Matrix Profile XIX: Time Series Semantic Motifs: A New Primitive for Finding Higher-Level Structure in Time Series

Shima Imani
University of California, Riverside
Siman003@ucr.edu

Eamonn Keogh
University of California, Riverside
eamonn@cs.ucr.edu

***Abstract*—Time series motifs are approximately repeated patterns in real-valued temporal data. They are used for exploratory data mining methods including clustering, classification, segmentation, and rule discovery. Their current definition is limited to finding *literal* or *near-exact* matches and is unable to discover higher level semantic structure. Consider a time series generated by an accelerometer on a smartwatch. This data offers the possibility of finding motifs in human behavior. One such example is the motif generated by a handshake. Under current motif definitions, a single-pump handshake would not match a three-pump handshake, even though they are culturally and semantically equivalent events. In this work we generalize the definition of motifs to one which allows us to capture higher level semantic structure. We refer to these as *time series semantic motifs.* Surprisingly this increased expressiveness does not come at a great cost. Our algorithm *Semantic-Motif-Finder* takes approximately the same time as current state-of-the-art motif discovery algorithms. Furthermore, we demonstrate the utility of our ideas on diverse datasets.**

***Keywords*—time series, motif discovery, semantic data, higher-level motif**

I. INTRODUCTION

Time series motifs are approximately repeated patterns in real-valued data [2][24] which are useful for exploratory data mining. In recent years there has been significant progress in the scalability of motif discovery using Euclidean distance [27][28].

If a time series pattern is *conserved* there may exist some high-level mechanism that causes that pattern to be conserved [27]. Euclidean distance is somewhat forgiving of noise, so by analogy with Hamming distance (HD) in strings, if `oscar` and `oskar` appeared in a random string, we could discover this conserved name, in spite of minor spelling variants.

However, consider the following two variations of the name of the Irish poet, `oscar w wilde` and `oscar wills wilde`. Let us embed them into an unpunctuated string. Could we discover them using Hamming distance?

xmargaritazvmargaritexyxoscarwwildeabcoscarwillswildedef

For any motif length of 1 to 8, substrings of "margarit" provide a distance of zero. For lengths of 9 to 15, superstrings of "margarite" provide the minimum distance. Using Hamming distance with "don't cares" is not a solution to this problem, nor is using any variant of string edit distance.

In contrast, suppose there is a conserved *semantic* pattern consisting of two parts, each of length five, separated by between zero and five spaces. If we measure similarity between two occurrences of this pattern HD(prefix,prefix) + HD(suffix,suffix), then HD('`oscar`','`oscar`') + HD('`wilde`','`wilde`') minimizes this function with a score of zero.

We claim that this problem has an analogue in real-valued time series that creates such *time series semantic motifs* in datasets, and these cannot be found with current time series motif discovery tools [2][3][24][27]. While we show many such examples in Section VI, we also preview a few examples below:

- Entomologists use an electrical penetration graph (EPG) apparatus to extract telemetry that reflects insect behaviors. Fig. 1 shows an example of a semantic behavior known in the Asian citrus psyllid (*Diaphorina citri*), a pest of citrus.

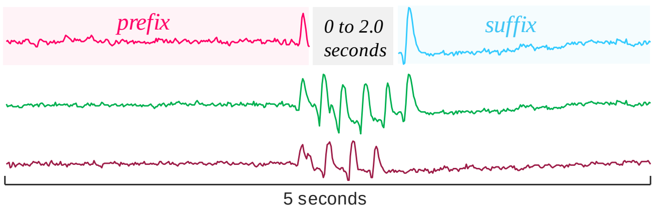

Fig. 1.*top*) An idealized version of Asian citrus psyllid phloem-ingestion behavior. It consists of a highly conserved prefix and suffix, but with zero to two seconds of much more variable behavior in-between. *bottom*) Two realizations of this semantic motif in data obtained by [25].

- In basketball, a free throw is typically comprised of one or more *bounces*, followed by a *throw* after a short (yet variable) period of time. Both "bounce" and "throw" also frequently occur independently, but it is their sequential occurrence, in the right order, within a short period of time that suggests a higher-level semantic motif of *free-throw*.
- Many quotidian human behaviors can be modeled as pairs of sequential events. At a short time-scale, the handshake alluded to earlier is an example. Fig. 2 shows the much longer semantic motif of *doing-laundry*, which we discovered in an electrical power demand dataset.

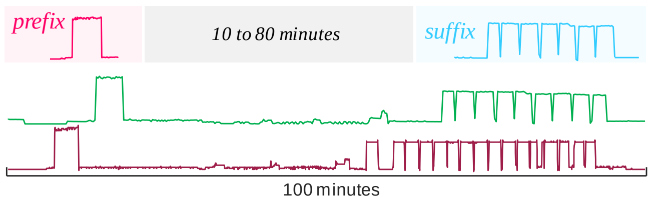

Fig. 2. *top*) An idealized version of the "doing laundry" behavior. Consists of a highly conserved prefix (washing machine) and suffix (dryer), but with tens of minutes of more variable behavior in-between. *bottom*) Two realizations of this semantic motif in data obtained by [17].

These are examples of *known* ordered occurrences of events, which can be modeled by semantic motifs. However, we are interested in the discovery of these *unknown* event sequences. Once known, they can be discovered elsewhere using a handful of existing search techniques.

The rest of this paper is organized as follows. In the next section, we introduce all necessary definitions and notation. In Section III, we formally introduce time series semantic motifs. We review related work in Section IV, before discussing our algorithm for Semantic Motif discovery in Section V. In Section VI, we perform an empirical evaluation. Finally, we summarize our results and outline future work in Section VII.

II. DEFINITIONS AND NOTATION

We begin by describing the necessary definitions and notation. The data type of interest is *time series*:

Definition 1 (time series): A time series T of length n is a sequence of real-valued numbers t_i: $T = t_1, t_2, \dots, t_n$.

We are primarily interested in the behavior of *local* regions. A local region of time series is called a *subsequence*:

Definition 2 (subsequence): A subsequence $T_{i,m}$ of a time series T is a continuous ordered subset of the values from T of length m starting from position i . $T_{i,m} = t_i, t_{i+1}, \dots, t_{i+m-1}$, where $1 \le i \le n - m + 1$.

Almost all algorithms in the literature use the following definition of *time series motifs*:

Definition 3 (motif): A time series motif is the most similar subsequence pair of a time series. Formally, $\mathrm{T_{a,m}}$ and $\mathrm{T_{b,m}}$ is the motif pair of length m iff $\mathrm{dist(T_{a,m}, T_{b,m}) \le dist(T_{i,m}, T_{j,m})}\ \forall\, \mathrm{i, j} \in [1, 2, \dots, \mathrm{n - m + 1}]$ where $\mathrm{a \neq b}$ and $\mathrm{i \neq j}$, and dist is a function that computes the z-normalized Euclidean distance between the input subsequences.

We can store the distance between a subsequence of a time series with all the other subsequences from the same time series in an ordered array called *distance profile* [16].

Definition 4 (distance profile): A *distance profile* $D \in \mathbb{R}^{n-m+1}$ of a time series T and a given query $T_{i,m}$ is a vector which stores $dist(T_{i,m}, T_{j,m})\ \forall\, j \in [1, 2, \dots, n - m + 1]$.

III. SEMANTIC MOTIF DEFINITION

Our task reduces to finding ordered pair of subsequences within a long time series. As hinted at in Fig. 1 and Fig. 2, each pattern consists of three parts: *prefix*, *don't-care* and *suffix*. The prefixes are mutually similar, as are the suffixes. The shapes of the don't-care regions are arbitrary and are ignored. Moreover, we allow the *lengths* of the don't-care regions to be different. Each can have any value in the range 0 to r, where r is a user specified constraint. Formally, a pattern meeting all these requirements is a time series *semantic motif*:

Definition 5 (semantic motif):

$T_{a,m}$ and $T_{b,m}$ is the semantic motif pair iff

$$(a, b) \sim argmin_{i,j}\{dist(T_{i,m}, T_{j,m}) + dist(T_{k,m}, T_{\ell,m})\}$$

where

$$1 \le i \le n - 2(r + m) + 1$$
$$i + r \le k \le n - (r + 2m) + 1$$
$$k + m \le j \le n - (r + m) + 1$$
$$j + m + r \le \ell \le n - m + 1$$
$$\forall\, i, j, k, \ell \in [1, 2, \dots, n - m + 1]$$

Where m is the length of the semantic motif, $i \neq j, k \neq \ell$ and $dist$ is a function that computes the z-normalized Euclidean distance between the input subsequences. r is the maximum length of don't-care region and n is the length of time series. The first term i.e. $dist(T_{i,m}, T_{j,m})$ is the distance between prefixes and the second term i.e. $dist(T_{k,m}, T_{\ell,m})$ is the distance between suffixes with respect to the ordered triple (prefix, don't-care, suffix). We will explain this definition in more details in Section V.

We can use the semantic motif pair distance for each semantic motif pair to form a *semantic matrix profile*:

Definition 6 (semantic matrix profile): A semantic matrix profile is a meta time series SMP that stores the distance between each semantic motif and its nearest neighbor.

By Definition 5, the location of the (mutually) lowest pair of points corresponds to the top-1 semantic motif.

Consider the following example, which demonstrates one possible use of semantic motifs. There is a known feeding behavior in certain insects that can be described as prefix , and a suffix , with a don't-care region in-between [25]. We asked an entomologist to provide a handful of data snippets from an insect, *some* of which contain this feeding behavior. Fig. 3 shows two attempts at clustering this data.

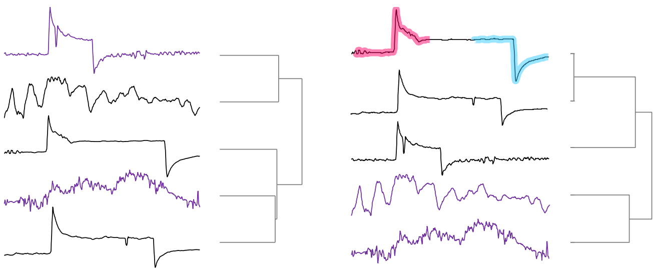

Fig. 3. Two clusterings of insect data that include samples of a behavior called "non-ingestion-C". *left*) The clustering produced by Euclidean distance does not correctly group the behaviors, but the semantic motif distance (*right*) does.

We often care about the ordering of events. This is reflected in the requirement that the index of a suffix be greater than its corresponding prefix. For example, "`..Oscar loved..`" and , "`..loved Oscar..`" are semantically different (in the first Oscar is the *subject*, in the second Oscar is the *object*). There may be a handful of cases in which order does not matter, for example "`..shiny red..`" and "`..red shiny..`". However this generalization is trivial if needed.

We want to avoid very long don't-care regions, as this helps mitigate the discovery of spurious motifs. For example, consider a person who routinely has medical checkups in early January and June. That same person may adjust their clock twice a year for daylight savings time. Clearly it would be bizarre to view `ClockAdjust <don't care> Exam` as a semantic motif, as the two events happen months apart. However, suppose our individual follows the advice of the U.S. Fire administration, and uses the clock adjusting as prompt to change the batteries in her smoke detectors. Then `ClockAdjust <don't care> ChangeBattery` is a meaningful semantic motif we would hope to discover.

In order to avoid arbitrarily long semantic motifs, we choose a threshold r to be the maximum length of the don't-care regions for each subsequence. The best choice of r is domain dependent, but we envision it being at most a small multiple of the length of the prefix/suffix.

IV. Related Work

In recent years there has been an explosion of interest in time series motifs [20][12][1][9][10][13]. While there has been significant progress in scalability of motif discovery, there has been much less progress in expanding the expressiveness of the definition of 'motif'. There have been proposals to replace the Euclidean distance with Dynamic Time Warping [23], length-invariant Euclidean distance [8], or other L_p norms. However, none of these generalizations offer the pattern of expressiveness we propose.

A. *Dismissing Apparent Solutions*

It is important to dismiss *apparent* solutions to this problem before introducing our technique:

Dynamic Time Warping (DTW), while useful in many time series data mining tasks, is not a solution to the task at hand. DTW is able to compensate for small local misalignments [23]. Consider Fig. 1, DTW cannot map four peaks to six peaks. It must attempt to "explain" all data, while semantic motifs allow us to ignore some data.

There are many techniques to search for *known* patterns with the ability to support some "don't-care" regions. These include weighted Euclidean distance, (real-valued versions of) longest common subsequence and variants of Markov models. These ideas have utility for *one-to-all* similarity search. However, we are searching for unknown patterns, a much harder *all-to-all* case. However, we are interested in unsupervised pattern matching in the form of an all-to-all case. It is not clear that the aforementioned approaches could be adapted to create meaningful results for these tasks. For example, weighted Euclidean distance allows the use of don't-care regions, but the length of each don't-care region must be specified ahead of time. In contrast, semantic motifs only require an upper bound for the length of the don't-care region.

Finally, even if we could adapt any of the existing approaches, it is not clear how we could make them scalable. Each takes *at least* $O(m)$ time to process one comparison, and motif discovery requires $O(n^2)$ comparisons to be computed or admissibly pruned, giving us a time complexity of $O(mn^2)$. In contrast, our approach is just $O(n^2)$. When m is in the thousands, this difference is three orders of magnitude.

There are "fast" methods to find *classic* motifs using discretization of data into "words" of length w, hashing of the discretized words, and a post-processing of the hash buckets to refine the motifs [5]. All such methods in the literature are approximate and require careful tuning of several parameters. Yeh et al [27] empirically shows that the discretization overhead adds a huge constant factor to the execution time, and in practice these approximate methods may not be much faster than exact methods.

We will show in the next section (and in our experimental section) that the "classic" approach to motif discovery [28] is unable to discover repeated structure that semantic motifs can find. We do not believe that this is achievable through simple modification to existing motif discovery algorithms.

V. Finding Semantic Motifs

In this section we introduce an exact algorithm *Semantic-Motif-Finder*, which allows us to find all semantic motifs in a time series of length n in $O(n^2)$ operations.

A. *Semantic-Motif-Finder Algorithm*

Our goal is to calculate the semantic motif pair (Definition 5). We solve this problem by computing the semantic matrix profile described in Definition 6. We then find the semantic motif pair by locating the two lowest values in the semantic matrix profile.

The inputs to the algorithm *Semantic-Motif-Finder*, are a time series T, a prefix/suffix length s and a maximum don't-care length r. The outputs are a semantic matrix profile *SMP* and semantic matrix index *SMI*. The semantic motif pair is located at the minimum value of the *SMP*, and the *SMI* shows the location of the pair.

Let us apply our ideas to the example of examining a working day in the life of a graduate student through accelerometer data recorded by her phone. She reaches the elevator on her way to her office on the fourth floor, where she typically starts working around 9:00 am. Upon reaching her office, she realizes that she forgot to buy coffee. She skips the busy elevator and descends two flight of stairs to reach the coffee house which is located on the second floor[1]. After buying coffee, she returns to her office via the elevator and continues her work. Both elevator trips are annotated in Fig. 4.

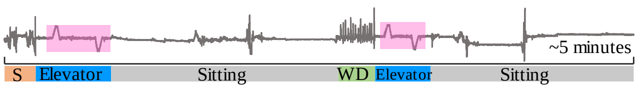

Fig. 4. The hip-mounted X-axis acceleration of a graduate student standing, using elevator, sitting, walking down stairs, using elevator and sitting again. The data is real but edited for visual brevity [18].

The pink highlighted regions in Fig. 4.*top* contain a semantic motif corresponding to the ascending-in-elevator behavior. As shown in Fig. 5, this semantic motif contains a "bump" caused by the acceleration of the rising elevator, i.e. prefix. It also contains a relatively flat region, the length of which is proportional to the number of floors the elevator ascends, i.e. the don't-care region. Finally, the motif has a "valley" caused by deceleration when the elevator stops, i.e. suffix.

The don't-care region for the first subsequence shows the elevator ascending four floors. In the second subsequence, the elevator only ascends two floors. The prefixes and suffixes are mutually similar, while the don't-care regions differ, with one being six seconds longer.

[1] We are using the American convention that the floor at street level is the *first* floor, the one above is the *second* floor etc.

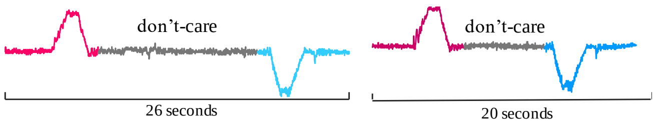

Fig. 5. A semantic motif pair related to the ascending elevator in the time series shown in Fig. 4.

The starting location of the semantic motif is the same as the starting position of its prefix. The suffix starting position could be anywhere as follows:

$$\text{psp} + s < \text{suffix starting position} < s + r + \text{psp}$$

where psp is the prefix starting position, r is the maximum length of the don't-care region and s is the length of prefix region.

We want to compute the distance between semantic motifs beginning at indices a and b with prefix and suffix of length s. We compute the distance as follows:

$$Semantic-motif distance(a,b) =: dist(prefix(a), prefix(b)) + dist(suffix(a), suffix(b)) \quad (1)$$

Recall that $dist$ is a function, which computes the z-normalized Euclidean distance between two input subsequences.

The first part of Equation 1 can be written as:

$$dist(prefix(a), prefix(b)) = dist(\text{T}[\text{a:a}+\text{s}], \text{T}[\text{b:b}+\text{s}])$$

The second part of Equation 1 is more challenging due to various possible starting positions for the suffix as hinted in Fig. 6.

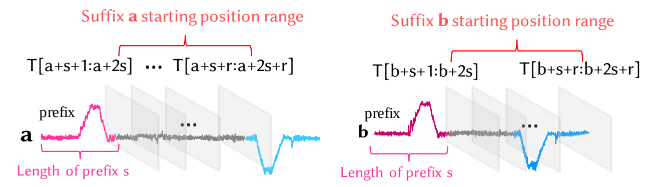
Suffix a starting position range
T[a+s+1:a+2s] ··· T[a+s+r:a+2s+r]
prefix
a
Length of prefix s
Suffix b starting position range
T[b+s+1:b+2s]
T[b+s+r:b+2s+r]
prefix
b
Length of prefix s

Fig. 6. The semantic motif pair corresponding to the elevator acceleration for the time series shown in Fig. 4. The red/blue regions are the prefixes/suffixes with length s. Suffix a/suffix b can start in any position after the prefix region and before the maximum length of don't-care region. (The gray panels show the legal possible staring locations of the suffix.)

For the suffix component of Equation 1, i.e. $dist(suffix(a), suffix(b))$ we compute a *set-of-suffix-distances* as follows.

$$\begin{gathered}\text{set-of-suffix-distances} = \\ \{dist(\text{T}[\text{a}+\text{s}+1:\text{a}+2\text{s}], \text{T}[\text{b}+\text{s:b}+2\text{s}]) \\ dist(\text{T}[\text{a}+\text{s}+2:\text{a}+2\text{s}+1], \text{T}[\text{b}+\text{s}+1:\text{b}+2\text{s}+1]), \ldots \\ dist(\text{T}[\text{a}+\text{s}+\text{r:a}+2\text{s}+\text{r}], \text{T}[\text{b}+\text{s}+\text{r:b}+2\text{s}+\text{r}])\} \\ \forall\, a, b \in T \text{ and } b \neq a\end{gathered}$$

From the set-of-suffix-distances, we choose the value which minimizes Equation 1. Here the set-of-suffix-distances contains r^2 members. Further, computing the value for each member of the set-of-suffix-distances takes $O(s)$ operations. The time complexity to compute the semantic motif distance between a and b is then $O(sr^2)$. In addition, to find the best semantic motif pair, we must perform the above calculation for every choice of a and b in the time series, which means that the overall time complexity is $O(n^2r^2s)$.

This naïve way to compute the semantic motif pair is untenable. However, we introduce an algorithm that can drastically reduce this untenable time complexity.

Now we are in a position to explain our algorithm, which is outlined in TABLE I. In lines 2 to 4, we calculate the distance profile (Definition 4) for each position in the time series. We call this matrix D. The distance between subsequence $T_{i,s}$ and the time series is D_i, which is located at the i_{th} row of the matrix D as shown in Fig. 7.

Assume that we want to find the semantic motif pair corresponding to position i. The distance between $prefix(T_i)$ and the time series is just D_i. The distance between $suffix(T_i)$ and the time series is the column-wise-minimum of sub-matrix $D_{i+s+1}, \ldots, D_{i+s+r}$ (i.e. blue region in Fig. 7) which we call M, in line 5.

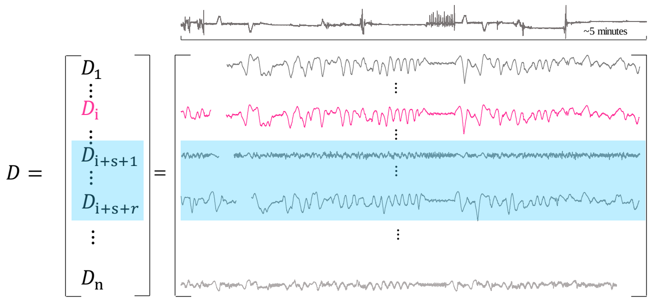

Fig. 7. The data in Fig. 4. The row of matrix D are the distance profile (Definition 4) which correspond to each position in the time series.

Fig. 8 shows the distance profile D_i, M_i and the green highlighted region corresponds to the prefix and suffix of acceleration of the elevator at index i.

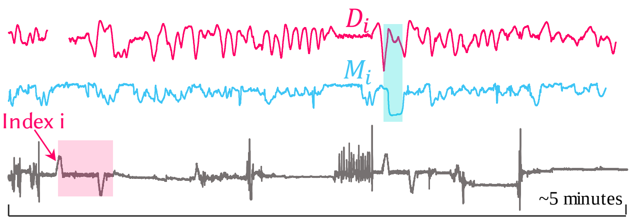

Fig. 8. *top*) The distance between subsequence $T_{i,s}$ and the time series. *middle*) the column-wise-minimum of sub-matrix $D_{i+s+1}, \ldots, D_{i+s+r}$, called M_i. *bottom*) The time series shown in Fig. 4.

In order to handle variable length don't-care regions for each suffix, we create an "envelope" of size r over M_i that we call N_i, as shown in line 6 of TABLE I.

The value of semantic matrix profile at position i, SMP_i is the minimum value of the summation of D_i and N_i, line 7. We add the corresponding index of the minimum value to the semantic matrix index SMI. SMP at index i shows the position of the closest semantic motif in the time series T to the semantic motif at position i, and the value shows the distance of this pair. Finally, we return the semantic matrix index SMI and the semantic matrix profile SMP.

The semantic motif pair can be found by locating the two lowest values in the SMP. Note, the definition of the semantic motif pair is symmetric with respect to the indices, which means if we find the semantic motif pair of index i at position j, then the semantic motif of index j is located at position i.

Fig. 9 shows the result for calculating summation of D_i and N_i for the location i in the time series (green meta time series).

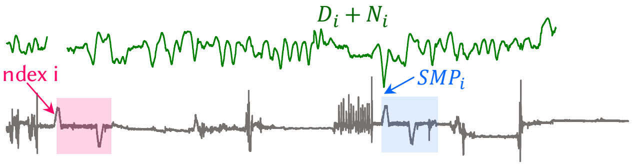

Fig. 9. *top*) The meta time series that is the summation of D_i and N_i. The minimum of the meta time series is the location of semantic motif pair corresponding to index i. *bottom*) The time series. The blue block is the semantic motif pair of the red block.

Note that the minimum of the green meta time series is SMP_i , showing the location of the semantic pair corresponding to the index i.

TABLE I. SEMANTIC-MOTIF-FINDER ALGORITHM

```
Algorithm: Semantic-Motif-Finder
Input: time series T, suffix/prefix length s, maximum don't-care length r
Output: semantic matrix profile SMP, semantic matrix index SMI
1  n ← length(T)
2  for i ← 1 to n - s + 1
3      D[i, :] ← distance profile(T [i:i+s-1], T )   // Def 4
4  end
5  M ← column-wise-min(D [s+1:end-s+1,s+1:end-s+1]) // r col's
6  N ← row-wise-min(M)                               // r rows
7  (SMP, SMI) ← min (D+N )
8  return SMP, SMI
```

B. Semantic-Motif-Finder Algorithm Complexity

The time complexity of our proposed *Semantic-Motif-Finder* is $O(n^2)$. Computing all the distances in the sliding window takes $O(n^2)$ and we require an additional $O(n)$ for finding the minimum in the sliding window of size r. Hence the total time complexity is $O(n^2)$. In practice we do not need to keep the whole matrix D. In each iteration we need to keep the submatrix $D_i, \ldots, D_{i+s+r}$ which is sufficient to update SMP_i. As a result, the space complexity in our slightly more sophisticated implementation is just $O(n(r+s))$.

As a practical matter, memory is *not* a bottleneck for this or any other motif discovery algorithm [27]. We can easily handle time series data with lengths in the hundreds of millions in the main memory of an off-the-shelf desktop machine, without having to resort to disk access.

C. Semantic-Motif-Finder Algorithm with Different Lengths

We also implemented a more expressive version of the Semantic-Motif-Finder. In this version the user can choose different lengths for prefix and suffix as shown in TABLE II. However, this algorithm needs an extra parameter in comparison to the Semantic-Motif-Finder algorithm. For example, setting the (prefix, suffix) to (100, 200) is not the same as (200, 100). For completeness, we will show one experiment of using this algorithm in Section VI , however in most cases requiring the prefix and suffix to be the same length does not appear to be a significant limitation.

TABLE II. SEMANTIC-MOTIF-FINDER WITH DIFFERENT SUFFIX AND PREFIX LENGTH

```
Algorithm: Semantic-Motif-Finder with variable length
Input: time series T, prefix length p,
suffix length s, maximum don't-care length r
Output: semantic matrix profile SMP, semantic matrix index SMI
1  n ← length(T)
2  for i ← 1 to n - p + 1
3      Dp[i, :] ← distance profile(T [i:i+p-1], T ) // Def 4
4  end
5  for i ← 1 to n - s + 1
6      Ds[i, :] ← distance profile(T [i:i+s-1], T ) // Def 4
7  end
8  M ← column-wise-min(Ds [s+1:end-s+1, s+1:end-s+1])//r col's
9  N ← row-wise-min(M)                                // r rows
10 (SMP, SMI) ← min (Dp+N )
11 return SMP, SMI
```

D. The Expressiveness of Semantic Motifs

When introducing a new primitive, it is natural to ask how its expressiveness relates to the current state-of-the-art. The ideal case is if the new representation subsumes an existing method that is well understood and widely adopted. For example, constrained Dynamic Time Warping subsumes Euclidean distance. Can we make similar claims for semantic motifs? In particular, do semantic motifs subsume classic motifs when r is set to zero?

The answer is *almost*, and the difference is quite telling. In the classic case, all subsequences of length m are z-normalized [27][28]. But for semantic motifs, the prefix and suffix are independently z-normalized. Thus, these two cases are not *quite* equivalent, and may return different results.

Note that we *can* force these two cases to be the same, by a small modification of our code (in line 3 of TABLE I) that forces the prefix and suffix to be normalized together (see Appendix A and *then* Fig. 23). However in most circumstances that is not a good idea. To see why, let us use a toy dataset. We define a "camelback" pattern as two consecutive parabolic sections, each of length 200. However, the heights of the parabolas can differ by up to 1/3. The example shown in Fig. 10.top show a camelback pattern embedded within a sequence of noisy sine waves.

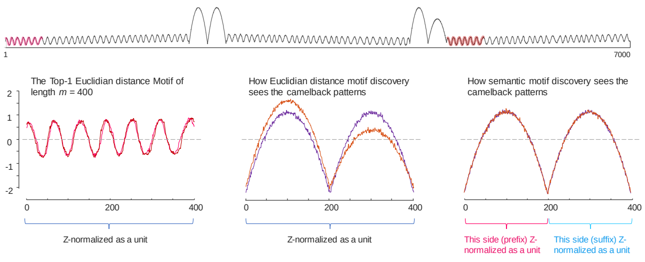

Fig. 10. *top*) A synthetic dataset contains two examples of the camelback pattern embedded in some noisy sine waves. *Bottom*: *left* to *right*) The top-1 classic motif. The camelbacks normalized together, the camelbacks normalized in the semantic motif representation.

To the eye, the camelback patterns seem like the obvious classic motif of length 400, but this is not the case. Fig. 10.*bottom.center*, which shows both pairs of camelbacks with z-normalization, as Euclidian distance "sees" them. It is clear that they will produce a large Euclidean distance. In contrast, Fig. 10.*bottom.right* shows the same data and how they will appear to the semantic motif distance, as being almost identical. This toy example is contrived, but we will show several real examples that exhibit the same issue in Section VI.

As another example, let us revisit the elevator data introduced in Section V. This example had the cellphone perfectly vertical in the student's back pocket. Suppose that on the second ascent, the student quickly removed the phone to check the time, and when she replaced it, it was askew in her pocket at 73 degrees. The different pattern caused by this motion during the ride is not an issue, and our "don't care" region will take care of it. But the new phone angle would result in the second bump being only 2/3 the height of the first, making the elevator motif harder to find, if we did not do independent normalization of the prefix and suffix.

E. On the Robustness of our Definition

Before experimentally evaluating our ideas, we take the time to ward off a possible misunderstanding. The classic motif definition is quite robust to mismatches between user-provided subsequence length, and the true motif length. For example, suppose there is a single motif in a dataset, and it is *exactly* ten minutes long. Even if the user sets her desired subsequence length to anywhere between six to twelve minutes[2], she is very likely to find the motif [27]. We inherit this robustness as our definition essentially combines two independent classic motif discovery's (with a temporal constraint). For example, suppose that there is a conserved two-part behavior that is manifested as a four-second-long pattern, followed by a seven-second-long pattern. We are likely to be able to find this if we set our single *prefix*/*suffix* length to be anywhere in the range of three to eight seconds. Critically, we do *not* need both parts of the two-part behavior to be of the same length. This robustness is tested by our choice of datasets. For example, we could not contrive to make both parts of a two-part bird song, or seal behavior equal length.

Likewise, we reiterate that the *don't-care* parameter is an upper bound to a range, it does not need to be precisely set.

VI. Expermimental Evaluation

To ensure that our experiments are easy to reproduce, we have created a website that contains all data/code/raw spreadsheets for all the experiments [19]. Our commitment to reproducibility extends to all the examples shown in the previous sections.

Before showing formal experimental evaluations of the scalability and robustness of Semantic-Motif-Finder algorithm, we will show some examples of semantic motifs in diverse domains to demonstrate the generality of our ideas.

A. Marine Mammals

We consider a seal behavior dataset [15] that contains motion recordings of seventy-two seals, belonging to four species. As Fig. 11 shows, this dataset contains data from a wearable accelerometer mounted on the seal's back.

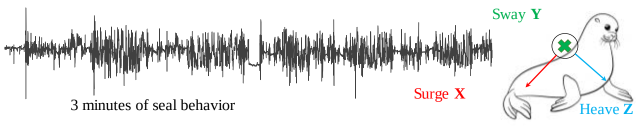

Fig. 11. A short snippet of seal behavior hints at the complex structure of the data.

We choose seals because, as Fig. 11 hints, the data is very complex and challenging, and because pinnipeds are known to be very intelligent, and thus good candidates for having higher level complex semantic behaviors.

We had previously noted that classic motif discovery often finds highly conserved motifs in the range of about two to sixteen seconds in these datasets, but at lengths beyond that, the motifs returned seem "random". Given the expressiveness of semantic motifs, we increased our ambition and searched for motifs with prefix/suffix length of 16 seconds and the don't-care region of length 16 seconds. That is to say, motifs that are expressed somewhere over the 32 to 48 seconds range. Fig. 12 shows the top motif discovered in a seven-minute snippet of Australian Fur Seal (*Arctocephalus pusillus*) or AFS. Note classic motif discovery algorithms may fail here. The variable length of the spurious don't care regions differ by four seconds, which places the two highly featured atomic events out of phase, defeating the Euclidean distance.

The semantic motif shown in Fig. 12.*top* is clearly highly conserved, but what is it? A recent observational study of AFS describes one behavior in great detail: "*hunting involves actively pursuing prey, before accelerating the whole body to seize prey in the teeth… …once captured in the jaws, prey items were manipulated and re-oriented using further mouth movements or chews so that they could be swallowed head first*" [11]. This description fits the observed behavior, and our dataset is accompanied by observer's notes, which confirm our discovery [15].

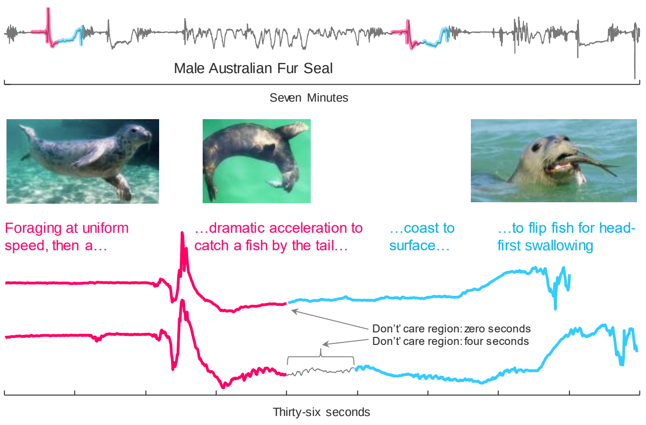

Fig. 12. *top*) A seven-minute snippet of AFS behavior. *middle*) Three phases of the hunting behavior. *bottom*) The top semantic motifs in this dataset corresponds to two instances of fish capture/consumption.

We discovered many other semantic motifs in these datasets, the vast majority of which cannot be discovered by classic motif discovery. Fig. 13 shows just two more examples that illustrate the two invariances that semantic motif discovery offers over classic motif discovery.

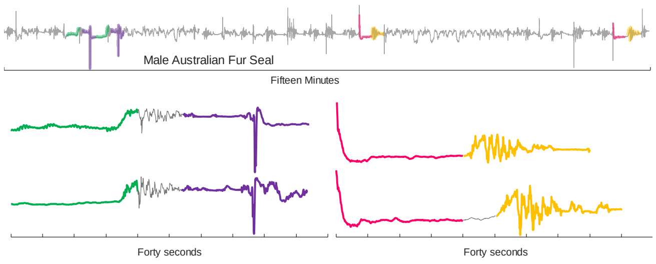

Fig. 13. *top*) A fifteen-minute snippet of AFS behavior. The top two semantic motifs illustrate the two reasons classic motif discovery often fails. *left*) The

[2] Note this robustness is asymmetric, the user provided length can be a *little* longer, or a *lot* shorter than the true conserved length.

don't-care region is the same length in both instances, but not conserved between instances. *right*) The don't-care region is longer in one instance, putting the conserved prefixes and suffixes out of phase with each other.

B. *Semantic Motifs in Bird Calls*

Bird calls have long been used as a testbed for motif discovery, using an MFCC representation [7]. Birds are rewarded for conserving their calls, as "*more typical versions of song types function better in male–female communication*" [14]. However, researchers have only successfully applied motif discovery to very simple bird calls. For example, the calls of Chickadees are often transcribed into guidebooks as simply "`fee-bee`" and "`chick-a-dee`".

Researchers studying song complexity note that songs can be more intricate. For example, song sparrows have calls that "*begin with three clear notes and then a complicated and* <u>*variable*</u> *series of single notes, ... (ending with) trills*" (our emphasis) [26]. We show that classic motif discovery has great difficulty discovering such songs, whereas semantic motifs do not.

In Fig. 14 we show two snippets of the song of a White-browed Scrub Robin (*Cercotrichas leucophrys*) recorded in Zimbabwe. This particular bird's call lasts about 2.5 seconds, and consists of two descending notes (A), several short chirps (b), followed by a short rising and falling note (C). In just one minute we heard the following variants: AAbbbC, AAbbC, AAbbbbC. Such calls can be represented as semantic motifs if we consider A to be the prefix, C the suffix, and a variable number of b appearing between A and C as the don't-care regions.

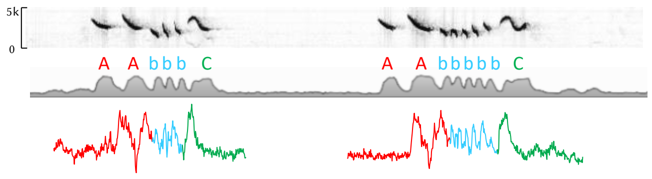

Fig. 14. Six seconds of bird song. *top*) A spectrogram. *middle*) An energy profile (i.e. the volume). bottom) The 10th MFC Coefficient reveals the similarity in time series space.

If we attempt classic motif discovery with motifs of 2.5 seconds, these variants are different enough that they are *not* discovered as motifs using classic definitions of motifs [5]. Fig. 15 shows the motif pair (red regions) for the classic motif discovery for the bird song time series.

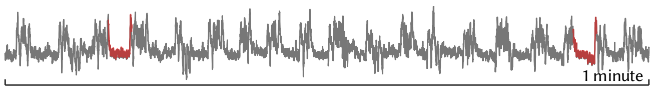

Fig. 15. A snippet of bird song. The red highlighted regions are the results when using the classic motif discovery.

A careful audio review reveals that the motifs discovered do not correspond to the bird song. The motifs discovered correspond to the approximately U-shape pattern formed by a relatively quiet period in-between two louder sounds.

If we search for semantic motifs with a prefix/suffix length of one second and a don't-care region of at most 0.5 seconds, then as shown in Fig. 16, we successfully discover the Robin's call as the top motif.

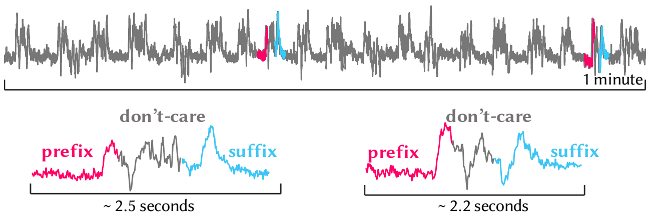

Fig. 16. *top*) The semantic motif discovery result for the time series shown in Fig. 15. *bottom*) The semantic motif pair, with one second of prefix/suffix and 0.5 seconds of the don't-care region.

C. *Semantic Motifs in Human Speech*

The example above shows that semantic motifs of the form prefix<*don't-care*>suffix exist in birdcalls, presumably because this structure is pleasant to the hens [26]. Should we expect such structures in human communication? In fact, this structure is *symploce* (or *complexio*), a rhetorical device known to make text/speech more pleasant and easier to remember. For example, St. Patrick's prayer makes extensive use of the device: "…Contra *combustionem*, Contra *demersionem*,…" [21]. While it is trivial to find such examples in ASCII text strings, in this section we will test the ability of our algorithm to find semantic motifs in low quality representations of speech.

Consider the classic poem "The Raven" by Edgar Allen Poe. The familiar first verse of this poem is:

> *Once upon a midnight dreary, while I pondered, weak and weary,*
> *Over many a quaint and curious volume of forgotten lore—*
> *While I nodded, nearly napping, suddenly there came a tapping,*
> *As of someone gently rapping, rapping at my chamber door.*
> *"'Tis some visitor," I muttered, "tapping at my chamber door—*
> *Only this and nothing more."*

Can we find examples of semantic motifs in this poem? We use the audio of this poem [22] to create a time series as shown in Fig. 17.*top*.

There are many short motifs in this poem corresponding to "*chamber door*", "*at my*", "*rapping*", etc. However, we hope to find higher-level structure corresponding to the poetic device of *symploce*. As shown in Fig. 17.*top*, it is not clear whether *any* structure exists in this data.

Using our proposed *Semantic-Motif-Finder* algorithm, we *can* find meaningful semantic structures. Fig. 17.*bottom*. shows the semantic motif pair.

As shown in Fig. 17 the semantic motif pairs are different in the don't-care regions, as one has two extra words (i.e. "*it is*").

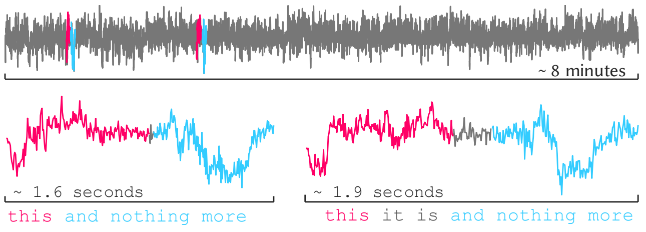

Fig. 17. *top*) The time series corresponding to the 10th coefficient of MFCC using the audio of "The Raven" poem [22] with the semantic motif pair, prefix (red) and suffix (blue) highlighted. *bottom*) A zoom-in of the semantic motif pair.

The prefix and suffix are not as well preserved as in some of the other examples (c.f. Fig. 12 and Fig. 13), however we deliberately used a poor quality audio recording and an unsuitable MFCC coefficient (relative to the speaker pitch) to demonstrate that Semantic-Motif-Finder algorithm works with challenging data.

We also ran an experiment with the Semantic-Motif-Finder algorithm with different suffix and prefix length discussed in Section V. We chose the pair (prefix length, suffix length) to be (2.4 seconds, 1.6 seconds). The result is shown in Fig. 18. Note that for this experiment we need to specify prefix and suffix length separately.

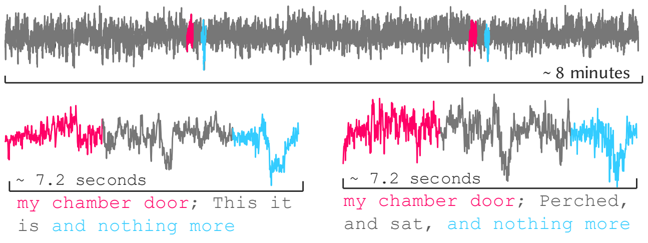

Fig. 18. *top*) The time series shown in Fig. 16 with the semantic motif pair, prefix (red) and suffix (blue) highlighted. *bottom*) A zoom-in of the semantic motif pair.

D. *Effectiveness and Efficiency*

In this section we show that semantic motifs can discover repeated structure that would evade classic motif discovery [5], and every published variation of it that we are aware of [2][3][23][24][25][8].

To create an initial test set, we embed real patterns extracted from [4] into a smoothed random walk. The embedded data consists of motion capture data where a fork was used to feed a dummy. The original data contains x, y, z. Here we only use the x-component.

The patterns themselves are atomic. For example, "`spear a carrot`" and "`twist a noodle`" both appear in the data. From a single class, we take two exemplars and concatenate them with two exemplars from a different class, to produce a higher-level semantic eating event. Fig. 19 shows an example.

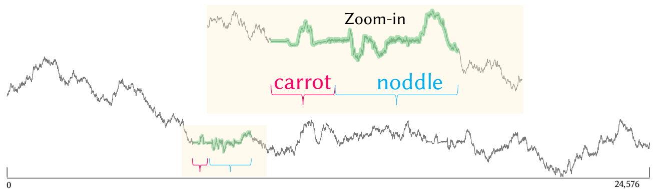

Fig. 19. An example of a dataset with two examples of the motif "`eating carrots and noodles`" embedded in otherwise unstructured data. Without the color highlighting, the embedded patterns are not obvious.

Like many of the motifs discovered in this work, this motif has a higher-level semantic meaning, *`eating-carrot-noodles`*, a popular dish in trendy Thai fusion restaurants.

Note that the events here are only approximately the same length, they reflect the real behaviors of several humans dealing with different size morsels in different positions on a plate etc.

In order to compare Semantic-Motif-Finder algorithm with the classic motif discovery algorithm, we performed the following experiment. We created a random walk with different lengths and embedded eating-carrot-noodles behavior as shown in Fig. 19. The objective of this experiment was to discover which algorithm is more successful in finding the embedded motifs in the time series.

For the purpose of comparison, we define the *success rate* of each algorithm as the number of times it locates embedded motifs over the total number of trials. We allow classic motif discovery to consider three different subsequence lengths, with each choice of subsequence length reported as a single experiment.

$$\textit{subsequence length}\begin{cases}\textit{length of carrot behvior}\\ \textit{length of noodle behvior}\\ \textit{length of carrot + noodle behvior}\end{cases}$$

The success rates for the three experiments using classic motif discovery are denoted by MP_{carrot}, MP_{noodle} and $MP_{noodle+carrot}$ respectively. We performed the same experiment using *Semantic-Motif-Finder*, with fixed don't-care and prefix/suffix lengths, given by:

Maximum don't-care length = 5% (length of carrot+ noodle behavior)
prefix/suffix length = 95% (length of carrot+ noodle behavior)/2

We refer to the success rate of our algorithm as SMP. We ran each experiment 100 times, giving the same data to all algorithms. We also calculated the default rate, Random Sampling, which is the ratio of the embedded motifs length to the full time series length. TABLE III summarizes the results, showing that our algorithm outperforms all variations of the classic motif discovery for all conditions.

TABLE III. THE PERFORMANCE OF ALGORITHMS FOR CARROT-NOODLE DATASET.

Data Length	*Random Sampling	MP_{carrot}	MP_{noodle}	$MP_{noodle+carrot}$	SMP
8,192	20.7	66	80	88	**94**
12,288	13.8	50	50	62	**72**
16,384	10.4	30	49	56	**57**
24,576	6.9	23	22	32	**36**

* Calculated exactly, not computed experimentally.

To guard against happening to choose a dataset that favors our algorithm, we model an arbitrary semantic event in which an umpire signals a "*Leg Bye*" (LB) but then immediately changes his mind and signals "*Penalty Runs*" (PR) [6]. We embedded instances of these two behaviors in the random walk time series. For the motif discovery we performed the experiment using the subsequence length of LB/PR and the length of combination of two classes, MP_{LB} and MP_{LB+PR}. For our algorithm, SMP, we simply chose the prefix, suffix and the don't-care region to be the same length. Each region was one third of the length of the combination of two classes. The result is shown in TABLE IV, showing once again that our algorithm outperforms all variations of the classic motif discovery for all conditions.

TABLE IV. THE PERFORMANCE OF ALGORITHMS FOR CIRCKET DATASET.

Data Length	*RandomSampling	MP_{LB}	MP_{LB+PR}	SMP
8,192	4.8	25	50	**55**
12,288	3.2	16	20	**40**
16,384	2.4	14	14	**24**
24,576	1.6	5	13	**18**

* Calculated exactly, not computed experimentally.

It is important to note that we did not contrive these datasets to favor our algorithm. If we had added a small amount of random walk between the two atomic events (a more realistic model), our accuracy would remain essentially unchanged, whereas the classic approach would suffer greatly. Moreover, in [19] we show additional similar experiments in other domains, with equally impressive results.

Finally, we consider the efficiency of our algorithm. We created a time series like Fig. 17 with a length of 32,768 and measured the running time of our algorithm over different subsequence lengths for prefix/suffix. The length of the don't-care regions was constant for different runs. We have noted elsewhere that there are no other algorithms that are as expressive as our algorithm, so we just compare our method with the fastest classical motif discovery algorithms, STOMP [27]. We measured the running time of STOMP over different subsequence lengths. The result is shown in Fig. 20.

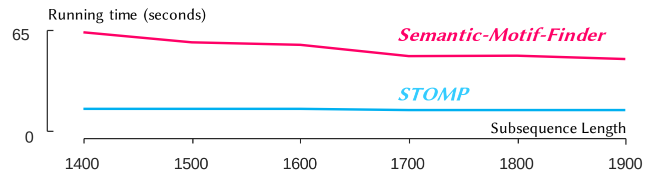

Fig. 20. The running time of Semantic-Motif-Finder and STOMP algorithm over different subsequence lengths. The length of don't-care region is constant. Semantic-Motif-Finder is not significantly slower than STOMP, in spite of being much more expressive in the regularities it can find.

As expected, the performance of our algorithm does not change by varying the prefix/suffix length. Note that we are only four times slower than STOMP, which is a less expressive algorithm, and the state-of-the-art for classic motif discovery [27]. Some of that timing difference is intrinsic, but we believe that we can close the gap to less than a factor of two, with further optimizations of our implementation.

In the second experiment we measure the running time of Semantic-Motif-Finder over different lengths of the don't-care region. We choose the length of the don't-care region to be in the range of 2.5%, 5%, …, 25% of the subsequence length, with a constant prefix/suffix length. Fig. 22 shows that Semantic-Motif-Finder algorithm's time complexity has minor changes as we vary the length of the don't-care regions.

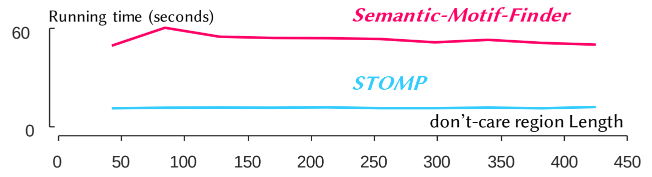

Fig. 21. The running time complexity of Semantic-Motif-Finder over different don't-care regions length. The prefix/suffix length is constant.

In the next experiment we show the effect of varying the prefix/suffix length. Recall the classic poem "The Raven" , used in the example in Section VI. We varied the prefix/suffix length from 100 to 500 in steps of size 10. We observe that for a very wide range of prefix/suffix lengths, we find the same semantic motif. Fig. 22 shows a binary vector for different choices of prefix/suffix length. The binary vector is the representation of finding vs. not finding the motifs. For the range of length 130 to 350 we find the same semantic motif for this example, suggesting the robustness of our definition to user-choices.

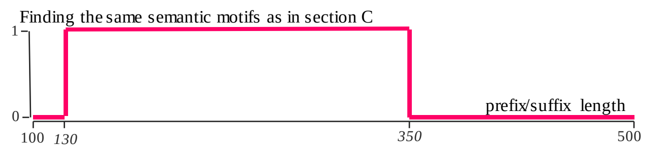

Fig. 22. The binary vector which shows finding the exact semantic motifs for the example in Section VI. The x axis is the prefix/suffix length and y axis equal to one indicates finding the same semantic motif as in Section VI whereas zero indicates finding different semantic motifs.

VII. DISSCUSSION AND CONCLUSIONS

We have introduced time series semantic motifs, a generalization of classic time series motifs. We have also shown that semantic motifs are much more expressive than classic motifs, and this allows us to search large complex datasets for repeated structure that would not be discoverable with any of existing techniques [2][3][5][8][24][13].

To make the expressiveness of our definition more clear, in Fig. 23 we show the relationship between current motif definitions/algorithms.

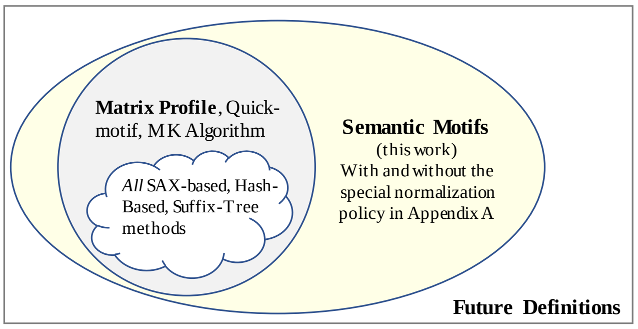
Matrix Profile, Quick-motif, MK Algorithm
All SAX-based, Hash-Based, Suffix-Tree methods
Semantic Motifs
(this work)
With and without the special normalization policy in Appendix A
Future Definitions

Fig. 23. The expressiveness of motif definitions shown as a nested hierarchy. See Appendix A to appreciate the claim that our work fully subsumes the Matrix Profile.

We have shown that our increased expressiveness does not come at a great cost, the time and space overhead to support it is inconsequential. In future work we plan to investigate the implications of our work for algorithms that use motifs as inputs, including time series rule discovery and segmentation. Furthermore, working with domains experts and attempting to further understand their needs, we will explore additional generalizations of motif discovery.

ACKNOWLEDGMENT

We acknowledge all the donors of data, and funding from NSF awards DGE 1631776, CNS 1544969, and from Google, Mitsubishi and NetApp.

REFERENCES

[1] Abdoli, A., Murillo, A.C., Yeh, C.C.M., Gerry, A.C. and Keogh, E.J., 2018, December. Time Series Classification to Improve Poultry Welfare. In *2018 17th IEEE International Conference on Machine Learning and Applications (ICMLA)* (pp. 635-642). IEEE.

[2] Balasubramanian, A., Wang, J. and Prabhakaran, B., 2016. Discovering multidimensional motifs in physiological signals for personalized healthcare. *IEEE journal of selected topics in signal processing*, *10*(5), pp.832-841.

[3] Berlin, E. and Van Laerhoven, K., 2012, September. Detecting leisure activities with dense motif discovery. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing* (pp. 250-259). ACM.

[4] Bhattacharjee, T., Song, H., Lee, G. and Srinivasa, S.S., 2018. Food manipulation: A cadence of haptic signals. *arXiv preprint arXiv:1804.08768*.

[5] Chiu, B., Keogh, E. and Lonardi, S., 2003, August. Probabilistic discovery of time series motifs. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 493-498). ACM.

[6] Dau, Hoang Anh et al (2018). The UCR Time Series Classification Archive. Retrieved from https://www.cs.ucr.edu/~eamonn/time_series_data_2018.

[7] Gao, Y. and Lin, J., 2017, November. Efficient discovery of time series motifs with large length range in million scale time series. In *2017 IEEE International Conference on Data Mining (ICDM)* (pp. 1213-1222). IEEE.

[8] Gao, Y. and Lin, J., 2018. Exploring variable-length time series motifs in one hundred million length scale. *Data Mining and Knowledge Discovery*, pp.1-29.

[9] Gharghabi, S., Imani, S., Bagnall, A., Darvishzadeh, A. and Keogh, E., 2018, November. An ultra-fast time series distance measure to allow data mining in more complex real-world deployments. In *IEEE Int. Conf. on Data Mining (ICDM2018)*.

[10] Gharghabi, S., Imani, S., Bagnall, A., Darvishzadeh, A. and Keogh, E., 2018, November. Matrix Profile XII: MPdist: A Novel Time Series Distance Measure to Allow Data Mining in More Challenging Scenarios. In *2018 IEEE International Conference on Data Mining (ICDM)* (pp. 965-970). IEEE.

[11] Hocking, D.P., Salverson, M., Fitzgerald, E.M. and Evans, A.R., 2014. Australian fur seals (Arctocephalus pusillus doriferus) use raptorial biting and suction feeding when targeting prey in different foraging scenarios. *PloS one*, *9* (11), p.e112521.

[12] Imani, S., Alaee, S., Keogh, E., 2019, May. Putting the Human in the Time Series Analytics Loop. In *Companion Proceedings of The 2019 World Wide Web Conference* (pp. 635-644). ACM.

[13] Imani, S., Madrid, F., Ding, W., Crouter, S. and Keogh, E., 2018, November. Matrix Profile XIII: Time Series Snippets: A New Primitive for Time Series Data Mining. In *2018 IEEE International Conference on Big Knowledge (ICBK)* (pp. 382-389). IEEE.

[14] Lachlan, R.F., Anderson, R.C., Peters, S., Searcy, W.A. and Nowicki, S., 2014. Typical versions of learned swamp sparrow song types are more effective signals than are less typical versions. *Proceedings of the Royal Society of London B: Biological Sciences*, *281*(1785), p.20140252.

[15] Ladds, M.A., Thompson, A.P., Slip, D.J., Hocking, D.P. and Harcourt, R.G., 2016. Seeing it all: evaluating supervised machine learning methods for the classification of diverse otariid behaviours. *PloS one*, *11*(12), p.e0166898.

[16] MASS: www.cs.unm.edu/~mueen/FastestSimilaritySearch.html

[17] Murray, D., Stankovic, L. and Stankovic, V., 2017. An electrical load measurements dataset of United Kingdom households from a two-year longitudinal study. *Scientific data*, *4*, p.160122.

[18] Parnandi, A., Le, K., Vaghela, P., Kolli, A., Dantu, K., Poduri, S. and Sukhatme, G.S., 2009, October. Coarse in-building localization with smartphones. In *International Conference on Mobile Computing, Applications, and Services* (pp. 343-354). Springer, Berlin, Heidelberg.

[19] Project Website: https://sites.google.com/site/semanticmotifs/

[20] Silva, D.F., Yeh, C.C.M., Batista, G.E.D.A.P.A. and Keogh, E., 2016. SIMPle: Assessing music similarity using subsequences joins. In *International Society for Music Information Retrieval Conference, XVII*. International Society for Music Information Retrieval-ISMIR.

[21] St. Patrick. "*Hymn of St. Patrick*." The Catholic Layman, vol. 1, no. 2, 1852, pp. 16–18. JSTOR, www.jstor.org/stable/30064868.

[22] The Raven read by James Earl Jones Video. (Oct 29, 2008). Retrieved Feb 1, 2019 from https://www.youtube.com/watch?v=sXU3RfB7308.

[23] Truong, C. & Anh, D., 2017. A novel clustering-based method for time series motif discovery under warping measure. *International Journal of Data Science and Analytics*, *4*(2), pp.113-126.

[24] Vahdatpour, A., Amini, N. and Sarrafzadeh, M., 2009, July. Toward Unsupervised Activity Discovery Using Multi-Dimensional Motif Detection in Time Series. In *IJCAI* (Vol. 9, pp. 1261-1266).

[25] Willett, D.S., George, J., Willett, N.S., Stelinski, L.L. and Lapointe, S.L., 2016. Machine learning for characterization of insect vector feeding. *PLoS computational biology*, *12*(11), p.e1005158.

[26] Wilson, H. Bird Song Complexity. Blog posting, retrieved Feb 1, 2019 from web.colby.edu/mainebirds/2012/08/07/bird-song-complexity/

[27] Yeh, C.C.M., Zhu, Y., Ulanova, L., Begum, N., Ding, Y., Dau, H.A., Zimmerman, Z., Silva, D.F., Mueen, A. and Keogh, E., 2018. Time series joins, motifs, discords and shapelets: a unifying view that exploits the matrix profile. *Data Mining and Knowledge Discovery*, *32*(1), pp.83-123.

[28] Zhu, Y., Zimmerman, Z., Senobari, N.S., Yeh, C.C.M., Funning, G., Mueen, A., Brisk, P. and Keogh, E., 2016, December. Matrix profile ii: Exploiting a novel algorithm and GPUs to break the one hundred million barrier for time series motifs and joins. In *2016 IEEE 16th ICDM* (pp. 739-748). IEEE.

Appendix A: Special Normalization Policy

We noted in Section V that we could force a special case of semantic motifs to exactly subsume classic motifs. Given a classic motif of length m, semantic motifs will return identical results, if we set $r = 0$, set $|\text{prefix}| + |\text{suffix}| = m$, and we make the following minor change to our code. Instead of z-normalizing the prefix and the suffix independently, we z-normalize them together. In MATLAB this is simply done like:

```
temp   = zscore([prefix suffix]);
prefix = temp(1 : length(prefix/2)) ;
suffix = temp(length(prefix/2) + 1 : end);
```

However, as we noted in Section V, in most cases there are good reasons *not* to use this option. None of the experiments in this work used this approach.

Appendix B: Discounting Simply Modifying the MP

We shared a preview of our code with other researchers to stress test our algorithms and ideas, and to garner feedback. While the feedback was positive, a significant fraction of them asked something to the effect of: "*Could you not just compute the classic Matrix Profile, and then do some post hoc analysis on it to find the sematic motifs?*". We were caught off guard by this question, as we assumed it was obvious that the answer is in the negative. If we *could* do this, it would be an attractive approach that we would have perused, as we could directly avail of the newest "off-the-shelf" Matrix Profile algorithms, including GPU versions etc. [28], with little effort. Below we will show why this idea is not possible.

As before, let us first consider the discrete analogue of text. Imagine we have a sematic motif of a person's name, with a misspelled version of it; `Ada P. Ray` and `Aja Roy`. Assume that these are embedded in a string…

`icdm19`**`AdaPRay`**`icml09`**`AjaRoy`**`ecml10cikm99`…

Here the sematic motif is obvious to the human eye. However, using the classic definition of motif, the best-motif of length three is {`cml`,`cml`}, and the best-motif of length six is {`icdm19`,`icml09`} etc. In fact, for *any* motif length, and for *any* value of top-K motif, the embedded name is the *last* item to be considered by classic motif discovery definitions. Generalizing these observation to real-valued data, it is easy to construct datasets for which there are sematic motifs that achieve a zero distance under our Definition 5, but rank arbitrarily low under classic motif definitions produced by the classic Matrix Profile [27][28].

While the above example is clearly contrived, we find that this exact issue also shows up in virtually all our real-valued datasets (the results in TABLE III and TABLE IV hint at that). For example, as shown in Fig. 24, if we revisit seal example shown in Fig. 12, we find that none of the best motifs, of any length in the range 14 to 36 seconds, overlap with the high-level `successful-hunt` behavior that sematic motifs could discover.

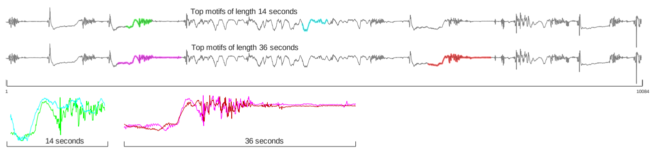

Fig. 24. (Contrast with Fig. 12) No matter what subsequence length is used with the classic Matrix Profile algorithm; it does not discover motifs that include the successful hunting behavior discovered with the sematic motif definition.