Matrix Profile XVI: Efficient and Effective Labeling of Massive Time Series Archives

Frank Madrid, Quentin Chesnais, Kerry Mauck, Shailendra Singh, Eamonn Keogh

Matrix Profile XVI: Efficient and Effective Labeling of Massive Time Series Archives

Frank Madrid[1] Quentin Chesnais[2] Kerry Mauck[2] Shailendra Singh[1] Eamonn Keogh[1]
[1]Computer Science, [2]Entomology @ University of California, Riverside
{fmadr002, singhs, eamonn}@ucr.edu, {chesnais.quentin,kerry.mauck}@ucr.edu

ABSTRACT
In domains as diverse as entomology and sports medicine, analysts are routinely required to label large amounts of time series data. In rare cases, this can be done automatically with a classification algorithm. However, in many domains, complex, noisy, and polymorphic data can defeat state-of-the-art classifiers, yet easily yield to human inspection and annotation. This is especially true if the human can access auxiliary information and previous annotations. This labeling task can be a significant bottleneck in scientific progress. For example, an entomology lab may produce several days' worth of time series, each day. In this work, we introduce an algorithm that greatly reduces the human effort required. Our interactive algorithm groups subsequences and invites the user to label a group's prototype, brushing the label to all members of the group. Thus, our task reduces to optimizing the grouping(s), to allow our system to ask the fewest questions of the user. As we shall show, in a deployed system for entomologists, we can reduce the human effort by at least an order of magnitude, with no decrease in accuracy.

Keywords
Time Series, Segmentation, Labeling, Classification

1. Introduction
A common problem for data analysts across many domains and disciplines is the annotation of long sections of time series data [17]. In some cases, this can be done automatically, using a time series classification algorithm [4][8]. However, in many circumstances, the performance of even the state-of-the-art algorithms can be significantly worse than a human expert. Consider for example the three snippets of insect behavior shown in Figure 1. To an expert in phytophagous (plant eating) insects with piercing-sucking mouthparts, these are obviously all examples of the same `probing-pathway phase` behavior. However, this data shows high levels of noise, and significant variability. This variability may be due to individual characteristics of the insects, or the vagaries of the sensing apparatus. In either case, no off-the-shelf time series classification system we are aware of would work well here. This is because *extracting* and reformatting such data is itself a much harder problem than the *classification*, and this preprocessing step cannot (in general) be done automatically [8] (See Appendix A).

Annotating such time series data necessitates *human* inspection; requiring the expensive services of a domain expert. Moreover, it can be significantly outpaced by the rate at which the time series data is generated. For example, a recent paper by an international team of animal movement ecologists bemoans the fact that annotating time series data is "*time consuming and error-prone for the domain expert and is now the limiting factor for realizing the value of* [*the time series*]" [17].

To prevent confusion, we refer to the task of annotating time series data as "labeling". While we could use the more familiar term "classification", that term is more typically used to describe the vast body of research that considers the special (and, as [8] and others argue, often *unrealistic*) problem, in which the time series has been contrived into a relational data format [8][13].

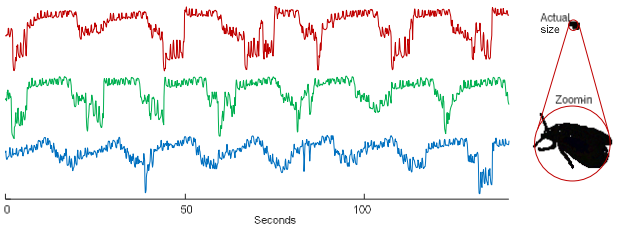

Figure 1: Examples of probing-pathway phase behavior for whiteflies (*Bemisia tabaci*). Note that this behavior has significant variability, some attributable to intrinsic variability of the insects, and some due to variability of the recording apparatus, which must be very sensitive, given the insects size.

Our problem setup, which we will formalize in Section 3, is as follows. Assume we are given a long time series, *T*. Some fraction (possibly *all*) of *T* is comprised of well-defined behaviors from a finite set of possibilities *S* which may or may not be known in advance. We assume a human domain expert can correctly identify any snippet of the time series representing any of the well-defined discrete behaviors in *S*, if given the opportunity to view it. Note that the human annotator may have the possibility to consider additional, auxiliary information. For example, the time series may be accompanied by video or audio recordings. For any time series snippets that are uncertain or ambiguous in the time-series-only view, the human labeler may expend the extra effort to review the parallel multimedia data.

Given this model, the human could label all the data. However, suppose that she labels one-second chunks at a time. For the whitefly problem introduced in Figure 1, entomologists often run eight parallel twelve-hour experiments each day, that would total 345,600 one-second snippets to classify daily. Even if our entomologist could label ten behaviors a second, during an eight-hour shift, she still could not keep up with the daily production rate.

Our task-at-hand seems to invite automation by use of a time series classifier [17]; however, there are many circumstances in which even a state-of-the-art time series

classifier will not produce human competitive results. For example:

- A physiologist may need to label activities from participants in a study. However, while the activities may be visually obvious to the physiologist, variability caused by even minor differences in sensor placement, can confound time series classifiers [2]. Moreover, the physiologist may have insights and intuitions she can use, that would not be easily available to an algorithm. For example, if a noisy time series snippet appears equally likely to be `whisking-food` or `swimming`, but the surrounding time series was clearly `chopping-food`, she could use this context to resolve the ambiguity.
- Entomologists routinely use an apparatus called an Electrical Penetration Graph (EPG) to produce recordings of insect/plant interactions by insects with piercing-sucking mouthparts [16]. Some of these insects, such as the whitefly shown in Figure 1 are smaller than this dot • [7]. Naturally recording such tiny insects requires a very sensitive apparatus. This sensitivity means the signal, while interpretable to a trained entomologist, has variability caused by vibration, air currents, etc., all of which conspire to make it difficult for automatic classification [1][16]. Moreover, entomologists have insights and intuitions into the behavioral patterns of piercing-sucking insects which may be difficult to encode into algorithms, the classic "*I know when I see it*" skills. For example, a `path-phase` must always precede a `phloem-phase` since the Asian citrus psyllid must first puncture the plant with its piercing-sucking mouthparts before beginning sap ingestion [16].

Our proposed solution to this problem is the *Like-Behaviors Labeling Routine* (LBLR). The LBLR system greatly reduces the need for human annotation, by showing the user "clustered" snippets from the time series and brushing the given label to all the elements of the cluster. By using the Minimum Description Length (MDL) [9][11], to carefully reasoning about which snippets, and in which order to show the user, we can typically reduce the burden of human time by one to two orders of magnitude, with little or no loss of accuracy.

2. Related Work and Notation

We begin with a brief review of related work before introducing the notation needed to understand our framework.

2.1 Related Work

While the literature on time series classification is vast, see [17] and the references therein, there is very little on time series *labeling* [6]. Our proposed algorithm is superficially like *active learning* [13]; however, there are enough differences that the large active learning literature is of little help. In particular:

- In active learning, the unlabeled data points are typically shown to the user (teacher/oracle) one object at a time[1]. In contrast, by exploiting the ability to "overplot" time series (see Figure 2.*right*) we propose to show the user *collections* of objects, exploiting the human ability to "batch process".
- In active learning, the goal is typically to build a more accurate classifier. In contrast, we are only interested in annotating data to allow downstream analytics. Thus, active learners *exploit* unlabeled data until the model accuracy plateaus. In contrast, we annotate unlabeled data, until it is all labeled, or the annotator prematurely terminates the labeling procedure.

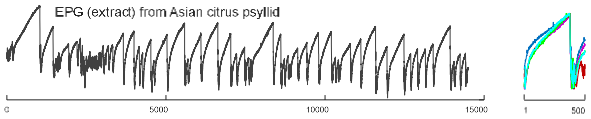

Figure 2: (*left*) A snippet of insect EPG data from an Asian citrus psyllid (*Diaphorina citri*) [18]. Rather than label each pattern one-by-one in this view, we propose to group related patterns (*right*) to label multiple patterns at once with a single interaction.

A recent paper introduces a new shapelet-based informativeness metric for time series *active learning* [13]. However, they assume that all the data has been perfectly segmented and arranged into a relational format. As [8] points out, partitioning a long time series stream into this format is *much harder* than the classification or labeling task that follows. Moreover, [13] assumes that all snippets belong to some well-defined class, and that all possible classes are known ahead of time. As we shall show below, these are unrealistic assumptions for real-world problems.

2.2 Notation

We begin by defining the data type of interest, the *time series*:

Definition 1: A *time series* $T \in \mathbb{R}^n$ is a sequence of real-valued numbers $t_i \in \mathbb{R} : T = [t_1, t_2, \ldots, t_n]$ where n is the length of T.

We are typically not interested in the global properties of a time series, but in the local regions known as *subsequences*:

Definition 2: A *subsequence* $T_{i,m} \in \mathbb{R}^m$ of a time series $T \in \mathbb{R}^n$ is a contiguous proper subset of the values from T of length m starting from position i. Formally, $T_{i,m} = [t_i, t_{i+1}, \ldots, t_{i+(m-1)}]$ where $m < n$.

We plan to increase labeling efficiency by grouping similar subsequences together, these are known as *time series motifs*:

Definition 3: A *time series motif* is the most "similar" subsequence pair of a time series. Let $T_{a,m}$ and $T_{b,m}$ for some $a, b \in [1,2,\ldots,n-m+1]$ and $a \neq b$ be two distinct subsequences of T. $T_{a,m}$ and $T_{b,m}$ is a motif pair iff $dist(T_{a,m}, T_{b,m}) \leq dist(T_{i,m}, T_{j,m})\ \forall i, j \in [1,2,\ldots,n-m+1]$ where $i \neq j$ and $dist$ is a function that computes the z-normalized Euclidean distance between a and b [20][21].

One of the most efficient ways to locate time series motifs is to compute the *matrix profile* [20] of T.

Definition 4: A *matrix profile* $P \in \mathbb{R}^{n-m+1}$ of a time series T is a meta time series that stores the z-normalized Euclidean distance between each subsequence and its nearest neighbor

[1] There is research on *batch mode active learning*. However, here the unlabeled examples are extracted in batches to reduce computation effort in retraining a classification model.

where n is the length of T and m is the given subsequence length. The top-1 motif can be found by simply locating the two lowest values in P. The remaining motifs can be extracted by finding the next lowest values in the matrix profile.

To avoid trivial matches in which a pattern is matched to itself, or a pattern that largely overlaps with itself, the matrix profile incorporates an "exclusion-zone" concept, which is a region before and after the location of a given query that should be ignored. The exclusion zone is heuristically set to $m/2$ [20].

Figure 3 illustrates a matrix profile on a small toy dataset. The time complexity to compute a matrix profile P is $O(n^2)$. This may seem untenable for time series data mining, but several factors mitigate this concern. First, note that the time complexity is independent of m, the length of the subsequences. Thus, unusually for a time series algorithm, the time and space complexity do not depend on the length of the *subsequences*.

Secondly, the matrix profile can be computed with an *anytime algorithm*, and, in most domains, in as few as $O(nc)$ steps the algorithm converges to what would be the final solution [20] (c is a small constant). Finally, the matrix profile can be computed with GPUs, cloud computing and other HPC environments that make scaling to at least tens of millions of data points trivial [20]. Even using standard hardware, all the examples in this paper can be computed *much* faster than real-time. For example, 30 minutes of EPG sampled at 60Hz takes about four minutes to compute using STOMP [20]. If that was not fast enough, STAMP can produce a very high-quality approximation in under five seconds [20].

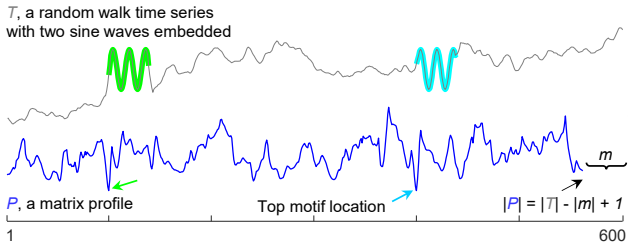

Figure 3: A synthetic time series T which has two (highlighted) motifs imbedded, and its matrix profile P. Note that P minimizes at the location of the motifs.

2.3 Minimum Description Length

As we shall show in the next section, a key subroutine in our system requires our algorithm to reason about which motifs are semantically the same, and which are distinct. To achieve this, we plan to exploit MDL to decide which group of subsequences are semantically similar and thus can be grouped together to receive a single label from a user [9][10][11]. We can gain some intuition about this idea by first considering the text analogue of time series.

2.3.1 MDL: Text

Suppose that "motif" discovery has managed to whittle down the character representation of a long text string to just four candidates, T_1, T_2, T_3, and T_4 :

T: `scatters shatters swatters syzygies`

Here we see the first three words **`scatters`**, **`shatters`** and **`swatters`** as being similar enough to warrant grouping into a single class. MDL helps us to realize the appropriate grouping by attempting to perform lossless compression on the data by exploiting regularities shared *among* the time series.

For example, since **`s-atters`** is a repeated structure among some of the words, we consider the first word **`scatters`** and think of it as being a potential model or *hypotheses* for the four words. We can use this model to try to "explain" the rest of the data, by encoding each word T_i with the hypothesis H:

H = `scatters`

$T|H$: `·····` `·h·····` `·w·····` `·yzygie·`

where · indicates a shared character between T_i and H. Using this encoding, we only need H and the differing (symbol, position) pair values to reconstruct T_i. Thus, if $DL(T_i)$ is the original description length of T_i, the reduced description length $DL(T_i, H)$ is $DL(H) + n(\lceil \log_2 |S| \rceil + \lceil \log_2 |H| \rceil)$ where n is the number of different (symbol, position) pairs, S is the number of unique symbols and $|H|$ the length of H. With 26 unique symbols (letters of the alphabet) and word lengths of 8, we achieve the description lengths depicted in Figure 4. From these values, we can see that $\mathrm{DL}(T_i, H) < \mathrm{DL}(H) + \mathrm{DL}(T_i)$ for T_2 = **`shatters`** and T_3 = **`swatters`** indicating {**`scatters`**, **`shatters`**, **`swatters`**} is a logical group.

	$\mathrm{DL}(T_i)$	$\mathrm{DL}(T_i\|H)$	$\mathrm{DL}(T_i, H)$	<	$\mathrm{DL}(H) + \mathrm{DL}(T_i)$
T_1	20	–	–	–	–
T_2	20	8	28	T	40
T_3	20	8	28	T	40
T_4	20	48	68	F	40

Figure 4: The description lengths for the original subsequences $\mathrm{DL}(T_i)$, the description length of modeling T_i given H $\mathrm{DL}(T_i|H)$, and the reduced description length $\mathrm{DL}(T_i, H)$. Since the reduced description length for T_4 is less than its reduced description length, T_4 should not be grouped with H. Conversely, T_2 and T_3 should be grouped with H.

2.3.2 MDL: Time Series

While MDL is well defined in the intrinsically discrete space (text, DNA etc.), it requires some modifications to work in the real-valued time series space[2] [10][11]. In particular, we quantize our time series T using the following discretization function.

Definition 6: The *Discretization* function is used to map a real-valued time series T into $(b - a) + 1$ discrete values in the range $[a, b]$ and is defined as:

$$\mathrm{Discretization}_{(a,b)}(T) = \mathrm{round}\left((b-a) \times \left(\frac{T - \min}{\max - \min}\right) + a\right)$$

where min and max are the minimum and maximum values of T respectively if T is not constant[3].

[2] We wish to disclaim that our model may better be described as *MDL-like* or *MDL-inspired*. Our goal is to build a practical system and not make any claims about MDL model selection.

[3] In this work we consider the range $[1,2^4] \rightarrow [1,16]$ yielding $Dicretization(T) = \mathrm{round}\left((16-1) \times \left(\frac{T - \min}{\max - \min}\right) + 1\right)$.

Since a time series is a sequence of real-valued numbers, the discretization of T results in a reduction of precision. However, it has been shown that this reduction in precision does not result in a significant reduction in classification accuracy [9], suggesting that little or no useful information is lost in the process. Note that as we are working with z-normalized times series, in practice all subsequences have very similar max and min values. Figure 5 shows the effects of discretization on four insect data snippets.

For any discretized time series T, we are interested in approximating how many bits it takes to represent it or its *description length*:

Definition 7: The *description length* DL of a time series T is the total number of bits required to represent it. When Huffman Coding is used to compress the time series, the description length of time series T is defined by:

$$\mathrm{DL}(T) = |\mathrm{HuffmanCoding}(T)|$$

One of the key steps in finding clusters of semantically similar subsequences is identifying a subsequence, or *hypothesis*, which exemplifies a common substructure. Using this hypothesis, we calculate the *reduced description length* of a time series:

Definition 9: A *reduced description length* of a time series T given hypothesis H is the number of bits used to encode T, exploiting information in H. The reduced description length of T using H is defined as:

$$\mathrm{DL}(T, H) = \mathrm{DL}(H) + \mathrm{DL}(T|H)$$

The first term $\mathrm{DL}(H)$ is the *model cost* or the number of bits required to store the hypothesis H while the second term $\mathrm{DL}(T|H)$, the *correction cost*, is the number of bits required to rebuild the entire time series T from H. Storing the difference vector $H - T$, we can easily regenerate T; thus, $DL(T|H) = \mathrm{DL}(H - T)$ [9].

The following example offers a visual intuition of these ideas. In Figure 5.*left* we show four time series from an insect EPG dataset [5] and their corresponding discretizations. In Figure 6 we use the first time series H as our hypothesis and model the remaining three time series $H - T_i$. Lastly, the results in Figure 7 indicate $\{H, T_1, T_2\}$ is a logical cluster. As we shall explain in Section 4. The effectiveness of our labeling algorithms hinges on the ability of the algorithm to decide which patterns should be grouped together and presented to the user as a *single* entity deserving of a *single* label.

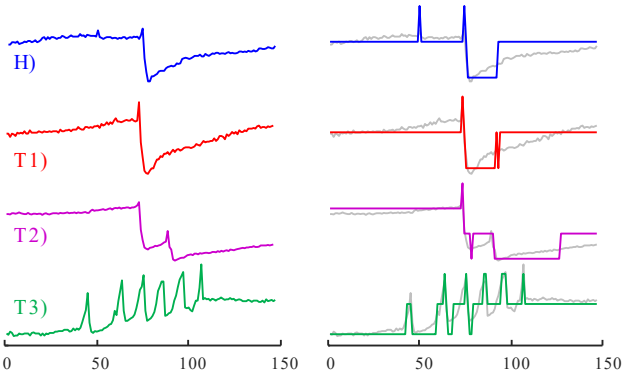

Figure 5: Given four z-normalized time series of insect dataset [5]. To avail MDL, we must discretize each time series (*left*) into their 4-bit representations (*right*).

The first three patterns, H, T_1, and T_2 in Figure 5.*left* are the same semantic behavior, but the fourth T_3 is a different behavior and should not be included in the grouping. One might imagine that one could learn a Euclidean distance "radius" that covers all members of the same class, at least within a given domain. However, this in impractical for several reasons; more complex shapes tend to need a much greater radius that simpler shapes (i.e. complexity bias [4]). Even if you could learn a good radius for a given length, the best radius for different lengths can scale non-linearly (indeed, *non-monotonical*ly for z-normalized time series).

MDL allows us to bypass these issues. As we show in Figure 6, we can treat one time series H, as the hypotheses and then encode T_i using H by calculating the difference vector $H - T_i$ (right). Comparing $\mathrm{DL}(H) + \mathrm{DL}(T_i)$ to $\mathrm{DL}(T_i, H)$, we can then judge if T_i should be grouped with H, that is to say, we ask if $\mathrm{DL}(T_i, H) < \mathrm{DL}(H) + \mathrm{DL}(T_i)$.

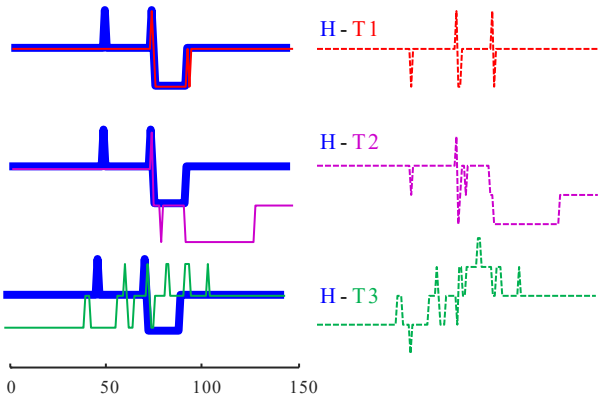

Figure 6: *left*) The hypotheses overlaid with three candidate time series. *right*) Subtracting the time series from the hypotheses produces a *difference vector*, the "simplicity" of which is suggestive of similarity of the two signals.

In our experiments, $DL(H) + DL(T_i)$ proved to be too conservative of an upperbound for $DL(T_i, H)$ bound while $DL(T_i)$ proved to be too liberal. Giving the annotator the functionality to easily add or remove subsequences gives us the ability to be slightly conservative or slightly liberal when proposing candidate groupings to the annotator.

	$DL(T_i)$	$DL(H-T_i)$	$DL(T_i)-DL(H-T_i)$
T_1	169	156	13
T_2	264	240	24
T_3	220	257	−37

Figure 7: The description lengths for the original subsequences $DL(T_i)$, the description length after modeling T_i with H $DL(H-T_i)$, and their difference, or the number of bits saved. Since T_1 and T_2 result in a positive number of bits saved, they are considered to be semantically the same as H.

As we can see in our examples from Figure 6, the discretizations of H and T_1 are so similar, their difference vector $H-T_1$ consists mostly of zeros, with just a handful of non-zero values, which will require less bits to encode than the original raw data. From Figure 7, modeling T_1 and T_2 with H resulted in a positive number of bits saved or a reduction in the number of required bits while modeling T_3 with H resulted in a negative bitsave; thus, we confidently feel that T_1 and T_2 belong in the semantic group as H while T_3 does not. Thus, in this example, the set of time series {H, T_1, T_2} would be presented to the user as a (tentative) group that can be assigned a label with a single interaction, rather than three interactions.

3. LBLR

We begin with a concrete statement of the problem we wish to address. While this addresses a common task [17], to the best of our knowledge, this problem has not been formalized.

> **Problem Definition**: *Time Series Labeling*. Given a time series T of length n, which is comprised (at least partly) of regions that correspond to discrete well-defined behaviors (which may or may not be known in advance) and given access to an oracle that can label subsequences of T, into these behaviors. We wish to produce A, an integer vector of length n, which correctly annotates T, while minimizing the number of accesses to the oracle.

This notation is illustrated in Figure 8.

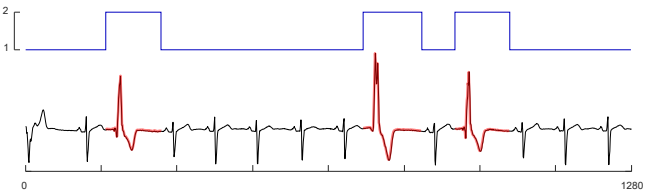

Figure 8: An example illustrating a time series T, and its annotation vector A. The key to A is domain dependent, for example, here 1 = *normal beat* and 2 = *ventricular contraction*.

Without loss of generality, we assume the oracle always produces the correct label. As noted above, the oracle may have access to out-of-band information (such as video or audio recorded in parallel to the time series), and may be able to label even unseen data, based on context. For example, if an oracle was asked to label a previously unknown activity, she might notice that it happens five times a day, and the file name is `Ali_Muhammad_02.txt`. This would surely allow our oracle to suspect that the behavior is connected to the *salat*, the five daily prayers of a Muslim.

To solve our problem, we envision a system that shows the human annotator (hereafter, just *annotator*) a set of time series snippets believed to be in the same class and asks for a label. This framework requires us to define the snippet grouping policy and the set of annotator operators. For clarity of presentation, we discuss these below in reverse order.

3.1 User Interactions

Any active-learning interaction system for labeling time series must at least allow the following annotator operators during each interaction:

- **Label as a Predefined Behaviors:** These behaviors are previously known to the annotator. For example, predefined behaviors for EPG include `active sap ingestion`, `intercellular penetration`, and `intracellular puncture` [16].
- **Label as a New Behavior**: These behaviors are previously unknown to the annotator. For example, a behavior study in "youth activity" only expected to see a handful of recognizable behaviors such as `walking`, `running`, etc. However, after inspecting the data an unexpected behavior, which was later realized to be `skipping` was found.
- **Label as an Unknown Behavior:** The annotator may label a snippet as `unknown`. These types of behaviors may simply be sufficiently unique, or the annotator does not simply have enough information to identify the behavior.
- **Label as a Polymorphic Behavior:** The identified system contains two or more distinct behaviors and should be reconsidered using a more conservative grouping. The active-learning system may be too aggressive and attempt to group behaviors that the annotator considers distinct. Thus, the annotator must be able to tell the system to reconsider the grouping, and instead present multiple, more conservative groupings of the data. Alternatively, the annotator may create a new behavior which is an amalgamation of the two or more distinct behaviors. For example, during our case study, LBLR identified the ending of a path-phase and the beginning of a phloem phase in which the creation of the behavior `path-phloem` would have accurately labeled the behavior.
- **Temporarily Pass:** The annotator chooses to revisit the current motif during some future iteration. The annotator may be unsure of a label for a snippet. Yet later in the process, once they had gained some experience with the data, they would be able to classify it. For example, they may be unsure if a snippet of gait represents a fast walk or a slow run. But later, when they have seen an unambiguous walk section, they would realize that the original ambiguous section was indeed run.
- **Terminate the Labeling Process:** The annotator may preemptively end the labeling process after a sufficient fraction of the time series has been labeled. The remaining snippets may be unknown/unclassifiable, sufficiently unique, or unambiguously consistent.

Having defined all the possible user interactions, we are finally able to introduce our algorithm.

3.2 The LBLR Algorithm

In Section 2.3, we considered text and time series toy examples and demonstrated how to we can determine whether two

subsequences are semantically similar. In this section, we will describe and then explain our algorithm in detail. Table 1 outlines a high-level overview of our algorithm for the rapid labeling of a time series.

Table 1: LBLR ALGORITHM

Input:	T : Time Series, l : Model length
Output:	L : Labels corresponding to T
1	D = DiscreteNormalization$_{16}$(T)
2	**while**(T has unlabeled data ∧ ¬(user(quits))
3	M = FindModel(T*,l)
4	S = MDL(D*,M)
5	BrushLabels(S)
6	Cleanup()
7	**end while**

We begin our algorithm by quantizing a real-value time series T into a discrete-value time series D to accommodate MDL (line 1). Next, we begin the labeling process and continue until no unlabeled data remains (lines 2-7). We also allow the user to prematurely end after any iteration if they feel enough of the data has been annotated or if the remaining data does not warrant annotation.

Beginning each iteration, our algorithm identifies our model M as a time series motif of T^*, the union of unlabeled data points of T and an inclusion range m heuristically set $l/2$ (line 3). Using M as our hypothesis, our algorithm produces S, the set of subsequences of the discretized data corresponding to T^*, sorted by their reduced description length (line 4). If left unsupervised, LBLR would automatically group together all subsequences $S^* \in S$ whose reduced description length is less than its original description length, that is,

$$\mathrm{DL}(S^*, M) < \mathrm{DL}(S^*)$$

The user may add or remove subsequences from an overplot of the elements in S (see Figure 9), if they feel that the MDL-driven grouping was too conservative or too liberal (Line 5).

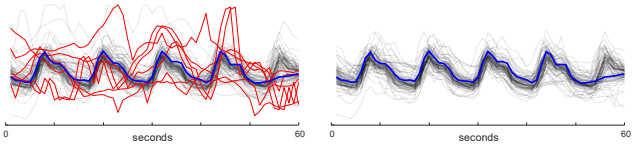

Figure 9: An example of LBLR clustering subsequences from an EKG where the hypothesis depicts sustained ventricular tachyarrhythmia. LBLR identified 96 similar subsequences of which 8.33% were false positives. The annotator may easily remove such sequences with minimal interaction.

Lastly, during the cleanup phase, we automatically label any unlabeled subsequences with length less than l. The labels will be split between the shared label of the subsequences neighboring behaviors if the neighboring behaviors have the same label or is shared between the two neighboring behaviors if they have different labels (see Figure 10) (line 6).

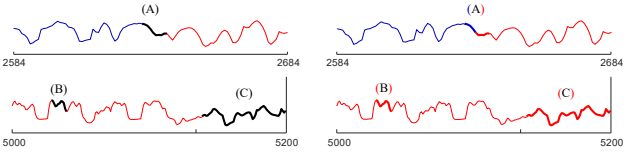

Figure 10: Two time series snippets of an EPG which contain unlabeled sequences with length less than l (left) and were automatically labeled (right). Subsequence (A) indicates a transition between two separate behaviors and is thus shared between the two while subsequences (B) and (C) share the only available label between their neighboring behaviors.

The basic algorithm can benefit from an optional module which we discuss in the next section.

3.3 Priming Run

Before LBLR interacts with the user, it can perform and optional *priming run* on the time series data. In this phase, the algorithm annotates any "low-hanging fruit" in the data; regions that it can confidently classify *without* human intervention. The set of such regions is domain dependent, but some examples include:

- **Constant Regions**: As Figure 11 hints at, many scientific and medical datasets have regions of constant values. They are typically the result of a disconnected or faulty sensor, or a sensor recording a value that greater/smaller that its precision allows. One caveat to note is that some disconnected sensors report what is visually a constant line, but it may have some tiny variance due to electrical noise. As we are working in z-normalized space, the z-normalization will magnify such data. Thus, we may have to set a domain specific threshold to define "constant" in our domain. For example, for our EPG example, $|\Delta V| < 1.0$ mV is only observed when the whitefly is dead, or the wire is broken or disconnected from the insect.

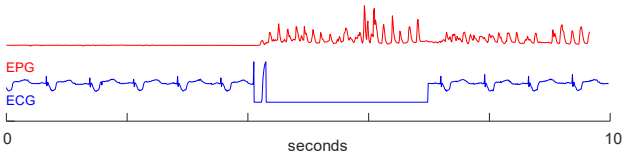

Figure 11: Two ten-second snippets of data, EPG from a whitefly and ECG data from an anonymous patient. Both contain regions of "constant" values due to sensor disconnect.

- **Trivially Classifiable Patterns**: Many domains produce some patterns that are classifiable by simple algorithms, although these may not be particularly interesting. Consider Figure 12 which shows a near perfect sinusoidal pattern imbedded in what is otherwise a typical EGC signal identified as interference from an insulin pump. This pattern is common, and well conserved, thus worth removing in a priming run, rather than wasting an annotator's time.

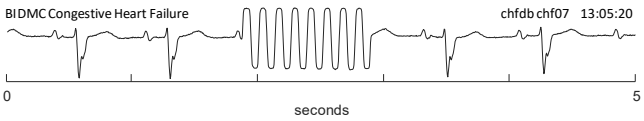

Figure 12: A snippet of ECG data that contains an artifact from electrical interference from an insulin pump.

Note that we do not attempt to exclude very noisy regions during our priming run, despite their ubiquity. As we have explained, our MDL scoring function would in any case prioritize these subsequences last.

4. Experimental Evaluation

To ensure that our experiments are reproducible, we have built a website which contains all data/code/raw spreadsheets for the results, in addition to videos of the system in action. LBLR has also been made available for open source distribution.

Note that while this tool was built specially for entomology, labeled datasets are hard to find in this domain. Moreover, all the labeled datasets we do have access to, where labeled by authors of this paper, or their workmates, thus presenting the possibility of bias. For that reason, in addition to testing on entomology data, we also test on proxy datasets, including cardiology and human-activity data.

4.1 Reproducible Experiments

In addition to the somewhat anecdotal domain specific case studies in later sections, we want to have experiments that can not only be reproduced but could be potentially *improved* on by the community. However, clearly, we cannot make our domain experts available in perpetuity. With mild assumptions, we can address this issue.

We have created a diverse collection of completely annotated time series we can use to mimic human feedback. When labeling clustered subsequences, a collection of potentially related time series, instead of showing it to a human labeler, we simply use the ground truth annotations to assign the group the majority label. This simple scheme does not use the full expressiveness of the system, as it does not allow the *Label as Unknown, Temporarily Pass* or *Label as Multiple Behaviors* interactions.

We created datasets in each of three domains: (1) Entomology, (2) Cardiology and (3) Human-activity which each exhibit two class behavior. Recall that the Cardiology and Human-activity were chosen as plausible proxies for entomological data.

While in principle a two-class dataset could be labeled with just two interactions, in practice, most datasets have a wandering baseline, noise, drift and polymorphic behaviors that make that unlikely. Nevertheless, we would hope the system could label most of the data accurately while minimizing annotator interactions. To summarize the system's performance, we will use *progress plots* as illustrated in Figure 13.

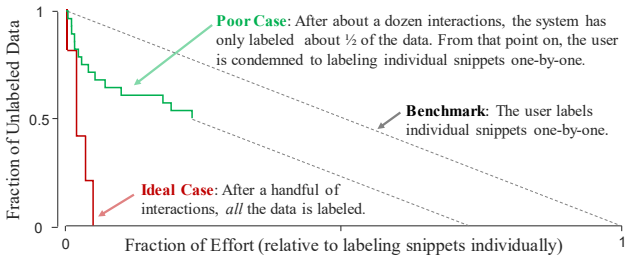

Figure 13: A key to interpreting progress plots for a labeling system. In future plots we will omit the dashed lines, which are implicit if a line does not terminate at 0 on the y-axis.

Note that these plots do not encode the *accuracy* of system, however this was always greater than 99%. The interested reader can visit [22] to see all the original data and a forensic step-by-step trace of each experiment.

Our only inputs into the system are the time series, and a suggestion of the subsequence length, after that, the proxy system (AutoLBLR) runs without human intervention, until all the data is labeled.

4.1.1 Entomology

For our first domain of interest, we selected five independent snippets of EPG data from an Asian citrus psyllid (*Diaphorina citri*) illustrated in Figure 14. Though each behavior (waveform) has characteristics to facilitate identification, the waveforms exhibit high variability attributed to the sensitivity of the recording apparatus and the intrinsic variability of the insect.

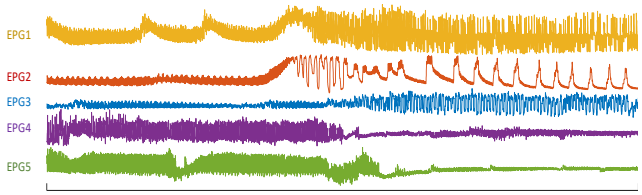

Figure 14: Five one-two minute time series snippets of insect EPG data from an Asian citrus psyllid *(Diaphorina citri)* each featuring two behaviors. EPG2 exhibits an easily identifiable behavior (to the human eye) but exhibits high variability which may impede LBLR.

Specifying each dataset and a fixed subsequence length of 100 to LBLR, AutoLBLR accurately labeled *all* datasets with less than 50% of the relative human effort (see Figure 15).

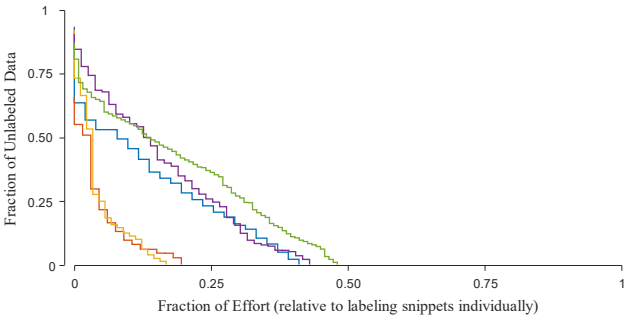

Figure 15: Progress plot for the corresponding datasets depicted in Figure 14. With less than half the relative effort, all datasets were completely labeled. While the EPG1 and EPG2 datasets achieved equal results with less than a quarter of the relative effort due.

AutoLBLR was particularly effective at labeling EPG1 and EPG2 requiring less than 25% of the human effort. These two plots contain a "long fall" which indicate a large grouping of subsequences suggesting the variability within the waveforms of these two datasets did not impede AutoLBLR's ability to quickly label these datasets. However, despite a few "minor successes", AutoLBLR was essentially labeling a single subsequence at a time in EPG5 though we were still able to completely label the entire dataset with half the human effort.

In Section 5, we discuss a few ideas which take advantage of the unique characteristics of each waveform to more quickly label time series from this domain.

After performing preliminary waveform analysis on the EPG datasets, the suggested subsequence length was not obvious since each different waveform can consist of a different length and is also dependent on the variability of the insect. We repeated each experiment, supplying a different subsequence length to LBLR. The effects of these changes on AutoLBLR's progress plots can be seen in Figure 16. AutoLBLR is indeed reflective of a change to the subsequence length but is not particularly sensitive.

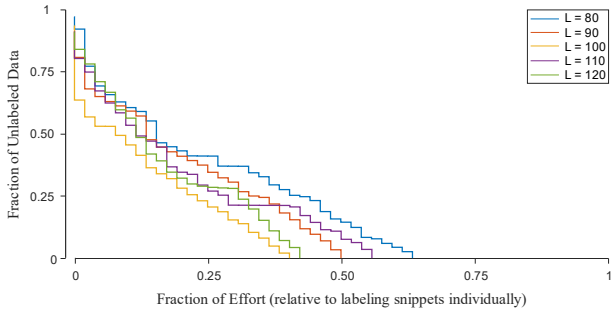

Figure 16: Sensitivity of AutoLBLR to the change of subsequence length on the EPG1 dataset from Figure 14. Though the subsequence length varied by up to 20%, the fraction of relative effort remained in a 25% range.

4.1.2 Medicine (as a proxy for entomology)

We now turn our attention to our second domain where we selected five snippets of medical data illustrated in Figure 17. The data includes both ECG and APB (Arterial Blood Pressure) traces. All data is from humans, except M1 which is from a pig used as a proxy for a stabbing victim. Though each behavior is well-defined, the difference between each class is very subtle, increasing the risk of false positives.

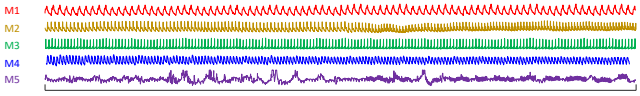

Figure 17: Five snippets of medical data featuring two district behaviors, as labeled by medical experts. The subject in M1 has undergone a fatal stabbing, M2 survey the effects of a tilt table on a subject, M3 and M4 show signs of Pulsus Paradoxus while M5 depicts an individual undergoing cardiac death.

Though LBLR has the propensity of being liberal in its proposed groupings, relying on an annotator to prune out false positives, AutoLBLR accurately labeled each dataset using about a quarter of the relative human effort and was *particularly* effective in labeling M2 requiring only three iterations, one iteration shy of the optimal possiblity (see Figure 18). Furthermore, despite having no behaviors detectable to the human eye, AutoLBLR correctly identified about half of M5 in just two iterations.

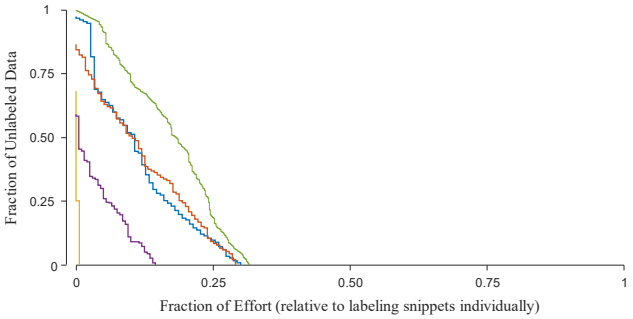

Figure 18: Progress plot for the corresponding datasets depicted in Figure 17. With about a quarter of the relative effort all datasets were completely labeled. While M2 was completely labeled in less than 1% of the relative effort.

4.2 Case Study: Entomology

At the time of going to press we have just begun the first human trials with LBLR (see Figure 19) inviting experts in entomology to us it to annotate EPG data.

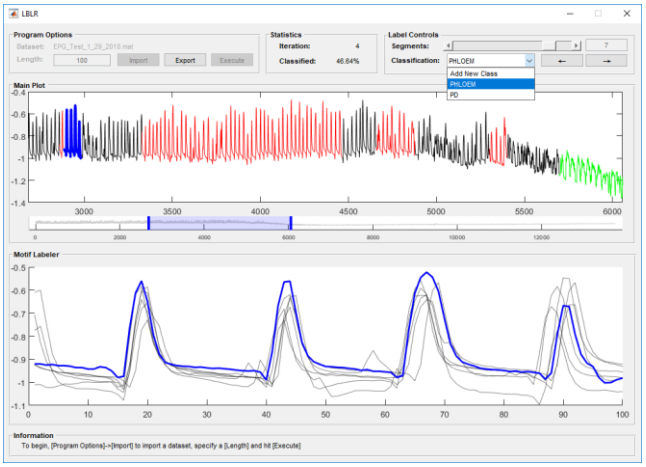

Figure 19: A snapshot of our deployed implementation of LBLR. After performing four iterations, the user is in the process of brushing the phloem label onto the subsequences identified in the bottom cluster. These changes will be reflected in the top graph and the user will again be shown another candidate cluster of subsequences.

Two such entomologists who are experts in whitefly (*B. tabaci*) waveform analysis used the application to label an EPG time series of an Asian citrus psyllid (*D. citri*). Their results, along with the results of the AutoLBLR are shown in Figure 20.*top-panel*.

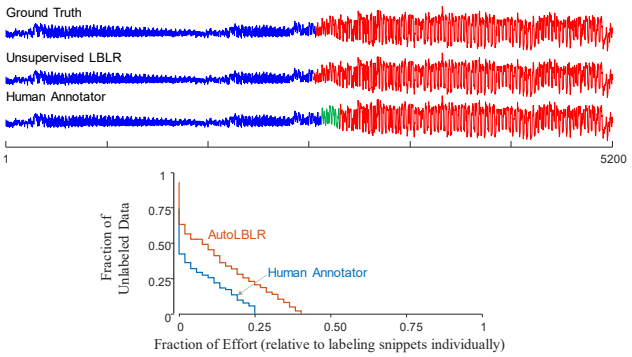

Figure 20: *top-panel*) LBLR results when used by an annotator (bottom) and AutoLBLR (middle) compared to the ground truth (top). AutoLBLR mislabeled the transition between the two behaviors achieving an accuracy of 99.80% while the human annotator designated the region as unclassifiable, achieving a 100% accuracy on the 95.83% of the data he had classified. *bottom-panel*) Comparison of the relative effort required to label the data. This result suggests that that AutoLBLR proxy for humans was pessimistic, the experiments in the previous sections should be considered lower bounds.

In Figure 20.*bottom-panel* we compare the relative efforts required for the annotator and AutoLBLR to completely label the dataset. These results are = good news for us. They suggest that our experiment in Sections 4.1.1 to 4.1.3 might have been somewhat pessimistic. Recall that the set of interactions available in these human-proxy experiments is a subset of the interactions available to actual humans.

5. Conclusions and Future Work

In this work we have concretely formulated a task which appears to be ubiquitous in science and medicine, yet poorly supported, or only supported by limited and very domain specific tools [17][18]. In contrast, our LBLR system is domain independent and generic. It requires only one input, the subsequence length. However that is one parameter that domain experts typical have some intuition for.

While our motivation comes from entomology, we have demonstrated the utility of our system on very diverse domains. In the best cases, for example the M2 dataset, the system can reduce the human effort one-hundred-fold. In other cases the reduction of human effort is not as great, however this is a real and essentially free improvement. That is to say, there is no additional cognitive overhead for using. Even if the system "only" eliminates three-quarters of the work this allows the entomologists to process four times as much data.

We see two avenues for improvement. The first is to optimize the user experience. To glean the necessary feedback to do this, we are currently conducting a user study in multiple entomological labs. The second avenue for improvement is in further improving the critical snippet grouping strategy (Section 2.3.2). It is possible that replacing the Euclidian distance with Dynamic Time Warping will help here [4].

We have made all code and data freely available to allow the community to confirm and extend our ideas.

Acknowledgments

We gratefully acknowledge NSF 1631776.

6. References

[1] F. Adasme-Carreno C. Munoz-Gutierrez, J. Salinas-Cornejo, and C. Ramirez. A2EPG: a new software for the analysis of electrical penetration graphs to study plant probing behaviour of hemipteran insects. Comput. Electron. Agric. 113: 128–135, 2015.

[2] L. Atallah, B. Lo, R.C. King, G.Z. Yang: Sensor Positioning for Activity Recognition Using Wearable Accelerometers. IEEE Trans. Biomed. Circuits and Systems 5(4): 320-329, 2011.

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

[4] G.E. Batista, X. Wang and E.J. Keogh, 2011, April. A complexity-invariant distance measure for time series. In Proc' of the 2011 SDM pp. 699-710.

[5] J.P. Bonani, A. Fereres, E. Garzo, M. Miranda, B. Appezzato-Da-Gloria, J. R. S. Lopes. Characterization of electrical penetration graphs of the Asian citrus psyllid, in sweet orange seedlings. Entomologia Experimentalis et Applicata 134: 35–49, 2010.

[6] Y. Chen, Y. Hao, T. Rakthanmanon, J. Zakaria, B. Hu, E.J. Keogh: A general framework for never-ending learning from time series streams. Data Min. Knowl. Discov. 29(6): 1622-64 (2015)

[7] G.S. Hodges, G.A. Evans. An identification guide to the whiteflies (Hemiptera: Aleyrodidae) of the Southeastern United States. Florida Entomologist 88(4):518–534, 2005.

[8] B. Hu, Y. Chen, E.J. Keogh. 2016. Classification of streaming time series under more realistic assumptions. *Data Mining and Knowledge Discovery*, 30(2): 403-437, 2016.

[9] B. Hu, T. Rakthanmanon, Y. Hao, S. Evans, S. Lonardi, and E.J. Keogh. "Discovering the intrinsic cardinality and dimensionality of time series using MDL." *Data Mining (ICDM), 2011 IEEE 11th International Conference on*. IEEE, 2011.

[10] K. Kawabata, Y. Matsubara, and Y. Sakurai. 2018. StreamScope: Automatic Pattern Discovery over Data Streams. In Proceedings of the First International Workshop on Exploiting Artificial Intelligence Techniques for Data Management (aiDM'18).

[11] Y. Matsubara, Y. Sakurai, C. Faloutsos: AutoPlait: A utomatic mining of co-evolving time sequences. SIGMOD Conference 2014: 193-204

[12] G.B. Moody, R.G. Mark, A.L. Goldberger. PhysioNet: a research resource for studies of complex phyiologic and biomedical signals, Comput Cardiol, 2000, vol. 27 (pg. 179-82)

[13] F. Peng, Q. Luo, L.M. Ni. 2017 ACTS: An Active Learning Method for Time Series Classification. In *2017 IEEE 33rd ICDE,*, 175-178, 2017.

[14] M.A. Rahman, W. Ma, D. Tran, and J. Campbell. A comprehensive survey of the feature extraction methods in the EEG research. Springer-Verlag, Berlin, 2012

[15] A. Reiss and D. Stricker. Introducing a New Benchmarked Dataset for Activity Monitoring. The 16th IEEE International Symposium on Wearable Computers (ISWC), 2012.

[16] W.F. Tjallingii. Comparison of AC and DC systems for electronic monitoring of stylet penetration activities by homopterans." Journal of Evolutionary Biology (2001): 41.

[17] Y. Walker, M. Jones, R. Laramee, O. Bidder, H. Williams, R. Scott, E.L.C. Shepard, and R. Wilson. TimeClassifier: a visual analytic system for the classification of multi-dimensional time series data. The Visual Computer 31(6-8), 1067-1078, 2015.

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

[19] J.R. Villar, M. Menendex, J. Sedana, E. Cal, V.M. Gonzalez. Analyzing accelerometer data for epilepsy episode recognition. 10th International Conference on Soft Computing Models in Industrial and Environmental Applications. Springer, 2015.

[20] C.M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, E. J. Keogh. Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets. *IEEE 16th International Conference on Data Mining (ICDM)*, 2016, pp. 1317-1322.

[21] Y. Zhu, Z. Zimmerman, N.S. Senobari, C.M. Yeh, G. Funning, A. Mueen, P. Brisk, E. J. Keogh. Matrix Profile II: Exploiting a Novel Algorithm and GPUs to Break the One Hundred Million Barrier for Time Series Motifs and Joins. *16th International Conference on Data Mining (ICDM)*, 2016, pp. 739-748.

[22] Project URL: www.cs.ucr.edu/~fmadr002/LBLR.html

Appendix A: Why not Automatic Classification?

In the introduction we pointed to Figure 1 and claimed, "*no time series classification system we are aware of would work here*". Given the extraordinary number of papers on time series classification (see [3][4] and the references therein) this claim may seem surprising, thus we take the time to briefly justify it here. It *is* possible that if we carefully extracted patterns from this dataset, and very carefully aligned them to begin and end at the same logical point (reinterpolating longer or shorter patterns as needed), and applied the appropriate smoothing/spike removal, that we could *then* apply one of the more than 100 algorithms that have been applied to the UCR archive and get reasonable results. However, this glosses over the fact that extracting and reformatting such data, is itself a *much harder* problem that cannot (in general) be solved automatically (this point is made in more detail in [8]). With a little introspection, this claim is obviously true. Consider that ECGs are possibly the most studied type of time series in human history, and they are highly constrained by physics and physiology. Yet, even here, ECG beat extraction is still considered a very difficult problem [14]. If the reader doubts the inadequacy of ECG extraction/classification algorithms, consider Figure 12 again. The state-of-the-art classifier used by Physionet misclassified the transition from insulin pump inference to a normal heartbeat as a *Premature Ventricular Contraction*, a trivial mistake no human would make [12].