the top-15 most similar sequences of each sample in each dataset using GENEX and the two alternative methods. Finally, we compute the average error rate over the 30 samples in each dataset for GENEX and PAA using the results of GDTWSys as ground truth. The time responses for each method are also measured by averaging the running times of these 30 samples for each dataset.

1.1. Accuracy and response time for finding best match sequence. We assess the accuracy of a solution by measuring its relative error to the ground truth calculated as follows: we denote d_{GENEX}, d_{PAA} and $d_{GDTWSys}$ as the respective distances between the given sample and the solution computed using each one of the three warped distances by GENEX, PAA and GDTWSys respectively. The relative errors of GENEX and PAA are calculated using the formula $|d_{GENEX} - d_{GDTWSys}|$ and $|d_{PAA} - d_{GDTWSys}|$. Since GDTWSys gives the ground truth, we only assess the relative errors of GENEX and PAA. Table II shows that the relative error of GENEX is up to 4 times lower than that of PAA.

TABLE II: Average errors of PAA and GENEX across 66 datasets

	PAA	GENEX
$GDTW_{ED}$	0.7×10^{-3}	0.2×10^{-3}
$GDTW_{MD}$	1.3×10^{-3}	0.8×10^{-3}
$GDTW_{Mink}$	7.7×10^{-3}	3.6×10^{-3}

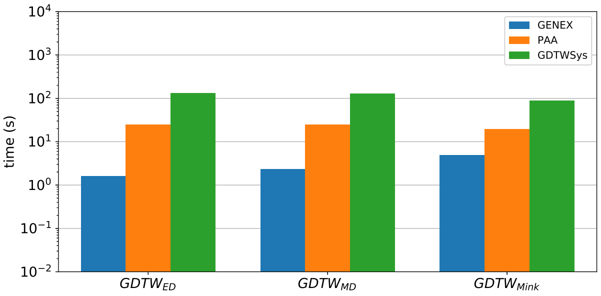


Fig. 4: Average response time of GDTWSys, PAA and GENEX by distance across 60 medium and small datasets.

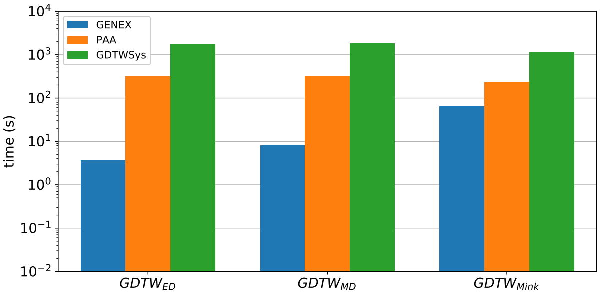


Fig. 5: Average response time of GDTWSys, PAA and GENEX by distance across 6 large datasets.

Fig. 4 displays the average response times of each method across 60 medium and small datasets. GENEX is approx. 3862 times faster than GDTWSys for $GDTW_{ED}$, 731 times for $GDTW_{MD}$ and 240 times for $GDTW_{Mink}$. GENEX is 980 times faster than PAA for $GDTW_{ED}$, 182 times for $GDTW_{MD}$ and 66 times for $GDTW_{Mink}$. Fig. 5, displays the average response times of the three methods across the 6 largest datasets. Here, GENEX is faster than GDTWSys 13106 times for $GDTW_{ED}$, 807 times for $GDTW_{MD}$ and 55 times for $GDTW_{Mink}$. GENEX is 3328 times faster than PAA for $GDTW_{ED}$, 180 times for $GDTW_{MD}$ and 15 times for $GDTW_{Mink}$.

This shows that the larger the datasets, the faster GENEX becomes, up to 4-5 orders of magnitude faster than the competitors.

We plot the individual relative errors and response times of all 66 datasets for the three distances in Fig. 6 and Fig. 7, respectively. In each subplot, from left to right, the datasets are sorted in ascending order by the number of subsequences they contain. The lines denoting GENEX for all three distances in Fig. 6 mostly lie below the PAA line, indicating high consistency of GENEX in achieving very low error rates. Furthermore, as the datasets increase in size in Fig. 7, the difference in the response times between GENEX and the other two methods increases dramatically, showing that GENEX is 4 to 5 orders of magnitude faster.

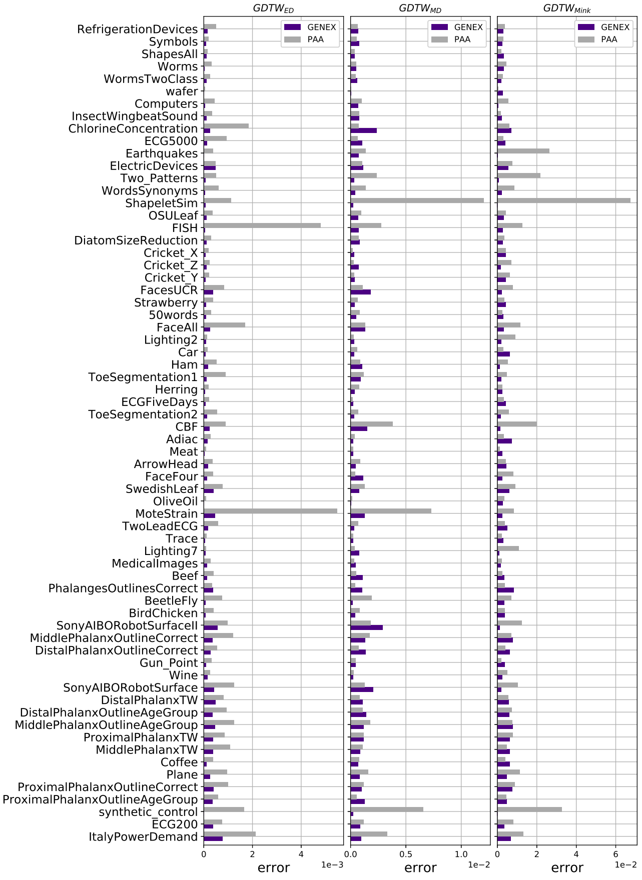


Fig. 6: Error of PAA and GENEX for 66 datasets using $GDWT_{ED}$, $GDWT_{MD}$, and $GDWT_{Mink}$

In summary, GENEX is up to 5 orders of magnitude faster than GDTWSys and 4 orders of magnitude faster than PAA.

1.2. Accuracy for top-15 most similar sequences. Here we showcase the ability of GENEX to find very fast ranked similar matches to a given sample with very high accuracy. We reuse the 30 samples from the experiment for finding the best match, but now we find the top 15 most similar sequences for each sample. We also show that GENEX can achieve perfect accuracy by varying the number of sequences explored. We use the same notation as in the previous experiment and compute the relative error based on the average relative errors of the top-15 matches, using the formula:

$$\frac{\sum_{i=1}^{k} |d_{GENEX_i} - d_{GDTWSys_i}|}{k}$$

Fig. 8 shows the GENEX similarity search error averaged across 66 datasets for k=15 using $GDTW_{ED}$, $GDTW_{MD}$, and $GDTW_{Mink}$ respectively. We note that as the percentage of explored sequences increases, the error rate rapidly declines and reaches 0 at very low percentage values. Specifically, the average and respectively maximum percentages to reach perfect accuracy are respectively 1.5% and 9.3% for $GDTW_{ED}$, 2.2% and 9.6% for $GDTW_{MD}$, and 0.6% and 4.3% for $GDTW_{Mink}$.

In summary, GENEX can achieve 100% accuracy by exploring on average less than 1.5% of all sequences in any of the 66 datasets.

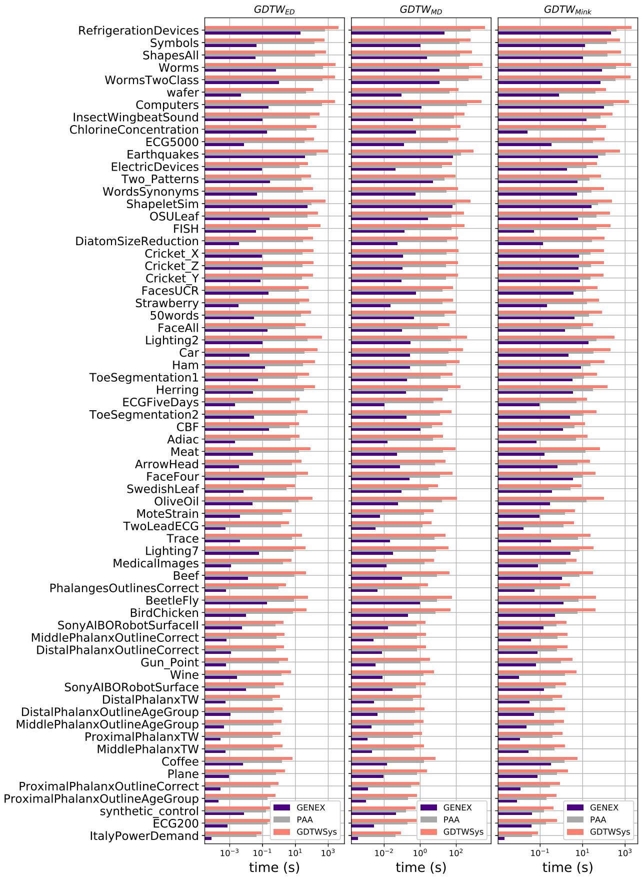


Fig. 7: Time response of PAA, GDTWSys and GENEX (in logarithmic scale) for 66 datasets using $GDWT_{ED}$, $GDWT_{MD}$, and $GDWT_{Mink}$

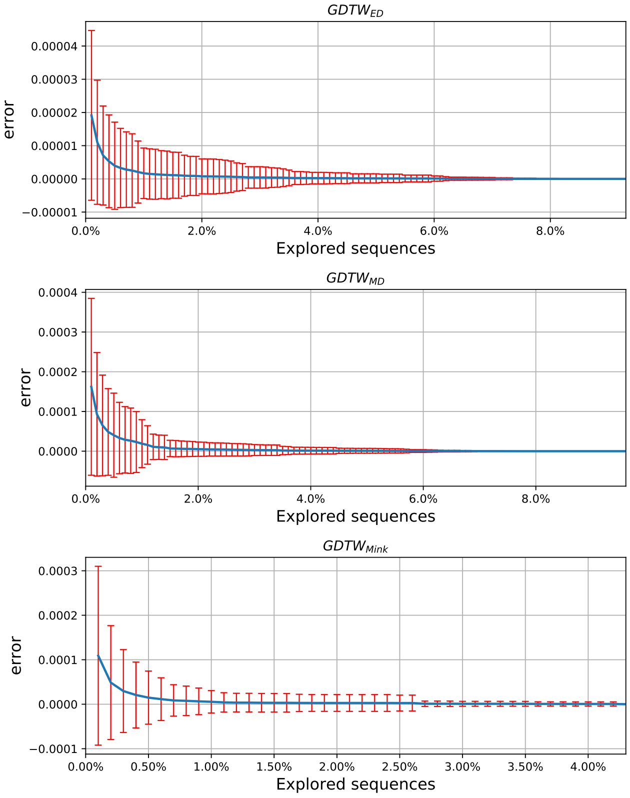


Fig. 8: GENEX similarity search error for $GDTW_{ED}$, $GDTW_{MD}$, and $GDTW_{Mink}$ of 66 datasets versus the percentage of explored sequences.

1.3. Trade-off between accuracy and response time As shown in [16], there is a trade-off between accuracy and response time when varying the similarity threshold, allowing analysts to use the most suitable ST to achieve the highest accuracy and fastest response time. We scale up this trade-off experiment to 66 datasets and across the three distances. The results for each distance are illustrated in Fig. 9. All three subplots in Fig. 9 reveal similar trade-off patterns for $GDTW_{ED}$, $GDTW_{MD}$ and $GDTW_{Mink}$. As we increase ST, the error rate increases, and the time response decreases. The “balanced” spot where we achieve the fastest response time and the lowest error rate is around 0.25 for $GDTW_{ED}$, 0.16 for $GDTW_{MD}$ and 0.24 for $GDTW_{Mink}$.

2) Evaluating GENEX Bases: Our method achieves a great advantage in speed and accuracy largely due to the compact representation in the form of similarity clusters performed during the preprocessing step. In this experiment, 66 datasets in the UCR archive have been pre-processed using ED, MD and Mink. Here we investigate how the use of these distances and varying similarity thresholds affect the construction time and the cluster compactness. Similar to [17], we define compression rate as:

$$100\% - \frac{\text{\# of cluster} + \text{avg. cluster size}}{\text{total \# of sequences}}\%.$$

This definition measures the ratio of the average number of sequences unexplored by GENEX to the original number of sequences. Fig. 10 shows that in general, over all datasets, the pre-processing times are faster for MD then ED. The processing times for Mink are slower than the other two distances. We note a correlating trend in the compression

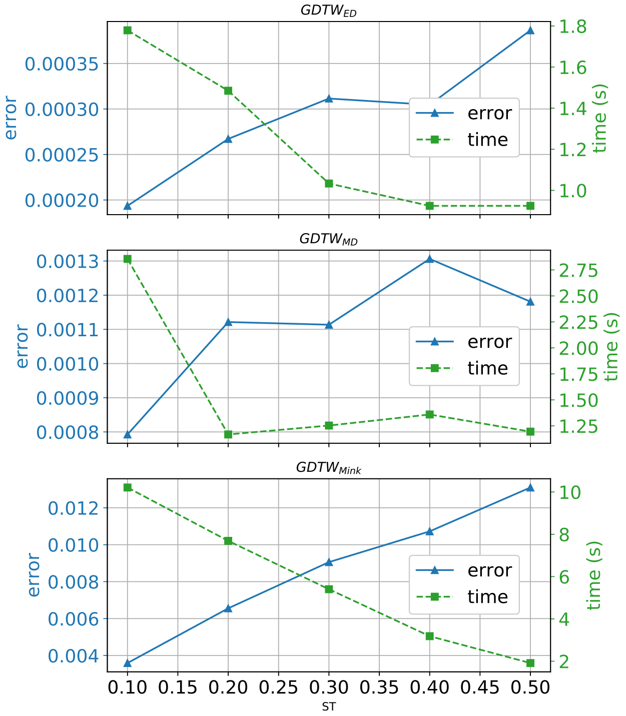


Fig. 9: Response time and error trade-off of $GDTW_{ED}$, $GDTW_{MD}$, and $GDTW_{Mink}$ varying ST.

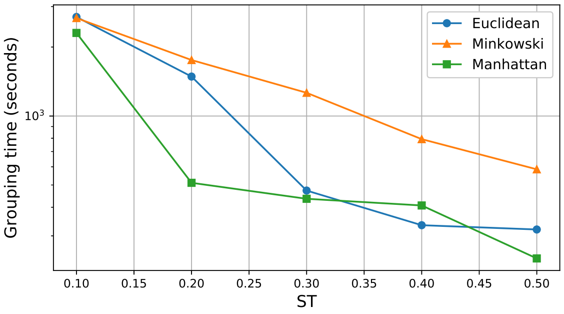


Fig. 10: GENEX bases preprocessing time

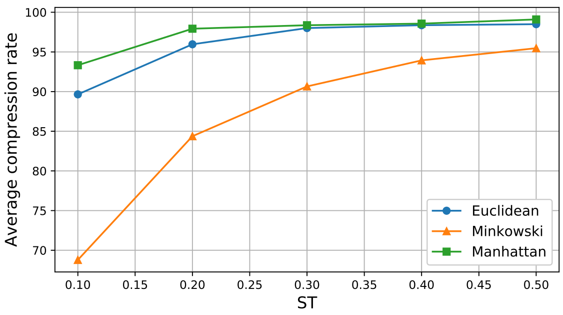


Fig. 11: GENEX bases compression rate

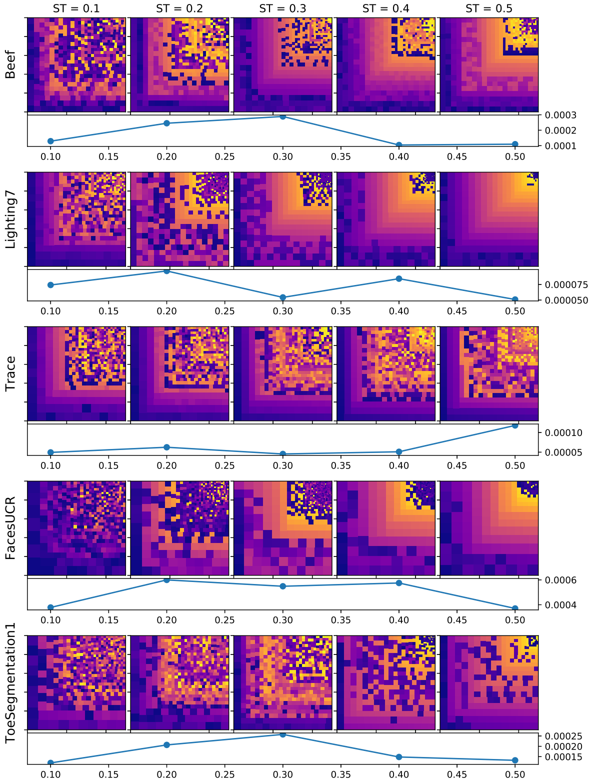


Fig. 12: Cluster distribution for ED.

rate as depicted in Fig. 11. On average, MD yields a smaller number of clusters, thus having the highest compression rate and the fastest preprocessing time. Conversely, Mink generates a larger number of clusters and has the lowest compression rate and the highest preprocessing time. In addition, we visualize the variation in the number of representatives while varying the similarity threshold for five select datasets using our three distances respectively in Fig. 12, 13, and 14. A row in each figure consists of five square subplots and one line subplot. The five square subplots correspond to the varying ST values for preprocessing the dataset, while the line subplot shows the respective average best match error rate of each ST setting. A square subplot consists of multiple cells colored on a blue-yellow spectrum: stronger-blue cells denote clusters of shorter-length sequences while stronger-yellow cells denote clusters of longer-length sequences. The area of a cell is commensurate with the number of sequences in the cluster. For each dataset we sort the clusters by their cardinalities in a decreasing order, then plot the top 600 clusters in each square subplot. The arrangement of the cells is generated using the Python library *squarify* [26], [27]. As a result, the sizes of the cells, starting with the largest from the bottom left corner of the subplot, decrease gradually towards the upper right corner. We call a square subplot ”ordered” if the colors of its cells smoothly transition from blue to yellow going from the bottom left corner to the upper right corner of the subplot. For example, the subplot at ST = 0.5 of the dataset Lighting7 (the last column of the second row) in Fig. 12 is highly ordered. This characteristic implies that the size of a cluster of a specific length is proportional to the number of sequences of that length. In other words, clusters of shorter-length sequences tend to contain more members as there are many more short sequences than longer ones and vice-versa. By correlating this characteristic with the error rate, we observe that a set of

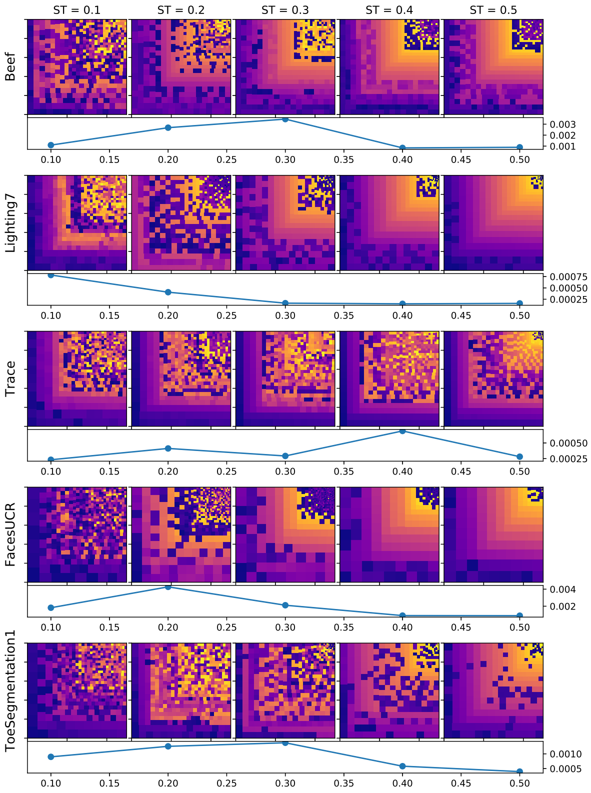


Fig. 13: Cluster distribution for MD.

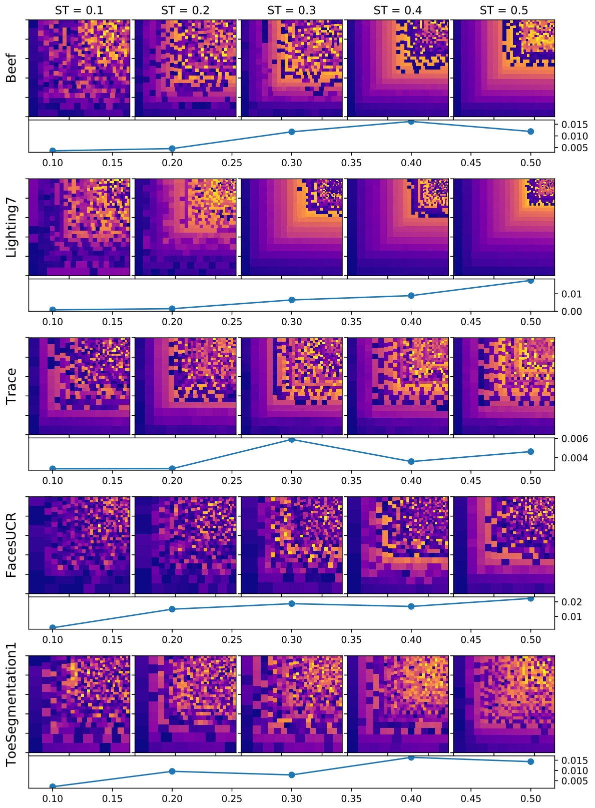


Fig. 14: Cluster distribution for Mink.

more ordered clusters generally achieves a lower error rate. Instances of this phenomena can be seen in datasets Beef, Lighting7, and FacesUCR in Fig. 12 and 13. The reverse is not necessarily true: a lower error rate does not guarantee ordered clusters. A possible explanation of this is that the chosen ST generates highly "even" clusters. Hence their boundaries do not overlap much. Consequently, the representatives becomes a better proxy for comparing similarity between a sample and the members of a cluster.

In summary, this visualization method provides analysts with a valuable tool to evaluate the quality of the clusters for varying similarity thresholds, and guide them towards setting the most appropriate similarity settings for exploring their dataset.

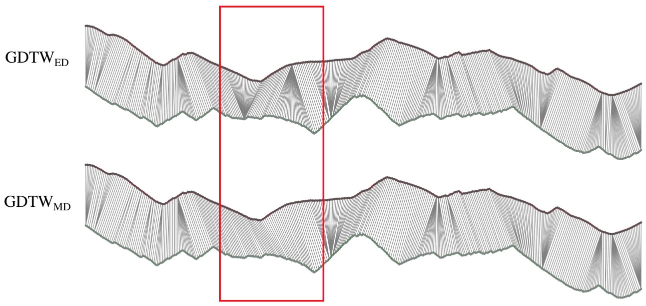


Fig. 15: Alignments of a pair of series generated by $GDTW_{ED}$ and $GDTW_{MD}$

C. Case Study: Botanical Applications

We showcase here the use of GENEX for K-nearest neighbors classification (KNN) on the OSULeaf dataset [8] using $GDTW_{ED}$, $GDTW_{MD}$, $GDTW_{Mink}$. OSULeaf contains one-dimensional outlines of 6 classes of leaves, each of length 427. The series were obtained by color image segmentation and boundary extraction (in anti-clockwise direction) from digitized leaf images of six classes: Acer Circinatum, Acer Glabrum, Acer Macrophyllum, Acer Negundo, Quercus Garryana and Quercus Kelloggii.

The Train set and Test set contain respectively 200 and 242 sequences. For each warped distance, we first determine the value K by performing KNN on a validation set containing 20% randomly selected sequences from the Test set. Then we run KNN on the entire Test set using the value K that gives the best accuracy on the validation set for both GENEX and GDTWSys. As shown in Table III, the accuracy of GENEX is comparable to that of GDTWSys. However, here GENEX is 2 to 3 times faster than GDTWSys. Among the three distances, $GDTW_{MD}$ produces the best accuracy. To see why this is the

TABLE III: K-nearest neighbors results for OSULeaf

Distance	K	GENEX		GDTWSys	
		Acc.	Time (s)	Acc.	Time (s)
$GDTW_{ED}$	1	0.55	9.58	0.55	31.5
$GDTW_{MD}$	5	0.60	12.6	0.60	25.9
$GDTW_{Mink}$	3	0.48	10.3	0.51	24.3

case, we select one leaf shape from the Test set that is incorrectly classified by $GDTW_{ED}$ but it is correctly classified by $GDTW_{MD}$, along with the leaf shape from the Train set that $GDTW_{ED}$ classifies as the nearest neighbor to the previous one. We then plot the alignments generated by $GDTW_{ED}$ and $GDTW_{MD}$ as shown in Fig. 15. The section marked by the red box shows that $GDTW_{ED}$ “collapses” a group of consecutive points in one series into a point on another. This phenomenon distorts the similarity measurement and results in an incorrect classification. Conversely, $GDTW_{MD}$ mitigates this problem by finding more intuitive alignments.

VI. Related Work

Many **similarity distances** have been widely used for mining time series. The ubiquitous Euclidean distance [9], [10] or variants [28] cannot handle misalignments and different length sequences. Although DTW [12] has been successfully used to handle misalignments in many domains, it can produce singularities [29]. To deal with singularities [30] penalizes whenever there is a deviation from a diagonal path, while [31] constrains the warping path by limiting the width along the diagonal. [19] replaces ED with another base distance to constrain the warping path, while [32] ”quantizes” the sequences into the range [0,1] and then places similar points in neighboring bins. GDTW [3] provides a framework to warp a large array of point wise distances. However, neither of these systems provides optimizations to reduce the computation of the warped distances beyond the use of dynamic programming, so to make it practical to mine large time series datasets.

Specific to DTW there are *indexing* techniques [9], [33], [11] and other optimizations such as using squared distance, lower bounds [34], early abandoning of ED and creating envelopes around the query sequence instead of the candidate sequences [15]. [35] embeds the work of [36] to speedup the DTW computation among pairs of time series that are not discarded by other pruning methods. [37] efficiently indexes datasets using a hierarchical K-means tree structure specially designed for DTW. These techniques are orthogonal to our work, and we indeed leverage some of them to optimize similarity search customized to specific warped distances.

Techniques for representing time series with **reduced dimensionality** exist, including Discrete Fourier Transformation (DFT) [4] , Piece Aggregate Approximation (PAA) [38] and Single Value Decomposition (SVD) [39] . The key aspect of these techniques is that they preserve the fundamental characteristics of the data and retrieve highly accurate results. However, most techniques focus on a single distance, tackling efficiency as their main goal and do not handle diverse distances. Conceptually similar [40] uses a local constant embedding which divides the data set into disjoint groups so that the triangle inequality holds within each group by constant shifting, but they have to pre-define the number of members in each group. Our method takes advantage of the specific data distribution in each dataset without having to impose any artificial parameters in defining groups. [41], [25] propose the multi-resolution symbolic SAX representation which can be used to create efficient indices over very large databases using Euclidean distance. However, the technique is optimized for Euclidean distance only thus limited in scope, while our work is geared towards the use of diverse warped distances. Conceptually similar, [42], [1], [16] **reduce data cardinality** by grouping similar sequences. [1] finds part-to-part correspondences between two time series characterized as multi-dimensional trajectories. The resultant dissimilarity is used as input for clustering algorithms. [43] uses DTW averages to create nearest centroid based classifiers for increased efficiency. Conversely, GENEX representatives are selected by construction and DTW is only used for comparing sample sequences to the representatives. [16] only supports DTW while GENEX is the first system to enable analysts to use a variety of warped distances within a single framework. [44] further extends [45] by providing strategies for reducing the training effort required to build an Elastic Ensemble for time series classification. [46] experimentally compares 7 similarity measures for time series classification.

VII. Conclusion

GENEX is a versatile exploratory tool for getting insights into time series datasets using multiple warped distances. Unlike prior work, GENEX provides the first efficient framework for query processing with newly warped point-wise distances on time series collections. The first practical solution for exploring large datasets using multiple robust alignment tools, GENEX yields highly accurate results with response times up to 5 orders of magnitude faster than state-of-the-art systems.

References

[1] S. Hirano and S. Tsumoto, “Cluster analysis of time-series medical data based on the trajectory representation and multiscale comparison techniques,” in *Data Mining, ICDM’06*. IEEE, 2006, pp. 896–901.

[2] E. Ruiz, V. Hristidis *et al.*, “Correlating financial time series with micro-blogging activity,” in *Proceedings of the fifth ACM International Conference on Web Search and Data Mining*. ACM, 2012, pp. 513–522.

[3] R. Neamtu, R. Ahsan *et al.*, “Generalized dynamic time warping: Unleashing the warping power hidden in point-wise distances,” in *2018 IEEE 34th International Conference on Data Engineering (ICDE)*, 2018.

[4] R. Agrawal, C. Faloutsos, and A. Swami, *Efficient similarity search in sequence databases*. Springer, 1993.

[5] K. Chan and A. Fu, “Efficient time series matching by wavelets,” in *15th International Conference on Data Engineering,*. IEEE, 1999, pp. 126–133.

[6] J. Aach and G. Church, “Aligning gene expression time series with time warping algorithms,” *Bioinformatics*, 2001.

[7] D. Gavrila, “The visual analysis of human movement: A survey,” *Computer vision and image understanding*, 1999.

[8] “Ucr time series classification archive,” www.cs.ucr.edu/~eamonn/time_series_data/, 2015.

[9] C. Faloutsos, M. Ranganathan, and Y. Manolopoulos, *Fast subsequence matching in time-series databases*. ACM, 1994, vol. 23, no. 2.

[10] E. Keogh, K. Chakrabarti *et al.*, “Locally adaptive dimensionality reduction for indexing large time series databases,” *ACM SIGMOD Record*, vol. 30, no. 2, pp. 151–162, 2001.

[11] B. Yi and C. Faloutsos, “Fast time sequence indexing for arbitrary lp norms.” VLDB, 2000.

[12] D. Berndt and J. Clifford, “Using dynamic time warping to find patterns in time series.” in *KDD workshop*. Seattle, WA, 1994.

[13] J. Gao, S. Giri *et al.*, “Plaid: a public dataset of high-resoultion electrical appliance measurements for load identification research: demo abstract,” in *1st ACM Conference on Embedded Systems for Energy-Efficient Buildings*. ACM, 2014, pp. 198–199.

[14] E. Keogh and M. Pazzani, “Scaling up dynamic time warping for datamining applications,” in *6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000, pp. 285–289.

[15] T. Rakthanmanon, B. Campana *et al.*, "Searching and mining trillions of time series subsequences under dynamic time warping," in *18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2012, pp. 262–270.

[16] R. Neamtu, R. Ahsan *et al.*, "Interactive time series exploration powered by the marriage of similarity distances," *Proceedings of the VLDB Endowment*, vol. 10, no. 3, pp. 169–180, 2016.

[17] C. Nguyen, C. Lovering, and R. Neamtu, "Ranked time series matching by interleaving similarity distances," in *IEEE International Conference on Big Data (BigData), 2017*. IEEE, 2017, pp. 3530–3539.

[18] J. Kruskall and M. Liberman, "The symmetric time warping algorithm: From continuous to discrete. time warps, string edits and macromolecules," 1983.

[19] H. Sakoe and S. Chiba, "Dynamic programming algorithm optimization for spoken word recognition," *IEEE Transactions on Acoustics, Speech and Signal Processing*, 1978.

[20] S.-H. Cha, "Comprehensive survey on distance/similarity measures between probability density functions," *City*, 2007.

[21] "Genex materials," goo.gl/WTKNTE, 2018.

[22] D. Yankov, E. Keogh *et al.*, "Detecting time series motifs under uniform scaling," in *Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2007, pp. 844–853.

[23] "Genex code," github.com/ctring/genex, 2019.

[24] P. Papapetrou, V. Athitsos *et al.*, "Embedding-based subsequence matching in time-series databases," *ACM Transactions on Database Systems (TODS)*, 2008.

[25] A. Camerra, T. Palpanas, J. Shieh, and E. Keogh, "isax 2.0: Indexing and mining one billion time series," in *Data Mining (ICDM), 2010 IEEE 10th International Conference on*. IEEE, 2010, pp. 58–67.

[26] "Squarify," github.com/laserson/squarify, 2018.

[27] M. Bruls, K. Huizing, and J. Van Wijk, "Squarified treemaps," in *Data visualization 2000*. Springer, 2000, pp. 33–42.

[28] T. Argyros and C. Ermopoulos, "Efficient subsequence matching in time series databases under time and amplitude transformations," in *Third IEEE International Conference on Data Mining, 2003. ICDM 2003.* IEEE, 2003, pp. 481–484.

[29] E. Keogh and M. Pazzani, "Derivative dynamic time warping." SIAM, 2001.

[30] D. Clifford, G. Stone *et al.*, "Alignment using variable penalty dynamic time warping," *Analytical chemistry*, 2009.

[31] Y. Hwang and S. Gelfand, "Constrained sparse dynamic time warping," in *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)*.

[32] G. Al-Naymat, S. Chawla, and J. Taheri, "Sparsedtw: A novel approach to speed up dynamic time warping," in *Proceedings of the Eighth Australasian Data Mining Conference-Volume 101*. Australian Computer Society, Inc., 2009.

[33] E. Keogh and A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowledge and information systems*, 2005.

[34] H. Ding, G. Trajcevski *et al.*, "Querying and mining of time series data: experimental comparison of representations and distance measures," *Proceedings of the VLDB Endowment*, 2008.

[35] D. Silva, R. Giusti, E. Keogh, and G. Batista, "Speeding up similarity search under dynamic time warping by pruning unpromising alignments," *Data Mining and Knowledge Discovery*, 2018.

[36] D. Silva and G. Batista, "Speeding up all-pairwise dynamic time warping matrix calculation," in *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM.

[37] C. W. Tan, G. I. Webb, and F. Petitjean, "Indexing and classifying gigabytes of time series under time warping," in *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 2017, pp. 282–290.

[38] E. Keogh, K. Chakrabarti *et al.*, "Dimensionality reduction for fast similarity search in large time series databases," *Knowledge and Information Systems*, 2001.

[39] D. Wu, A. Singh *et al.*, "Efficient retrieval for browsing large image databases," in *Fifth International Conference on Information and Knowledge Management*. ACM, 1996, pp. 11–18.

[40] L. Chen and X. Lian, "Efficient similarity search in nonmetric spaces with local constant embedding," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 3, pp. 321–336, 2008.

[41] J. Shieh and E. Keogh, "isax: indexing and mining terabyte sized time series," in *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 2008, pp. 623–631.

[42] L. Belbin, "The use of non-hierarchical allocation methods for clustering large sets of data," *Australian Computer Journal*, vol. 19, no. 1, pp. 32–41, 1987.

[43] F. Petitjean, G. Forestier *et al.*, "Faster and more accurate classification of time series by exploiting a novel dynamic time warping averaging algorithm," *Knowledge and Information Systems*, vol. 47, no. 1, pp. 1–26, 2016.

[44] G. Oastler and J. Lines, "A significantly faster elastic-ensemble for time-series classification," in *International Conference on Intelligent Data Engineering and Automated Learning*. Springer, 2019, pp. 446–453.

[45] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining and Knowledge Discovery*, 2015.

[46] R. Giusti and G. E. Batista, "An empirical comparison of dissimilarity measures for time series classification," in *2013 Brazilian Conference on Intelligent Systems*. IEEE, 2013, pp. 82–88.

Rodica Neamtu Dr. Neamtu is an Associate Teaching Professor of Computer Science and Data Science at WPI. Her main research interests are at the confluence of theoretical computer science, data mining, and Big Data. Her work contributes to developing and leveraging groundbreaking techniques for mining time series datasets. She focuses on exploring the theoretical underpinnings of these methods, as well as the practical issues at the heart of Big Data.

Ramoza Ahsan Ramoza Ahsan is a PhD candidate in Computer Science at WPI. Her research interests include data mining, big data analytics and machine learning. She has also worked on association rule mining and data integration problems.

Cuong Nguyen He completed his B.S. degree in Computer Science at WPI and is now a PhD student in Computer Science at University of Maryland, College Park, studying Database Systems.

Charles Lovering After completing his joint B.S. and M.S. degrees in Computer Science at WPI, Charles Lovering is now pursuing his PhD degree in Computer Science at Brown University researching Natural Language Understanding.

Elke Rundensteiner Dr. Rundensteiner is the founding Director of the interdisciplinary Data Science program at WPI and Professor in Computer Science. Her research, focused on big data management, machine learning and visual analytics, has been funded by agencies from NSF, NIH, DOE, FDA, to DARPA and resulted in over 400 publications, patents, and numerous honors and awards. She holds leadership positions in the big data field.

Gabor Sarkozy Dr. Sarkozy is a Professor in Computer Science at WPI. He is also a research fellow at the Renyi Institute of the Hungarian Academy of Sciences. His research interests are in graph theory, discrete mathematics, and theoretical computer science. His research has been funded by numerous agencies including the NSF, NSA and the Clay Institute. He is the founder and director of the Budapest Project Center at WPI.