PMP: Privacy-Aware Matrix Profile against Sensitive Pattern Inference for Time Series

Li Zhang† Jiahao Ding‡ Yifeng Gao‡ Jessica Lin†

Abstract
Recent rapid development of sensor technology has allowed massive time series data to be collected and set the foundation for the development of data-driven services and applications. During the process, data sharing is often required to allow modelers to perform specific time series data mining tasks based on the need of data owner. The high resolution of time series data brings new challenges in privacy protection, as meaningful information in high-resolution data shifts from concrete point values to shape-based patterns. Numerous research efforts have found that long shape-based patterns could contain more sensitive information and may potentially be extracted and misused by a malicious modeler. However, the privacy issue for time series patterns is surprisingly seldom explored in privacy-preserving literature. In this work, we consider a new privacy preserving problem: preventing malicious inference on long shape-based patterns while preserving short segment information to maintain utility task performance. To mitigate the challenge, we investigate an alternative approach by sharing Matrix Profile (MP), a versatile data structure that supports many time series data mining tasks. We found that while MP can prevent the concrete shape leakage, the canonical correlation in MP index can still reveal the location of sensitive long pattern information. Based on this observation, we design two attacks named Location Attack and Entropy Attack to extract the pattern location from MP. To further protect MP from these two attacks, we propose a Privacy-Aware Matrix Profile (PMP) via perturbing the local correlation and breaking the canonical correlation in MP index vector. We evaluate our proposed PMP against baseline noise-adding methods through quantitative analysis and real-world case study to show the effectiveness of the proposed method. Our source code is available at https://github.com/lzhang18/PMP.

1 Introduction
The wide use of sensors in personal devices and other infrastructures have allowed the collection of high-resolution time series and boosted the demand for data-oriented services and applications such as medical monitoring [27], industrial system prognostics [30] and smart homes analytics [26]. For example, a farm owner (referred as ‘data owner’ or ‘owner’) might collect massive data from their own smart sensor system to monitor the behavior of farm animals such as cows and chicken in real-time [1]. Since the owner does not have the expertise to analyze the data, they might seek for some data mining service (referred as ‘modeler’)

†George Mason University, {lzhang18, jessica}@gmu.edu
‡University of Houston, jding7@uh.edu
©University of Texas Rio Grande Valley, yifeng.gao@utrgv.edu

Copyright © 2023 by SIAM
Unauthorized reproduction of this article is prohibited
of short segments and privacy (for prevention of malicious inference on long shape-based patterns), we investigate an alternative approach by sharing Matrix Profile (MP) instead. MP is recently proposed as an efficient and versatile data structure that records, for each subsequence of a given length, the distance and the index of its closest match in the time series. There are two advantages to using MP. First, MP is ideal for a modeller to use as a versatile intermediate feature to support most fundamental data mining tasks such as motif discovery, anomaly detection, and more complex tasks such as rule discovery, segmentation, and time series chain discovery. Second, we found that it is difficult for the malicious modeler to recover the original time series and the patterns themselves by solely using MP due to the use of Z-normalized Euclidean distance (as will be discussed in Section 5). However, we found that the canonical correlation in MP index can still reveal the location of long patterns. Based on the observation, we designed two attacks named Location Attack and Entropy Attack to extract sensitive pattern location from MP. To further protect MP from these two attacks, we propose Privacy-Aware Matrix Profile (PMP) via perturbing the local correlation and breaking the canonical correlation in MP index vector. Our overall framework is shown in Figure 1. Instead of sharing the raw time series or perturbed time series, the modeler would only have access to the PMP. The shared MP could enable one to perform the utility task(s) based on PMP, and prevent inference the shapes and the locations of long patterns.

In summary, our contributions are listed as follows:

- We consider a new privacy preserving problem: preventing malicious inference on long shape-based patterns while preserving short segment information to maintain the performance of downstream tasks. To the best of our knowledge, this is the first work to investigate this problem, and it cannot be solved by existing point-based approaches.
- We investigate an alternative approach by sharing Matrix Profile (MP), a versatile data structure supporting many time series data mining tasks. We found that while MP can protect sensitive long pattern shape, it could also leak pattern locations due to correlation in MP Index (MPI).
- We design two attacks based on location and entropy of MPI to extract sensitive pattern location from MP.
- We further propose a defense algorithm called PatternHide to generate a Privacy-aware Matrix Profile (PMP), which can prevent the location leakage of sensitive patterns while keeping the downstream task performance.
- We evaluate PMP against baseline methods through quantitative analysis and multiple real-world case studies to show the effectiveness of the proposed method.

2 RELATED WORK

In the last two decades, a great amount of research efforts have been put into time series data mining tasks such as time series classification, anomaly detection, motif discovery, and segmentation. Different from point-based approaches, pattern-based time series approaches are based on defining similarity on the subsequence level instead of point values to capture the notion of shape information, which aligns more closely to natural human understanding and intuition. Pattern-based time series methods handle large-scale time series well with excellent performance and interpretation. Recently, Matrix Profile (MP) is proposed as an efficient and effective subsequence-level data representation that supports most major fundamental tasks in a broad range of downstream applications. Existing work such as typically use the raw data to compute MP as the initial feature generation step, and then design algorithms or models based on the computed MP. None of the work considers the use of MP in the context of data sharing, nor the privacy issues associated with sensitive patterns.
Most existing approaches for privacy-preserving release are designed for low-resolution time series based on differential privacy (DP) to protect the events where each event is associated with a single point. For example, Dwork et al. [7] proposed a binary tree based DP algorithm for single events in finite streams. Fan et al. [10] presented FAST for realizing DP on user-based finite streams with a framework of sampling-and-filtering. However, these approaches are designed for low-resolution time series and cannot be used to protect sensitive long patterns while maintaining the downstream task performance. In addition, our problem is also different from PatternLDP [24], which is designed to protect point values from malicious attacks while preserving the utility of local patterns, whereas our problem is in the opposite direction to protect sensitive long patterns while keeping the utility of short patterns.

In summary, these time series DP solutions cannot be directly applied to our setting, since these approaches mainly focus on computing the aggregated estimates (e.g., prefix-sum and moving average) of the values under DP, whereas our goal is to release subsequence-level representation that prevents leaking the information of long patterns while maintaining the performance of different downstream tasks.

Another line of research focuses on sharing encrypted time-series data [4] [5] [6]. However, these methods can only support aggregation statistics computation and cannot support time series data mining tasks such as motif discovery and anomaly detection. Moreover, they mostly rely on oblivious RAM, secure multiparty computation, and function secret sharing [4] [5] [6], which require a significant number of cryptographic operations and secure communication channels.

To the best of our knowledge, there has been no existing work that offers a solution to protect from long pattern inference while maintaining downstream task performance.

3 Background and Preliminaries

In this section, we first review necessary time series related notations.

**Time Series** $T = [t_1, t_2, \ldots, t_n]$ is a set of observation ordered by time, where $t_i$ is a finite real number and $n$ is the length of time series $T$.

**Subsequence** $S_{i,l}^T = [t_i, t_{i+1}, \ldots, t_{i+l-1}]$ of time series $T$ is a contiguous set of points starting from position $i$ with length $l$. Typically $l \ll n$, and $1 \leq i \leq n-l+1$.

Previous work such as [27] [10] require subsequence comparison be non-trivial match, which prevents the comparison of subsequences that overlap more than 50% of the length.

**Non-trivial Match** Given a time series $T$, a subsequence $S_{i,l}$ with length $l$ is considered a non-trivial match of another subsequence $S_{j,l}$ of length $l$ if $|i-j| > l/2$.

In many applications, we are interested in finding similar “shapes.” Z-normalized Euclidean distance is used to achieve scale and offset invariance [17].

**Z-normalized Euclidean Distance (Z-norm ED):**

$$d(S_{p,l}, S_{q,l}) = \sqrt{\sum_{m=1}^{l} \left( \frac{(\mu_{p,m} - \mu_{q,m})^2}{\sigma_{p,m}^2} + \frac{(\mu_{q,m} - \mu_{p,m})^2}{\sigma_{q,m}^2} \right)}$$

where $\mu_{p,l}$, $\sigma_{p,l}$ and $\mu_{q,l}$, $\sigma_{q,l}$ are the means and standard deviations of subsequences $S_{p,l}$ and $S_{q,l}$, respectively.

Z-normalization step is very critical, as noted in previous work — “without normalization time series similarity has essentially no meaning. More concretely, very small changes in offset rapidly dwarf any information about the shape of the two time series in question” [15]. There are some additional benefits of preventing data leaking brought by utilizing Z-normalization distance, and we will discuss them in detail in Sec. 5.

**Distance Profile** Given a query subsequence $Q_l$ of length $l$ and a time series $T$, a distance profile $D_T(Q)$ is a vector containing the Z-normalized Euclidean distances between $Q$ and each subsequence of the same length in time series $T$. Formally, $D(Q, T) = [d(Q_1, T_1), d(Q_2, T_2), \ldots, d(Q_l, T_{n-l+1})]$.

**Matrix Profile Distance (MPD):** Matrix Profile Distance of time series $T$ given subsequence length $l$ is a vector of the Z-normalized Euclidean distances between every subsequence $S_{i,l}$ and its nearest neighbor (most similar) subsequence in time series $T$. Formally, $MP = [\min(D(S_{1,l}, T)), \min(D(S_{2,l}, T)), \ldots, \min(D(S_{n-l+1,l}, T))]$.

**Matrix Profile Index (MPI):** Matrix Profile Index of time series $T$ is a vector of indices containing the index of the non-trivial match of nearest neighbor subsequence of subsequence $S_{i,l}$ in time series $T$. Formally, $MPI = [\arg \min(d(S_{1,l}, T)), \ldots, \arg \min(d(S_{n-l+1,l}, T))]$.

Matrix Profile (MP) consists of two vectors, MPD and MPI. MP contains rich information about the data and the pattern location. For example, time series motif [22] can be found by exacting minimum value of MPD and the corresponding MPI.

4 Problem Statement

In practice, companies may utilize sensitive data for data mining in order to provide better services. As shown in Fig. 1(a), some data owner (e.g., factories with smart sensors, e-commerce platforms and hospitals) may capture clients’ time series and send to cloud servers or a modeler for performing some data mining tasks. The modeler will run Matrix Profile and then
design a model for improving the quality of data owners’ service or production-related decision making. However, directly transmitting raw time series or perturbed time series to modelers would allow malicious modelers to use the long patterns to infer sensitive information as we explained earlier in Introduction.

Since it is risky to share the data directly, a better protocol (Fig. 1(b)) is to first generate the MPD and MPI with a given length for the utility tasks, and then send it to the external cloud servers/modelers for utility tasks. For convenience, this given length is referred as the utility length and denoted as \( L_{util} \). There are two key research problems we should answer to comprehensively evaluate the privacy preserving performance of Matrix Profile:

- Does sharing MPD and MPI of length \( L_{util} \) protect the shapes of long patterns?
- Does sharing MPD and MPI of length \( L_{util} \) protect the locations of long patterns?

### 5 Advantages of Matrix Profile for Privacy Protection

To answer the above questions, we conduct a comprehensive study of Matrix Profile in terms of the privacy protection for long patterns.

#### 1) Difficult to recover raw data from MPD and MPI

We found that it is very difficult to recover the concrete values of time series solely from the shared Matrix Profile because of Z-normalization. Recall that Z-normalization requires the knowledge of the means and the standard deviations for both subsequences (see Z-normalized Euclidean Distance in Section 3). To reconstruct the entire time series from MP, we would need to know the means and standard deviations for every pair of subsequences. Thus, if we would like to recover the data from the distance, we have to solve for the values of data from pairwise distance between subsequences with mean \( \mu_{i,l} \) and standard deviation \( \sigma_{i,l} \) as parameters in every equation. As \( \sigma_i \) is a non-linear function of variable \( t_i \) to \( t_{i+l-1} \), it makes the inverse problem ill-defined. One could get infinitely many possible time series data as possible solutions satisfying a given MP, and hence cannot recover the original data.

#### 2) Difficulty to infer the locations of long patterns from MPD

As pointed out by previous work [11], “the distance between a pair of short subsequences does not necessarily share similar behavior with the distance between long subsequences if the length difference is large”. As a result, MPD naturally suppresses the location correlation between the short pattern and the long pattern. Thus, it is difficult to recover long pattern locations from a short-length MPD.

### 6 Threat Model and Attack Methods

While MP provides these two advantages in protecting privacy, since both MPD and MPI have to be shared to fulfill task utility, we found that attackers could still potentially infer the locations of sensitive patterns from MPI. We first define the threat model as follows:

**Adversary Goal.** Given MPD and MPI, the adversary aims to locate a pair of sensitive motifs of a sensitive length and we assume it is much longer than utility length \( L_{util} \). For convenience, we refer to this length as attack length and denote it as \( L_{attack} \). Specifically, the adversary goal is to detect frequent patterns in time series, which is similar to motif discovery task [22, 21, 18, 20] with one exception – the attacker could only use the shared Matrix Profile of length \( L_{util} \) to detect long motifs. For the rest of the paper, we simply refer to frequent pattern in time series as motif.

Following the widely used evaluation criteria [13, 22, 21, 12], we use the success rate to measure how accurate the detected pair of patterns match the actual locations (as will be illustrated in Section 8.2).

**Adversary Knowledge.** We assume a strong adversary (e.g., honest-but-curious modeler), who has no access to the sensitive time series data, but has white-box access to the Matrix Profile Distance MPD and Matrix Profile Index MPI with short subsequence length \( L_{util} \) pre-computed from the raw data.

#### 6.1 Attack Methods

We use a real-world time series to demonstrate how MPI can leak long pattern information. Figure 2 top shows a snippet of a dishwasher power consumption time series. The time series contains two long dishwasher cycles. Visually, we can see that
We propose two scores based on MPI for this location (e.g., the peak in Fig. 3) (Line 7-8). In this highest frequent bin is assigned as the second instance’s time series data. However, our proposed attack methods length

We refer to the first strategy as Location-based Attack and the second strategy as Entropy-based Attack.

Note that existing motif discovery algorithms \[22, 21, 18, 20\] could not be used to detect the long motif of length \(L_{\text{attack}}\) through the shared Matrix Profile of length \(L_{\text{util}}\) because they require access to the original time series data. However, our proposed attack methods can still find long motif from a single shared MPI without the raw time series.

7 Defense Strategy

In this section, we first introduce a new concept named Consecutive Index Block (CIB), and a new matrix profile, Masked Matrix Profile (Masked MP), which will be used in the proposed algorithm. Finally, we introduce our proposed PatternHide algorithm.

7.1 Consecutive Index Block and Masked Matrix Profile

We introduce Consecutive Index Block (CIB) to capture the gradually changing regions in MPI that are likely to be indicative of long patterns. Given a Matrix Profile index MPI, a Consecutive Index Block (CIB) \(C_k\) consists of a set of consecutive index starting from index \(k\) where for any index \(i \in C_k\), we have \(|\text{MPI}(i) - \text{MPI}(i + 1)| < L_{\text{attack}}\).

Intuitively, the overall defense strategy of the proposed algorithm works by perturbing and hiding any outstanding CIB regions in MPI because long CIB potentially aligns with long pattern location and leads to pattern leakage. We next introduce Masked MP, the alternative “fake” MP used to hide real MP, while maintaining compatible functionality.

**Masked Matrix Profile** Given a mask vector \(M\), Masked Matrix Profile is computed through the same process as Matrix Profile but excludes any masked locations in \(M\) when computing MPI and MPD.

7.2 Proposed Algorithm

The proposed algorithm, PatternHide, is described in Algorithm \[2\]. Given a matrix profile MP, the algorithm consists of three steps. First, the algorithm obtains all the CIBs \(C\) in \(T\) and forms a set of sensitive segments \(S\) that may leak the pattern information. A segment is sensitive if it is close (index difference is less than \(L_{\text{attack}}\)) to a long CIB \(C_i\) with a length greater than \(L_{\text{perm}}\) (Line 4-8). Then the algorithm will perform the permutation step to generate “fake” 1-NN via masked matrix profile to break down long CIBs into small pieces that are similar to the non-sensitive area for every sensitive segment (Line 10-24). Finally, after checking all the sensitive segments, the algorithm will further examine existing cycles in MPI and modify any conflicting distance values to ensure consistency with the MPI.

1) Breakdown Long CIBs in Sensitive Segments

The goal of this component is to modify the indices in any sensitive segments that can potentially leak location information on long patterns, while keeping most of the functionality of the original MP. To achieve this goal, we replace the MPD and MPI with Masked MP.

---

**Algorithm 1 Proposed Attack Algorithm**

1: **Input**: MPD, MPI, \(L_{\text{attack}}\)
2: **Output**: long pair motif indices \(idx1, idx2\)
3: /* Calculate sum of distances in each sliding window of length \(L_{\text{attack}}^*\)*/
4: \(\text{SumDist} = \text{Score}(\text{MPD}, L_{\text{attack}})\)
5: \(idx1 = \arg \min(\text{SumDist})\)
6: /* Identify second motif index based 1st motif index*/
7: \(idx\text{Pool} = \text{MPI}[idx1 : idx1 + L_{\text{attack}} - 1]\)
8: \(idx2 = \arg \max(\text{GetIntervalFrequency}(idx\text{Pool}))\)
9: return \(idx1, idx2\)
Experimental Evaluation

(Line 10-23) which is computed with the sensitive area masked to ensure that no CIB of significantly large length exists. In the algorithm, if the consecutive index length is greater than a randomly generated threshold $L_{\text{rnd}}$ at time stamp $i$, we update the corresponding MPI and MPD with that of the Masked MP. Specifically, given a sensitive segment $S$, the algorithm maintains a mask vector $M$ to control the permuted MP (Line 10-11). Every time the condition in Line 14 is met, any subsequence overlapped with MPI($i$) is added into the mask vector (Line 16) and triggers the permutation process (Line 17-20). In the process, the algorithm computes masked MP with $M$ (Line 17) and replaces the remaining MPD and MPI in the sensitive segments with newly computed mask MP (Line 19-20). Then the algorithm samples another length threshold $L_{\text{rnd}}$ to prepare for the next pattern hiding operation (Line 13). The mask $M$ is set to zero vector after examining each sensitive segment. Since we assume that the sensitive segments are just a small portion in time series, the total indices that potentially need to be replaced are far smaller than the length of time series $n$. Replacing each index would take a time complexity of $O(n)$. Thus, the time complexity of this component is far less than computing the original MP, which is $O(n^2)$.

2) Resolve Conflicts in MPD Every MPD value is a distance and MPI may form a ‘cycle’ (i.e., given two subsequences $S_i$ and $S_j$, MPI($i$) = $j$ and MPI($j$) = $i$). In this case, we need to enforce distance symmetry constraint MPD($j$) = MPD($i$), otherwise a smart attacker might use this loophole to infer the modified locations. Therefore, we further generate fake symmetric distances (Line 25). Finally, the algorithm identifies any ‘cycles’ in the MPI and checks if MPD($i$) = MPD($j$) constraint is violated. If the distances are not equal, we adjust the MPD($i$), MPD($j$) values to be $\text{min}$(MPD($i$), MPD($j$)).

7.3 Advantage of Proposed Algorithm Our defense algorithm has three advantages attributed to the masked MP replacement. First, our algorithm plays a specific defense strategy protecting the location leakage through CIB. Second, our strategy only perturbs a small number of indices in the MPI vector that could potentially leak the pattern information, so the information loss compared to the original MPI is small. Third, our proposed PMP is generated by randomly swapping values in the original MP, which are still meaningful similarity information, thus making the defense location difficult to detect.

8 Experimental Evaluation

In this section, we demonstrate that the proposed defense methods can successfully defend the proposed two attacks while maintaining the utility task performance on both real-world and synthetic data. Unless otherwise specified, the parameter $L_{\text{perm}}$ is set to $L_{\text{perm}} = L_{\text{util}}/4$.

8.1 Detecting Planted Motif while Preventing Pattern Leakage We first evaluate the proposed defense method in motif discovery. Specifically, the utility task in the experiment is detecting motifs of length $L_{\text{utility}}$ while keeping the motif of length $L_{\text{attack}}$ protected. Following the previous planted motif evaluation experiment setting [13] [12] [21], we test our Privacy-Aware MP in two different scenarios:

1) Independent Scenario: In this scenario, sensitive patterns are independent of non-sensitive patterns. We randomly planted two independent motifs of lengths $L_{\text{util}}$ and $L_{\text{attack}}$, each one with two instances, into a random walk time series.

2) Correlation Scenario: In this scenario, the location of sensitive pattern is correlated with non-sensitive pattern. We randomly planted a motif of length $L_{\text{attack}}$ of two instances, and a motif of length $L_{\text{util}}$ of three instances into the random walk time series. Different from the first experiment, two out of three instances of the short motif overlap with the long motif. Fol-

Algorithm 2 Defense Algorithm: PatternHide

1: Input: $T$, MPD, MPI, $L_{\text{perm}}$, $L_{\text{attack}}$
2: Output: PMP = {MPD, MPI}  
3: $C$ = GetCIB(MPI) 
4: for Each $C_i \in C$ and $C_i > L_{\text{perm}}$ do 
5: /* Obtain Sensitive Intervals*/ 
6: $C_{\text{neighbor}} = \{|C| |C.\text{start} - C_i.\text{start}| < L_{\text{attack}}\}$ 
7: $S = \text{concat}(C_i \cup C_{\text{neighbor}})$ 
8: $S$ = add($S$) 
9: end for 
10: for $S \in S$ do 
11: $M = \{\}$ 
12: for $idx$ in $[S.\text{Start}, S.\text{End}]$ do 
13: $L_{\text{rnd}} \sim U(0, L_{\text{perm}})$ 
14: if ConsecutiveIdxCount(idx) > $L_{\text{rnd}}$ then 
15: /* Compute Masked Matrix Profile*/ 
16: $M.$add(OverlappingIntervals(MPI(idx))) 
17: MPD', MPI' = MaskMatrixProfile($T$, $M$) 
18: /* Replace Original Matrix Profile*/ 
19: MPD[\text{idx} : S.\text{End}] = MPD'[\text{idx} : S.\text{End}] 
20: MPI[\text{idx} : S.\text{End}] = MPI'[\text{idx} : S.\text{End}] 
21: end if 
22: end for 
23: end for 
24: /* Resolve any Symmetric Conflicts in Cycle Link Indexes by In-Place Update*/ 
25: MPD, MPI = FakeCycleLink(MPD, MPI) 
26: return MPD, MPI
lowing the setting of the large-scale planted motif experiment, the shape of motifs are randomly generated by using: \( p = \sum_{i=1}^{3} A_i \sin(\alpha i x + \beta) \), with random parameters \( A_i \in [0,10] \), \( \alpha_i \in [-2, 2] \) and \( \beta_i \in [-\pi, \pi] \). For each instance of a motif, \( \pm 5\% \) random noise is added. In both experiments, the length of the random walk time series is 10,000. We test four different \( L_{\text{attack}} = \{200, 300, 400, 500\} \) while keeping the utility length \( L_{\text{util}} = 100 \). All the experiments were repeated 50 times with different time series and motif shapes.

8.2 Evaluation Criteria There are two evaluation criteria: average utility task performance and average attack success rate. Both criteria are evaluated based on motif detection rate. The overlapping rate is measured by Jaccard similarity index: \( J(pred, gt) = \frac{\text{pred} \cap \text{gt}}{\text{pred} \cup \text{gt}} \). If the overlapping rate of the detected location and ground truth is greater than 0.25 in both instances, we say the motif is detected. The average attack success rate is computed based on the success of locating motif of length \( L_{\text{attack}} \) in the 50 experiments for each length. The utility task performance is measured by the average motif discovery success rate of length \( L_{\text{util}} \).

8.3 Baselines To the best of our knowledge, this is the first work that investigates the approach to prevent information leakage from deducing long patterns. None of the existing approaches are designed for this problem. Therefore, we compare with two simple baselines: (1) directly sharing MP and (2) adding noise in raw data. For the second approach, we test three different variations of noise values and examine attack success rate and downstream utility.

8.4 Attack Methods For MP sharing strategies, we evaluate the defense performance based on the success rate of two attack methods introduced in Sec. 6. For the approach that directly shares the matrix profile, we evaluate the performance by the success rate of directly detecting motif of \( L_{\text{attack}} \) given the shared time series.

8.5 Vs. Sharing Original Matrix Profile We first compare the proposed PMP with the original MP. We apply both attacks described in Section 6 and report the attack success rate. Fig. 4(a)-(b) show the attack success rates in the non-overlap setting. According to the figure, PMP can significantly reduce the attack success rate. When the sensitive pattern length increases, the attack success rate on sharing MP decreases. This is because the correlation between MP of \( L_{\text{util}} \) and \( L_{\text{attack}} \) is much smaller when the length increases, making it harder for attackers to retrieve information. However, our attack methods still maintain up to 70% success rate. Compared with sharing the original matrix profile, PMP maintains a stable low attack success rate (less than 0.15). Fig. 4(c)-(d) show the attack success rates in the overlap setting. Similar observation is found when we test the overlapping case. Sharing original matrix profile will lead to at least 0.7 success attack rate while the proposed approach can significantly reduce the chance of success in attack (up to 0.21). Moreover, the utility of PMP is shown in Table 1. From the table, it is about 0.88, which indicates our defense method can successfully defend from the attacks while keeping the utility of the shorter segments.

8.6 Vs. Perturbed Raw Series We next test our proposed method with perturbed raw data. Specifically, we compare with sharing raw data after adding Gaussian noise with variance \( \sigma^2 = \{0.1, 0.3, 0.5\} \), respectively. The attack success rates for all three cases are shown in Fig. 5(a)-(b) for both experiments. The utility task performance is shown in Table 1.

Table 1: Shared PMP vs. Perturb Raw Time Series

<table>
<thead>
<tr>
<th>Setting Method</th>
<th>small ( \sigma )</th>
<th>median ( \sigma )</th>
<th>large ( \sigma )</th>
<th>PMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Independent (utility)</td>
<td>0.84</td>
<td>0.32</td>
<td>0.115</td>
<td>0.875</td>
</tr>
<tr>
<td>Correlation (utility)</td>
<td>0.7</td>
<td>0.295</td>
<td>0.08</td>
<td>0.882</td>
</tr>
</tbody>
</table>

In both experiments (Independent Scenario and Correlation Scenario as explained in Section 8.1), adding only small amounts of noise will result in high attack success rate and high utility of the data. When more noise is added, both attack success rate and the utility decrease. However, the utility decreases faster than the attack success rate. This is because small utility length pattern is much easier to be affected compared

Figure 4: Compared with Sharing MP under Different \( L_{\text{attack}} \) (a-b) Independent Scenario, (c-d) Correlation Scenario.
with long pattern. Moreover, compared with noise-
adding approach, sharing PMP could maintain a very
high level of utility (0.875 success rate on motif detec-
tion) while keeping the attack success rate close to the
best defense performance of sharing perturbed time se-
ries. The experiment shows that the proposed approach
outperforms the perturbed raw time series protocol.

![Figure 5: Compared with Sharing Perturbed Time Series](image)

(a) Independent Scenario  (b) Correlation Scenario.

8.7 Motif Discovery on Electrooculogram Time Series

In sleep quality study, Electrooculogram is a
popular data type to study the sleep behavior of a subject [19]. In this case study, we test our proposed
method on preventing the leakage of long eye blinking
activities from the Electrooculogram time Series. We
utilized the EOG time series used by Madrid et al. [19].
The time series is shown in Fig. 6(a). According to
the authors, the time series is captured from a 66-year
old healthy male recorded during a sleep study. Two
types of motifs which correspond to two different types
of eye blinking activities are highlighted in the time
series. Without any perturbation, it is very easy for
the attacker to retrieve the type II eye blinking pat-
ttern through the location/entropy based attack method.
The detected pattern is shown in Fig. 9(b). After the
perturbation, the attack algorithm fails to detect the
long pattern. In fact, it locates the pattern somewhere
overlapped with the small motif. Therefore, the pro-
posed perturbed Matrix Profile protects the data from
the attacker.

9 Conclusion

In this paper, we consider a new type of privacy pre-
serving problem in time series, i.e., preventing mal-
icious inference on long shape-based patterns while pre-
serving short segment information for the downstream
task. To deal with the above problem, we introduced a
shared Matrix Profile (MP) approach by utilizing the
characteristics of MP as a stand-alone and versatile in-
termediate features. However, we illustrated that MP
index sharing can still reveal the location of sensitive
long pattern information based on two proposed attack
methods. We further proposed a Privacy-Aware Ma-
trix Profile (PMP) based on perturbing the index of
matrix profile at sensitive pattern regions. We evalu-
ated our proposed PMP sharing solution with several
classic defense methods through quantitative analysis
and demonstrated the effectiveness of protecting sensi-
tive information in the real-world case study. This work
will be a good first step that will hopefully inspire new
interesting research for privacy-aware shape-based time
series data mining methods.

References

[1] A. Abdoli, S. Alasoo, S. Imani, A. Murillo, A. Gerry,
L. Hickle, and E. Keogh. Fitbit for chickens? time se-
ries data mining can increase the productivity of poul-
try farms. In Proceedings of the 26th ACM SIGKDD
International Conference on Knowledge Discovery &
technology for personal healthcare. ACM Computing
A. Bostrom, P. Southam, and E. Keogh. The uea
multivariate time series classification archive, 2018.
S. Ratnasamy. {TimeCrypt}: Encrypted data stream
processing at scale with cryptographic access control.
In 17th USENIX Symposium on Networked Systems
Design and Implementation (NSDI 20), pages 835–850,
2020.
[5] L. Burkhalter, N. Küchler, A. Vianid, H. Shafagh, and
A. Hithnawi. Zeph: Cryptographic enforcement of
end-to-end data privacy. In 15th {USENIX} Symposium
on Operating Systems Design and Implementation
Waldo: A private time-series database from function
secret sharing. In 2022 IEEE Symposium on Security
Differential privacy under continual observation. In


