Journal of Intelligent Computing and Networking
https://www.ffspub.com/index.php/jicn/index
ISSN 3079-9228 (print)
ISSN 3107-2798 (online)
E-mail: jicn.office@ffspub.com

Article

SecMTSC: Achieving Privacy-Preserving and Efficient Multivariate Time Series Classification for Healthcare

Jieyi Wen[1,‡], Minhua Su[1,‡], Mei Cai[2,†], Jia-Nan Liu[1,†]
[1]*School of Computer Science and Technology, Dongguan University of Technology, Dongguan 523808, China*
[2]*Jinan University Library, Jinan University, Guangzhou 510632, China*
[†] *E-mail: tmclib@jnu.edu.cn (M.C.); j.n.liu@foxmail.com (J.-N.L.)*

Received: July 14, 2026/ Revised: August 7, 2026/ Accepted: August 11, 2026 / Published online: August 14, 2026

Abstract: Time series data are widely used in many application scenarios. For example, in healthcare scenario, time series-based health predictions require users to submit their time series data, so that service provider (e.g., medical center) can predict the user's health status through calculations (e.g., classification). To relieve the pressure on local computing, service provider may outsource its data to the clouds. However, since clouds are not fully trusted, how to outsource data to the cloud without sacrificing data security but allowing service provider to remotely predict user's health status is challenging. To address this, we present Secure Multivariate Time Series Classification (SecMTSC), a privacy-preserving and efficient multivariate time series classification scheme, which can be used for healthcare and other applications. Specifically, by leveraging two-party secure computation protocols, the proposed scheme enables service provider to respond the classification query launched by query users on their private sequence without disclosing the data privacy. To further enhance query efficiency, we design a Secure Filtering Protocol, which can filter and extract valid data from raw data before formal calculation, thereby selecting relatively qualified candidate sequences. Security proofs have shown that our scheme is secure under the semi-honest model. Furthermore, experimental results have demonstrated the promising practicality of our scheme.

Keywords: Multivariate Time Series Classification (MTSC); Privacy-Preserving; Secure Multi-Party Computation; Healthcare
https://doi.org/10.64509/jicn.23.134

1 Introduction

Time Series Classification (TSC) is a valuable technology for detecting abnormalities, monitoring trends, and making predictions [1–4]. It has become widespread in healthcare and other application scenarios, such as employing electrocardiogram (ECG) data [5] to identify ventricular contractions (polyvinyl chloride) in preterm infants, and leveraging photoplethysmography (PPG) data [6] to detect ventricular tamponade. In such applications, time series data can be collected through wearable or implantable Internet of Things (IoT) devices on the patient, and the data can be further analyzed and predicted [7–12].

With the increase of time series data, new challenges arise. Particularly for medical institutions, the vast volumes of data significantly slow down service towards patients. While outsourcing data to cloud servers can be a viable solution, concerns about data privacy and security should not be ignored. For instance, if highly sensitive health information were leaked to insurance agencies, the user will face financial losses and safety threats. Say, the user will encounter exorbitant healthcare costs, discriminatory practices, and potential denial of coverage, and is vulnerable to identity theft, fraudulent claims, and even blackmail. One countermeasure is encrypting the data before uploading it. However, performing computations on encrypted data poses considerable challenges for cloud servers. Moreover, encryption may impose unbearable computational burdens on sensors in the IoT system.

Considering the above issues, Saha *et al.* [10] adopted Hidden Markov Model (HMM) to propose a privacy-preserving time-series activity classification algorithm. It focuses more on the classification of univariate time series while achieving strong privacy and high efficiency. Liu *et al.* [11] proposed a Dynamic Time Warping (DTW) based

[†] Corresponding author: Mei Cai; Jia-Nan Liu
[‡] These authors contributed equally to this work.
[*]Academic Editor: Xiuli Bi

privacy-preserving analysis method for medical time series data that can be applied to medical research such as public health disease screening. This scheme focuses on similarity queries and does not consider how to monitor the patient's health status. In addition, Liu *et al.*'s scheme can only support operations on univariate time series and cannot be directly migrated to multi-dimensional data. Mercier *et al.* [13] introduced a modular privacy-preserving time series classification framework that utilizes deep learning and demonstrates impressive levels of accuracy and security. But its computational complexity and communication costs are prohibitive. Recent MTSC methods, Tang *et al.* [14] proposed Hformer to learn hierarchical multi-scale temporal representations. Nuyts *et al.* [15] introduced TSelect to remove irrelevant and redundant channels. Oh *et al.* [16] proposed TSSI, which transforms multivariate time series into multi-channel images for CNN-based classification. However, these methods focus on classification performance rather than the cryptographic protection of outsourced data and private queries.Therefore, efficient privacy-preserving MTSC remains an important research problem.

Overall, existing privacy-preserving time-series analysis schemes still face challenges in balancing security and efficiency. Existing methods mainly focus on univariate time-series analysis, similarity queries, or general privacy-preserving machine learning, and cannot directly support privacy-preserving multivariate time-series classification (MTSC) in healthcare scenarios. In addition, existing DTW-based schemes mainly focus on similarity computation rather than secure nearest-neighbor classification. Therefore, this work does not simply combine DTWD with secure multi-party computation, but designs application-specific secure protocols tailored to the DTWD-based MTSC process, achieving an effective balance between privacy protection and efficiency.

In this work, we propose SecMTSC, a secure, privacy-preserving, and efficient scheme for multivariate time series classification. Specifically, we leverage two-party secure computation and design a dual cloud servers model for MTSC. In this model, the service provider can outsource its multivariate time series data to the two non-colluding servers and allows user to query its data later. To enhance the efficiency, we design a secure filtering protocol (SFP) to filter candidate sequences in advance when a query is launched by a user. We highlight our contributions in the following.

- We propose a privacy preserving scheme for multivariate time series classification based on DTW. Compared with existing works, our scheme extends the solution on multivariate time series classification. In our scheme, the multivariate time series sequences can be classified without disclosing any information, including the classification results, to either or both parties involved.
- To improve efficiency, we design a secure filtering protocol (SFP) to filter candidate sequences in advance. We believe this protocol may be of independent interest for other secure multi-party computation applications involving multivariate time series. We also implement our proposed scheme and conduct a comprehensive experiment to show the efficiency of the scheme.

The remainder of this paper is organized as follows. We first give the related works in Section 2 and describe problem formulation in Section 3. Then we describe necessary preliminaries in Section 4. In Section 5, we propose SecMTSC, the privacy-preserving multivariate time series classification scheme, followed by security analysis and performance evaluation in Section 6 and Section 7, respectively. Finally, we draw the conclusion in Section 8.

2 Related Works

Time series data is a type of data in which the values are observed, recorded, or measured over time. Essentially, time series data is a sequence that is closely related to time, and in such a sequence, each data point is associated with a specific timestamp or time interval. Time series data is commonly used in various fields, including medical care, finance, economics, signal processing, weather forecasting, etc. The key characteristic of time series data is that it is ordered in time, which makes general data classification algorithms unable to be directly applied to time series data.

Compared with univariate time series data, multivariate time series data has multiple variables measured over time, making it more difficult to analyze. A series of researchers studied multivariate time series data and proposed multivariate time series data classification (MTSC) algorithms [17–19], including distance measures, shapelets, histograms over a dictionary, interval summarising, deep learning/neural networks and so on. Proven by Bagnall *et al.* [20] and Ruiz *et al.* [21], the DTW classification method has the best results on classifying multivariate time series data. Josu Ircio *et al.* [22] leveraged DTW to propose a filter method to select a subset of time series. Veizaga *et al.* [23] adopted it to propose a methodology for the classification of voltage sag sources. Shokoohi-Yekta *et al.* [24] studied DTW and proposed two obvious strategies for DTW to deal with multivariate time series problems: independent DTW (DTW_I) and dependent DTW (DTW_D) approaches. The main difference between these two strategies is that DTW_D requires the time series data to have the same dimensions, while DTW_I can be applied to calculate time series data that has different dimensions.

Multivariate time series data classification (MTSC) algorithms can be applied in many research areas [25–30]. Chambon *et al.* [25] addressed the diagnosis of sleep disorders by classifying sleep stages from multidimensional time-series signals such as electroencephalograms (EEGs), electrooculograms (EOGs), electrocardiograms, electromyograms, and electromyograms (EMGs). Liu *et al.* [26] proposed a tensor scheme along with a novel deep learning architecture called a multivariate convolutional neural network (MVCNN) for multivariate time series classification, in which the proposed architecture considers multivariate and lag-feature characteristics. Wilson *et al.* [27] introduced the simultaneous tracking and activity recognition (STAR) problem, which exploits the synergy between location and activity to provide the necessary information for automatic health monitoring. Recent studies have further improved MTSC through advanced feature learning. Du *et al.* [31] introduced ST-Tree for multi-scale feature extraction and interpretable classification. Liu

et al. [32] proposed a multiple-attention model to capture temporal, cross-dimensional, and inter-channel dependencies. However, unlike above works, this work will focus on privacy-preserving properties.

In medical, industrial, or other application areas, multivariate time series data always have high privacy protection requirements and cannot be directly analyzed in plaintext form. To preserve privacy of multivariate time series data, numerous studies focus on the privacy-preserving time series query or classification. Zheng *et al.* [33] presented an efficient and privacy-preserving similarity range query scheme over encrypted time series data. In their scheme, they use the time warp edit distance (TWED) as the similarity metric. Then they organize time series data into a *k*d-tree by leveraging TWED's triangle inequality, and design the similarity range query algorithm. Wang *et al.* [34] also proposed an efficient and privacy-preserving arbitrary polygon range query scheme for dynamic and time-series location data. This scheme introduces a privacy-preserving arbitrary polygon range query algorithm based on symmetric homomorphic encryption (SHE) technique, which may be of independent interest for other location application scenarios.

Liu *et al.* [11] presented the first privacy-preserving DTW-based analytics system on distributed medical time series data. This system is efficient, but can only analyze customized medical data and can only process univariate time series data. Zheng *et al.* [12] proposed a privacy-preserving time-series activity classification algorithm by leveraging hidden markov model (HMM). Their scheme is not only privacy-preserving but also efficient in computational cost. However, this scheme is helpless for the multivariate time series classification problem. Mercier *et al.* [13] evaluated the transferability of recent state-of-the-art privacy-preserving machine learning methods to the time-series domain. Specifically, they performed simple tests of the efficiency of several methods, such as differential privacy, federated learning, secret sharing and homomorphic encryption, on time series data.

Shi *et al.* [35] proposed a construction in which an untrusted data aggregator is able to compute the sum of a group of participants' encrypted values in time-series data without sacrificing data privacy. Guan *et al.* [36] presented an efficient and privacy-preserving max aggregation query scheme for time-series data in IoT scenarios. They leveraged two encryption techniques to encrypt the time-series data and achieve the ranged max aggregation query with $O(\log L)$ time complexity, where L is the range length of the query. Xu *et al.* [37] also presented a privacy-preserving time-series data aggregation scheme with a semi-trusted authority, which supports arbitrary aggregate functions and fault tolerance to enhance the reliability and scalability of data aggregation.

Overall, existing privacy-preserving time-series analysis schemes still have several limitations. Existing DTW-based methods mainly focus on univariate time-series analysis or similarity queries and cannot directly support multivariate time-series classification. Meanwhile, existing time-series classification methods mainly focus on classification performance rather than privacy protection. Moreover, general secure computation frameworks still incur considerable overhead when applied to DTWD computation and nearest-neighbor classification. Therefore, achieving efficient classification while protecting the privacy of multivariate medical time-series data remains an important research problem.

3 Problem Formulation

In this section, we define the system model and security model. Furthermore, we introduce the main notations used in this paper.

3.1 System Model

As illustrated in Figure 1, this paper considers a typical cloud-based system which involves three types of entities, i.e., a trusted service provider, two cloud server P_0 and P_1, and a query user.

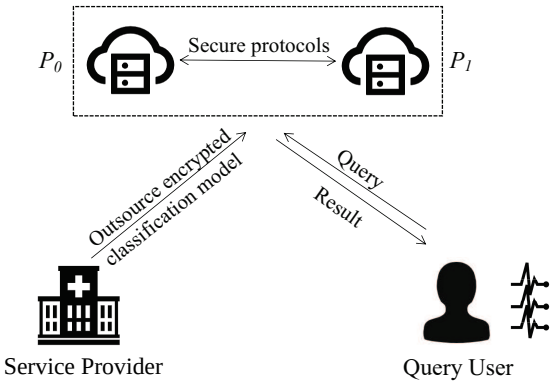

Figure 1: System model.

- **Service Provider**: The service provider has massive multivariate time series data and provides time series data classification services to users. In healthcare scenarios, the service provider may be medical centers, hospitals, or healthcare institutions, which are responsible for evaluating users' health conditions by classifying their multivariate time series. Specifically, the service provider has a personalized classification model. When a user sends his/her health record data (time series data) to request health status, the service provider identifies the class label through this model. Restricted by computational resources and facilities, the service provider may seek assistance from cloud servers. That is, the service provider outsources the classification model to cloud servers and delegates the servers to provide classification services for users. Additionally, to preserve the privacy of the classification model, it encrypts the classification model before outsourcing it.
- **Query User**: Query users can enjoy the query services provided by service provider and classify their own query data. For example, in healthcare scenarios, a query user U is equipped with IoT devices, to collect his/her time series health record data to monitor his/her health. Then, the user sends the time series data (encrypted) to request the cloud servers to perform health queries. Thereafter, the query user can get results, which indicate their own health status.
- **Cloud Servers**: Two cloud servers $P = \{P_0, P_1\}$ store the encrypted classification model and perform the query process. In practice, they can be deployed on independent

cloud platforms to provide secure computation services. Upon receiving query requests, the two servers collaboratively execute the proposed protocols and communicate during secure computation, while neither server can access the plaintext model or query data.

3.2 Security Model

We consider the problem of privacy preserving multivariate medical time series classification under the semi-honest adversary model (a.k.a., honest-but-curious adversary model). That is, we assume that parties are semi-honest ones who correctly follow the protocol specification, yet attempt to learn additional information by analyzing the transcript of messages received during the execution. Taking servers P_0 and P_1 as an example, they honestly store the classification model and respond to queries sent by users but are interested in the content of the model, user data, and classification results. In addition, our work assumes that the two servers do not collude.

Although malicious behaviors or collusion between cloud servers may exist in practical healthcare cloud environments, stronger security models usually introduce additional computational and communication overhead. Therefore, this work focuses on the semi-honest and non-colluding setting to balance privacy protection and efficiency. Supporting stronger security models will be considered in future work.

Under the foregoing system model and security model, our design goal is to introduce a privacy-preserving multivariate medical time series classification scheme. On the one hand, the scheme should keep the service provider's classification model secret to prevent the cloud servers from learning sensitive information. On the other hand, the scheme should protect the confidentiality of the query data and only allow explicitly defined leakage during the classification process. Specifically, under the semi-honest and non-colluding model, each cloud server only holds its own secret shares and cannot learn the original time-series data, classification labels, classification model parameters, or intermediate computation values beyond the leakage defined by the leakage function.

3.3 Notations and Definitions

Definition 1. *Univariate Time Series* $S = \{s_1, s_2, ..., s_n\}$ consists of individual observations recorded in sequence. Univariate time series in the medical field refers to a set of medical data that is collected and recorded over time, such as measurements of vital signs, laboratory results, or symptoms. The data is typically collected at regular intervals, such as daily, weekly, or monthly, and is used to track changes in a patient's condition or to monitor the progression of a disease.

Definition 2. *Multivariate Time Series (MDTS)* consist of n individual time series where each time series has d $(d \geq 2)$ observations, expressed as Equation (1):

$$X = \{X_1, X_2, ..., X_n\} \text{ where } X_i = (x_{i,1}, ..., x_{i,d}). \quad (1)$$

In the medical field, multivariate time series refers to medical data that is recorded with multiple observations over time, where each observation is a set of variables that are measured at the same time. For example, in electroencephalography (EEG), data from multiple channels are recorded simultaneously, with each channel representing electrical activity at a different location on the scalp.

Definition 3. *Dependent Dynamic Time Warping distance* (DTW_D) is proposed by Shokoohi-Yekta *et al.* [24] for multivariate problems. Given two multivariate time series, a data sequence $X = \{X_1, X_2, ..., X_n\}$ and query sequence $Q = \{Q_1, Q_2, ..., Q_m\}$, where $X_i = (x_{i,1}, ..., x_{i,d})$ and $Q_j = (q_{j,1}, ..., q_{j,d})$ with the same dimensions, the DTW_D distance is calculated as Equation (2):

$$D(i,j) = distance(X_i, Q_j) + min \begin{cases} D(i-1, j-1) \\ D(i-1, j) \\ D(i, j-1) \end{cases}, \quad (2)$$

in which $distance(X_i, Q_j) = \sum_{k=1}^{d} (x_{i,k} - q_{j,k})^2$. The final distance is $D(m,n)$.

3.3.1 Notations

The main notations used throughout this paper are shown in Table 1, and other special symbols will be introduced when using.

Table 1: The Summary of Notations.

Notations	Comments
$\langle x \rangle^A$	Arithmetic sharing of the value x.
$\langle x \rangle^B$	Boolean sharing of the value x.
$\mathscr{T}$	The database of classification model.
N	The number of values contained in the database.
$\mathscr{T}[i]$	Each sequence of database.
$\overline{\mathscr{T}[i]}$	The *encrypted* from of $\mathscr{T}[i]$
l	The length of $\mathscr{T}[i]$.
Q	The user's query sequence.
m	The length of Q.
U	The upper bound sequence of Q.
L	The lower bound sequence of Q.
d	The dimension of a time series.
$\mathscr{C}[0, ..., r-1]$	The candidate sequences $\mathscr{C}[0], ..., \mathscr{C}[r-1]$.
$A[0, ..., r]$	The value of DTW_D distance between Q and $\mathscr{C}[i]$.
η	The filter rate.
ϕ	A threshold.
π	a uniformly random permutation of $\{1, ..., N\}$.
ζ	The leakage function.

4 Preliminaries

4.1 Multivariate time series classification

At a high level, the MTSC algorithm first computes the dependent warping (DTW_D) distance $distance(Q, \mathscr{T}[i])$ between each sequence $\mathscr{T}[i]$ in the database $\mathscr{T}$ and the query sequence Q. Then it calculates $min\_index(distance(Q, \mathscr{T}[1]), ..., distance(Q, \mathscr{T}[N]))$ to obtain the minimum distance $dist_{min}$ and the corresponding sequence index $index_{min}$. The output of the algorithm is

$(dist_{min}, index_{min})$, and the final class label is determined according to the class information associated with the selected sequence. The algorithm is shown in Algorithm 1.

Algorithm 1 Plaintext multivariate time series classification algorithm

Input:

Query sequence: $Q = \{Q_1, Q_2, ..., Q_m\}$, where $Q_j = \{(q_{1,1}, ..., q_{1,d}), ..., (q_{m,1}, ..., q_{m,d})\}$

Classification model database: $\mathscr{T} = \{\mathscr{T}[1], ..., \mathscr{T}[N]\}$, where $\mathscr{T}[i] = \{(t^i_{1,1}, ..., t^i_{1,d}), ..., (t^i_{l,1}, ..., t^i_{l,d})\}$

Output: The minimum distance and the corresponding sequence index **res** $= (dist_{min}, index_{min})$

for $(i = 1; i \le N; i++)$ **do**
  Set a $m * l$ matrix $dist_{matrix}$ as all 0
  for $(j = 1; j \le m; j++)$ **do**
    for $(w = 1; w \le l; w++)$ **do**
      $dist = 0$;
      for $(v = 1; v \le d; v = v + 1)$ **do**
        $dist = dist + (q_{j,v} - t^i_{w,v})^2$
      $dist_{matrix}[j][w] = dist$;
  Initialize the DTW matrix: $D(0,0) = 0, D(i,0) = +\infty$ for $1 \le i \le m$, and $D(0,j) = +\infty$ for $1 \le j \le l$.
  for $(i = 1; i \le m; i++)$ **do**
    for $(j = 1; j \le l; j++)$ **do**
      $D_{min} = min\{D(i-1, j-1), D(i-1, j), D(i, j-1)\}$;
      $D(i,j) = dist_{matrix}[i][j] + D_{min}$;
  $distance(Q, \mathscr{T}[i]) = D(m, l)$;
$(dist_{min}, index_{min}) \leftarrow min\_index(distance(Q, \mathscr{T}[1]), \cdots, distance(Q, \mathscr{T}[N]))$;
res $= (dist_{min}, index_{min})$

The computational complexity of calculating the DTW_D distance between Q and one sequence $\mathscr{T}[i]$ is $O(mld)$, where m, l, and d denote the query length, sequence length, and dimensionality, respectively.

To improve the efficiency, we adopt a lower bound function LB_{mv} proposed by Rath *et al.* [38] into our scheme, which can filter out sequences that meet the preset threshold. Rath *et al.* proved that for any two sequence Q and X of the same length, $LB_{mv}(Q,X) \le DTW_D(Q,X)$ [38]. To compute the lower bound $LB_{mv}(Q,X)$, two envelopes $L = \{l_1, ..., l_m\}$ and $U = \{u_1, ..., u_m\}$ are constructed according to the query sequence $Q = \{(q_{1,1}, ..., q_{1,d}), ..., (q_{m,1}, ..., q_{m,d})\}$:

$$u_i = (u_{i,1}, ..., u_{i,d}), \tag{3}$$

where $u_{i,j} = max(q_{i-r,j}, q_{i+r,j})$,

$$l_i = (l_{i,1}, ..., l_{i,d}), \tag{4}$$

where $l_{i,j} = min(q_{i-r,j}, q_{i+r,j})$, and r satisfies $j - r \le i \le j + r$. Then we have,

$$LB_{mv}(Q,X) = \sum_{i=1}^{m} \sum_{j=1}^{d} \begin{cases} (x_{i,j} - u_{i,j})^2 & if \ \ x_{i,j} \ge u_{i,j} \\ (x_{i,j} - l_{i,j})^2 & if \ \ x_{i,j} \le l_{i,j} \\ 0 & otherwise \end{cases}. \tag{5}$$

Note that $X = \{(x_{1,1}, ..., x_{1,d}), ..., (x_{m,1}, ..., x_{m,d})\}$.

4.2 Secure Multiparty Computation

Secure multi-party computation (SMPC) allows n parties to securely compute a function $f(x_1, \cdots, x_n)$ on their respective inputs without revealing the inputs to each other. To realize secure computation, many different techniques have been developed, such as secret sharing [39], oblivious transfer [40], garbled circuit (GC) [41] and so on.

CryptFlow2 framework [42] is a cryptographic framework that implements secure inference over realistic Deep Neural Networks (DNNs) using secure 2-party computation (2PC). It provides general secure computation primitives, such as secure comparison, division, F_{DRelu}, and F_{MUX}, which serve as building blocks for higher-level secure protocols. Based on these primitives, we implement several basic secure operations, including SCMP, SMinP/SMaxP, and SSP, which serve as building blocks for our scheme. The functionality F_{DRelu} takes arithmetic shares of a as input and returns boolean shares of $DRelu(a)$, where $DRelu(a) = 1$ if $a \ge 0$ and 0 otherwise. The functionality F_{MUX} selects the input share according to the choice bit c. Furthermore, these basic operations are used to construct our privacy-preserving MTSC protocols, including SFP, SDTWDP, and SCFP. Specifically, SFP performs secure candidate filtering, SDTWDP computes secure DTWD distance, and SCFP enables secure classification.

To avoid numerical overflow during secure computation, all inputs and intermediate results are represented using fixed-point encoding with sufficient bit length. The bit width is selected according to the range of intermediate values in DTWD computation and secure protocol execution. After secure multiplication, truncation is performed to maintain precision and avoid overflow under the selected precision setting.

4.2.1 Secure Comparing Protocol (SCMP)

Assuming that P_0 inputs a set of shared values $(\langle x\rangle_0^A, \langle y\rangle_0^A)$ and P_1 inputs $(\langle x\rangle_1^A, \langle y\rangle_1^A)$, where $(\langle x\rangle_0^A, \langle x\rangle_1^A)$ and $(\langle y\rangle_0^A, \langle y\rangle_1^A)$ are arithmetic sharing of values x and y, respectively. Secure comparing protocol(SCMP) is to realize the function that returns boolean secret shares $(\langle c\rangle_0^B, \langle c\rangle_1^B)$ of c, where $c = 1$ if $x \ge y$ and 0 otherwise. SCMP works as follows:

1. P_b computes the secret difference of x and y as $\langle a\rangle_b^A = \langle x\rangle_b^A - \langle y\rangle_b^A$ locally.
2. P_0 and P_1 jointly invoke an instance of F_{DRelu} which inputs $\langle a\rangle_b^A$ and outputs the boolean shared value of the comparison result bit $\langle c\rangle_b^B$.

Correctness: The correctness of this protocol is obvious. $\langle a\rangle_b^A$ is computed locally by P_b as subtracting $\langle y\rangle_b^A$ from $\langle x\rangle_b^A$. Thus, $a = \langle a\rangle_0^A + \langle a\rangle_1^A$ is the difference between x and y. Then P_b invokes F_{DRelu} on $\langle a\rangle_b^A$, which is a secure and correctness function for computing the shared value of the boolean shared bit comparison result $\langle c\rangle_b^B$. This ensures that the comparison is performed correctly. Therefore, the protocol is correct based on the correctness of F_{DRelu}.

4.2.2 The Secure Minimum (Maximum) Computing Protocol (SMinP/SMaxP)

As the same as SCMP, assuming that P_0 and P_1 input $(\langle x\rangle_0^A, \langle y\rangle_0^A)$ and $(\langle x\rangle_1^A, \langle y\rangle_1^A)$, respectively. The secure minimum (maximum) computing protocol (SMinP/SMaxP) is to realize the function that returns the arithmetic shares $\langle min(x,y)\rangle_b^A$ or $\langle max(x,y)\rangle_b^A$. Specifically, SMinP/SMaxP works as follows:

1. P_b jointly invokes F_{DRelu} with input $\langle a\rangle_b^A$ to output $\langle c\rangle_b^B$, where $\langle a\rangle_b^A = \langle x\rangle_b^A - \langle y\rangle_b^A$ calculated locally by P_b. Thus, if $x \geq y$, $c = 1$, otherwise $c = 0$.
2. P_b jointly invokes F_{MUX} with input $\langle a\rangle_b^A$ and $\langle c\rangle_b^B$ to output $\langle t\rangle_b^A$. Notice that, if $x \geq y$, we have $t = x - y$, otherwise $t = 0$.
3. P_b computes $\langle min(x,y)\rangle_b^A = \langle x\rangle_b^A - \langle t\rangle_b^A$ ($\langle max(x,y)\rangle_b^A = \langle y\rangle_b^A + \langle t\rangle_b^A$) locally.

Correctness: According to the definition of F_{DRelu}, if $x \geq y$, we have $c = 1$, otherwise $c = 0$. Then, P_0 and P_1 jointly invoke F_{MUX} with the inputs a and c (shared form). According to the definition of F_{MUX}, we have that $t = a = x - y$ if $x \geq y$, otherwise $t = 0$.

Assuming $x \geq y$, let $t = x - y$. Then $x - t = y$ represents the minimum value between x and y, while $y + t = x$ represents the maximum value between them. Alternatively, if $x < y$, set $t = 0$. Then in this case, $x - t = x$ remains the minimum value between x and y, and $y + t = y$ still remains the maximum value between x, y.

That proves the correctness of SMinP/SMaxP.

4.2.3 The Secure Sorting Protocol (SSP)

Suppose that two tuples (x, I_x) and (y, I_y) are shared by cloud P_0 and P_1. Here, x and y are the data that should be compared, I_x and I_y are the corresponding indexes of x and y. The goal of SSP is to *securely* sort the two tuples, i.e., (x, I_x) and (y, I_y) are sorted as (x', I'_x) and (y', I'_y). Here, $x' = min(x,y)$, $y' = max(x,y)$, $I'_x = I_{min(x,y)}$ and $I'_y = I_{max(x,y)}$. Notice that, (x, I_x), (y, I_y), (x', I'_x) and (y', I'_y) are all secretly shared among P_0 and P_1, and P_0 cannot retrieve any original information about (x, I_x), (y, I_y), (x', I'_x) and (y', I'_y).

The secure sorting protocol (SSP) is shown in Algorithm 2. This protocol is very similar to invoke SMinP and SMaxP protocols simultaneously. According to the definition of F_{DRelu}, $v = 1$ if $x \geq y$, otherwise, $v = 0$. Thus, according to F_{MUX}, we have

$$t = \begin{cases} x - y & if \;\; x \geq y, \\ 0 & if \;\; x < y, \end{cases} \tag{6}$$

and

$$g = \begin{cases} I_x - I_y & if \;\; x \geq y, \\ 0 & if \;\; x < y. \end{cases} \tag{7}$$

Then, it is easy to have $x - t = min(x,y)$, $y + t = max(x,y)$, $I_x - g = I_{min(x,y)}$, $I_y + g = I_{max(x,y)}$. This also proves the correctness of SSP.

Algorithm 2 Secure sorting protocol (SSP)

Input: P_b inputs $(\langle x\rangle_b^A, \langle I_x\rangle_b^A)$ and $(\langle y\rangle_b^A, \langle I_x\rangle_b^A)$

Output: P_b outputs $(\langle x'\rangle_b^A, \langle I'_x\rangle_b^A)$ and $(\langle y'\rangle_b^A, \langle I'_y\rangle_b^A)$, where $x' = min(x,y)$, $y' = max(x,y)$, $I'_x = I_{min(x,y)}$, $I'_y = I_{max(x,y)}$.

1: P_b locally computes $\langle w\rangle_b^A = \langle x\rangle_b^A - \langle y\rangle_b^A$.
2: P_b locally computes $\langle h\rangle_b^A = \langle I_x\rangle_b^A - \langle I_y\rangle_b^A$.
3: P_0 and P_1 jointly invoke F_{DRelu} with input $\langle w\rangle_b^A$ to get $\langle v\rangle_b^B$, respectively.
4: P_0 and P_1 jointly invoke F_{MUX} with input $\langle w\rangle_b^A$ and $\langle v\rangle_b^B$ to get $\langle t\rangle_b^A$, respectively.
5: P_0 and P_1 jointly invoke F_{MUX} with input $\langle h\rangle_b^A$ and $\langle v\rangle_b^B$ to get $\langle g\rangle_b^A$, respectively.
6: P_b locally computes $\langle x'\rangle_b^A = \langle x\rangle_b^A - \langle t\rangle_b^A$, $\langle y'\rangle_b^A = \langle y\rangle_b^A + \langle t\rangle_b^A$, $\langle I'_x\rangle_b^A = \langle I_x\rangle_b^A - \langle g\rangle_b^A$, and $\langle I'_y\rangle_b^A = \langle I_y\rangle_b^A + \langle g\rangle_b^A$.

5 Secure Multivariate Time Series Classification

In this section, we propose the detailed privacy-preserving multivariate time series classification scheme (SecMTSC). We first give an overview design of our scheme. Then we present each stage contained in elaboration.

5.1 Scheme Overview

At a high level, our scheme consists of five stages, i.e., data outsourcing, user query, secure filtering, secure classification, and result retrieval. During the SecMTSC processes, the cloud server P_b learns nothing of the service provider's classification model and the user's query data. The secure classification process consists of three sequential protocols. SFP first filters unnecessary sequences and outputs the candidate set $\mathscr{C}$. Then, SDTW_DP computes the exact DTW_D distances of candidate sequences. Finally, SCFP finds the minimum distance and the corresponding sequence index to obtain the classification result.

In the data outsourcing stage, the service provider outsources its classification model to two cloud servers P_b. Specifically, the service provider leverages the secret sharing method to split its data into two sharing slices and uploads them to two cloud servers, respectively.

Then, in the user query stage, a user sends its query sequence in a secretly sharing manner to the cloud servers. Moreover, the query user also generates upper and lower bound sequences of the query sequence for cloud servers to perform secure filter to improve the efficiency.

In the secure filtering stage, the cloud servers filter candidate sequences based on the upper and lower bound sequences of the query sequence. To do this, we propose a secure filtering protocol (SFP). The SFP calculates a boundary distance named LB_{mv} before computing the DTW_D distance in order to quickly exclude unpromising sequences. Only sequences that meet certain criteria in the classification model can calculate the exact DTW_D values with the query sequence. Specifically, cloud servers P_b jointly calculate the LB_{mv} distance value between each sequence in T and the query sequence Q. Then they compare each LB_{mv} with a threshold ϕ given by the user,

and only sequences with the LB_{mv} smaller than the threshold ϕ can be used as candidate sequences.

After the secure filtering stage, both P_b cooperate to complete the classification stage. To deploy this stage securely, our main insight is to build a secure DTW_D distance protocol (SDTW_DP) and a secure classification protocol (SCFP). The secure DTW_D distance protocol (SDTW_DP) accurately calculates the DTW_D distance between the query sequence and the candidate sequence is screened during the filtering stage. The secure classification protocol (SCFP) finds the minimum of the exact DTW_D distance value and obtains the secret index of the minimum sequence (in the secretly shared form).

With the outputs of two cloud servers in the secure classification stage, the query user retrieves the final result in the result retrieval stage with negligible computation cost.

5.2 Data Outsourcing Stage

In this stage, the service provider outsources its classification model to the two cloud servers P_0 and P_1. This process mainly uses arithmetic sharing to share data. For arithmetic sharing, an ℓ-bit value is shared additively in the ring $\mathbb{Z}_{2^\ell}$ as the sum of two values. Specifically, for each sequence $\mathscr{T}[i] = \{(t^i_{1,1},...,t^i_{1,d}),...,(t^i_{l,1},...,t^i_{l,d})\}$, $\mathscr{T}[i]$ records the multidimensional time series data sampled at each time, which is a multidimensional matrix, and each data t has ℓ bits. The service provider randomly generates a $r \in_R \mathbb{Z}_{2^\ell}$ for each data t, sets $\langle t\rangle^A_0 = r$, and calculates $\langle t\rangle^A_1 = t - r$. Then it respectively sends corresponding shares to two cloud servers for $b \in \{0,1\}$. The detail of this stage is shown in Algorithm 3.

Algorithm 3 Data owner outsourcing process

for $i = 1$ **to** N **do**
  for $j = 1$ **to** l **do**
    for $k = 1$ **to** d **do**
      Selects a random number $r \in_R \mathbb{Z}_{2^\ell}$.
      Sets $\langle t\rangle^A_0 = r$, and calculates $\langle t\rangle^A_1 = t^i_{j,k} - r$.
      Sends $\langle t\rangle^A_b$ to P_b for $b \in \{0,1\}$.

5.3 User Query Stage

In this stage, a query user may try to identify the category to which his/her data belongs. For instance, in healthcare scenario, the user may want to check his/her health status by launching a query with his/her sequence data Q. To do this, he/she first computes the upper bound U and lower bound L according to the Equations (3) and (4) for Q. Note that, the upper bound U and lower bound L are used for the cloud servers to filter suitable candidate sequences from the large classification model to improve the efficiency of the scheme. After that, the user secretly shares the above information and sends $\langle Q\rangle^A_b$, $\langle U\rangle^A_b$, $\langle L\rangle^A_b$, and $\langle \phi\rangle^A_b$ to cloud server P_b, where ϕ is the threshold for the query sequence Q.

5.4 Secure Filtering Stage

When two servers receive the query request from the user, a series of operations should can be executed between P_0 and P_1. The goal of this stage is to retrieve the indices of the candidate sequences that are closer to the user query sequences, so as to improve the efficiency of the proposed scheme. To do this, we propose a secure square of the difference protocol (SSDP) and a secure filtering protocol (SFP). The SSDP is a fundamental protocol to construct SFP. The SFP realizes the calculation of the LB_{mv} distance and the selection of candidate sequences. Specifically, SSDP computes the squared differences, while SCMP and F_{MUX} securely select the valid boundary distances. The details of SSDP and SFP are shown in Algorithm 4 and Algorithm 5.

Algorithm 4 Secure square of the difference protocol (SSDP)

Input: Each cloud server P_b inputs $\langle x\rangle^A_b, \langle y\rangle^A_b$, respectively
Output: P_b outputs $\langle d\rangle^A_b$
1: P_b locally calculates $\langle a\rangle^A_b = \langle x\rangle^A_b - \langle y\rangle^A_b$.
2: P_b jointly calculates $\langle d\rangle^A_b$ by invoking secure multiplication protocol $Mul(\langle a\rangle^A_b \cdot \langle a\rangle^A_b)$.

Algorithm 5 Secure filtering protocol (SFP)

Input: $\langle \mathscr{T}\rangle^A_b$, $\langle U\rangle^A_b$, $\langle L\rangle^A_b$, $\langle \phi\rangle^A_b$.
Output: A index set $\mathscr{C}$.
1: **for** $i = 1$ **to** N **do**
2: Let $\mathscr{T}[i] = \{(t_{1,1},...,t_{1,d}),...,(t_{m,1},...,t_{m,d})\}$
3: Set $\langle LB_{mv}\rangle^A_b \leftarrow 0$
4: **for** $j = 1$ **to** m **do**
5: Set $\langle D\rangle^A_b \leftarrow 0$
6: **for** $k = 1$ **to** d **do**
7: $\langle d_u\rangle^A_b \leftarrow$ **SSDP**($\langle t_{j,k}\rangle^A_b, \langle u_{j,k}\rangle^A_b$).
8: $\langle d_l\rangle^A_b \leftarrow$ **SSDP**($\langle t_{j,k}\rangle^A_b, \langle l_{j,k}\rangle^A_b$).
9: $\langle c\rangle^B_b \leftarrow$ **SCMP**($\langle t_{j,k}\rangle^A_b, \langle u_{j,k}\rangle^A_b$).
10: $\langle z\rangle^B_b \leftarrow$ **SCMP**($\langle l_{j,k}\rangle^A_b, \langle t_{j,k}\rangle^A_b$).
11: $\langle d_1\rangle^A_b \leftarrow F_{MUX}$($\langle d_u\rangle^A_b, \langle c\rangle^B_b$).
12: $\langle d_2\rangle^A_b \leftarrow F_{MUX}$ ($\langle d_l\rangle^A_b, \langle z\rangle^B_b$).
13: $\langle d\rangle^A_b \leftarrow \langle d_1\rangle^A_b + \langle d_2\rangle^A_b$.
14: $\langle D\rangle^A_b \leftarrow \langle D\rangle^A_b + \langle d\rangle^A_b$.
15: $\langle LB_{mv}\rangle^A_b \leftarrow \langle LB_{mv}\rangle^A_b + \langle D\rangle^A_b$.
16: $\langle s\rangle^B_b \leftarrow$ **SCMP**($\langle \phi\rangle^A_b, \langle LB_{mv}\rangle^A_b$).
17: P_0 and P_1 jointly retrieve s and add i into $\mathscr{C}$ if $s = 1$.

Assuming P_b holds the secret shared values $\langle x\rangle^A_b$, $\langle y\rangle^A_b$ of x and y. To compute the secret shares of $(x-y)^2$, the SSDP should leverage a multiplication protocol $Mul(\langle a\rangle^A \cdot \langle a\rangle^A)$ [43]. As a basic MPC operation, Mul requires the two cloud servers to pre-compute multiplication triples, then P_b can easily and locally calculate its secret share $\langle d\rangle^A_b$ of $(x-y)^2$.

The secure filtering protocol (SFP) aims to calculate the distance LB_{mv} between Q and each data sequence $\mathscr{T}[i]$ in database $\mathscr{T}$ according to Equation (5), and then compares LB_{mv} with the threshold ϕ. If LB_{mv} is less than ϕ, then the corresponding sequence will be acted as a candidate sequence and added to the candidate set $\mathscr{C}$.

From Equation (5), we can observe that the value of LB_{mv} is the sum of a series of values between $(t_{i,j} - u_{i,j})^2$, $(t_{i,j} - l_{i,j})^2$, and 0. To securely achieve this, we use two temp values d_1 and d_2 to store intermediate results. According to the line 7-12 of the algorithm and the definition of F_{MUX}, we

have

$$d_1 = \begin{cases} (t_{j,k} - u_{j,k})^2 & if \ \ t_{j,k} \geq u_{j,k}, \\ 0 & otherwise, \end{cases} \tag{8}$$

$$d_2 = \begin{cases} (t_{j,k} - l_{j,k})^2 & if \ \ t_{j,k} \leq l_{j,k}, \\ 0 & otherwise. \end{cases} \tag{9}$$

Furthermore, since $u_{j,k}$ is greater than $l_{j,k}$, the sum of d_1 and d_2 exactly fits the definition of LB_{mv} given above. Then P_b can easily and locally computes the secret shared form LB_{mv} of Q and $\mathscr{T}[i]$. After that, by comparing LB_{mv} and the threshold ϕ, P_b can decide whether the sequence indexed by i is a candidate sequence. Besides, we denote the filter rate as $\eta = 1 - |\mathscr{C}|/N$, where $|\mathscr{C}|$ is the number of candidate sequences.

The threshold ϕ controls the trade-off between filtering efficiency and classification correctness. A smaller ϕ reduces the number of candidate sequences and secure DTWD computation overhead, but may remove the true nearest neighbor. A larger ϕ improves candidate preservation but reduces filtering efficiency. Therefore, ϕ should be selected by balancing filtering efficiency and classification correctness according to the requirements of the filtering process.

The computational complexity of SFP is $O(Nld)$, where N, l, and d denote the number of sequences, sequence length, and dimensionality, respectively.

5.5 Secure Classification Stage

After the security filtering stage, P_b receives a candidate index set, and the security classification phase can be performed between P_0 and P_1. The goal of this stage is to calculate the exact DTW_D values between sequences in the candidate set and the query sequence Q, and get the classification result through the classifier. Therefore, we propose a secure DTW_D distance protocol (SDTW_DP) and a secure classification protocol (SCFP) to achieve this. The SDTW_DP is used to securely calculate accurate DTW_D values. After obtaining the value of DTW_D and the corresponding index, SCFP finds the minimum distance and its index to extract the classification result. The details of SDTW_DP and SCFP are shown in Algorithm 6 and Algorithm 7.

5.5.1 Secure DTW_D Distance Protocol (SDTW_DP)

When two servers receive the candidate index set $\mathscr{C}[0,...,r-1]$, P_b can perform the SDTW_DP to calculate the specific DTW_D value between the candidate sequence and the query sequence. The details of the SDTW_DP are shown in Algorithm 6.

For each candidate index $\mathscr{C}[i]$ $(0 \leq i < r)$ of the candidate index set $\mathscr{C}$, P_b retrieves the shared $\langle \mathscr{T}[\mathscr{C}[i]] \rangle_b^A$ locally. Then, P_b can compute the specific DTW_D value by invoking the SSDP protocol and the SMinP protocol. According to Equation (2), when calculating the square of difference value of a candidate sequence $\mathscr{T}[\mathscr{C}[i]]$ and the query sequence Q, P_b should run the SSDP protocol to get the first shared matrix $\langle MAT1 \rangle_b^A$. Next, according to the $\langle MAT1 \rangle_b^A$, two cloud servers get the second sharing matrix with $\langle MAT2 \rangle_b$ by using the SMinP protocol. At this point, the matrix $\langle MAT2_{l,m} \rangle_b^A$ is the DTW_D value between $\mathscr{T}[\mathscr{C}[i]]$ and Q. Here, $MAT1$ stores the point-wise distance values, and $MAT2$ stores the accumulated DTW values obtained by dynamic programming.

The computational complexity of SDTWDP is $O(ml)$ for each candidate sequence, where m and l denote the lengths of the query sequence and candidate sequence, respectively.

Algorithm 6 Secure DTW_D distance protocol (SDTW_DP)

Input: $\langle Q \rangle_b^A$, $\langle \mathscr{T} \rangle_b^A$ and $\mathscr{C}[i]$
Output: $\langle A[i] \rangle_b^A$

1: For each index of $\mathscr{C}[i]$, P_b retrieves $\langle \mathscr{T}[\mathscr{C}[i]] \rangle_b^A$.
2: **for** $u = 1$ to l **do**
3: **for** $v = 1$ to m **do**
4: **for** $j = 1$ to d **do**
5: $\langle temp_j \rangle_b^A \leftarrow$ **SSDP**($\langle t_{u,j} \rangle_b^A$, $\langle q_{v,j} \rangle_b^A$).
6: $\langle MAT1_{u,v} \rangle_b^A \leftarrow \sum_{j=1}^{d} \langle temp_j \rangle_b^A$.
7: **for** $u = 1$ to l **do**
8: **for** $v = 1$ to m **do**
9: **if** $u = 1$ and $v = 1$ **then**
10: $\langle MAT2_{u,v} \rangle_b^A \leftarrow \langle MAT1_{u,v} \rangle_b^A$.
11: **if** $u = 1$ and $v > 1$ **then**
12: $\langle MAT2_{u,v} \rangle_b^A \leftarrow \langle MAT1_{u,v} \rangle_b^A + \langle MAT2_{u,v-1} \rangle_b^A$.
13: **if** $u > 1$ and $v = 1$ **then**
14: $\langle MAT2_{u,v} \rangle_b^A \leftarrow \langle MAT1_{u,v} \rangle_b^A + \langle MAT2_{u-1,v} \rangle_b^A$.
15: **if** $u>1$ and $v>1$ **then**
16: $\langle a_1 \rangle_b^A \leftarrow$ **SMinP**($\langle MAT2_{u-1,v-1} \rangle_b^A, \langle MAT2_{u-1,v} \rangle_b^A$).
17: $\langle min \rangle_b^A \leftarrow$ **SMinP**($\langle a_1 \rangle_b^A, \langle MAT2_{u,v-1} \rangle_b^A$).
18: $\langle MAT2_{u,v} \rangle_b^A \leftarrow \langle MAT1_{u,v} \rangle_b^A + \langle min \rangle_b^A$.
19: $\langle A[i] \rangle_b^A \leftarrow \langle MAT2_{l,m} \rangle_b^A$.

5.5.2 Secure Classification Protocol (SCFP)

The goal of the secure classification protocol is to find the minimum DTW_D distance from the resulting DTW_D values and the corresponding candidate sequence index. Details of the SCFP are introduced in Algorithm 7. In general, the main insight of SCFP is that P_0 and P_1 jointly find the minimum DTW_D value and the corresponding index between the candidate sequences and the query sequence. To do so, they compare other data pairs (value and index) one by one by invoking SSP. Here, $A[i]$ denotes the DTW_D distance of the i-th candidate sequence, and (A_{min}, ID_{min}) denotes the current minimum distance and its corresponding index.

Specifically, P_b first sets the the minimum value and index pairs $(\langle A_{min} \rangle_b^A, \langle ID_{min} \rangle_b^A)$ as the first pair of candidate sequence set $(\langle A[0] \rangle_b^A, \langle \mathscr{C}[0] \rangle_b^A)$. After that, SCFP leverages SSP to obtains the minimum value and the corresponding index according to the line 2-4 of the algorithm. By invoking SSP, the two input pairs $(\langle A_{min} \rangle_b^A, \langle ID_{min} \rangle_b^A)$ and $(\langle A[i] \rangle_b^A, \langle \mathscr{C}[i] \rangle_b^A)$ will be compared and the smaller one pair will be stored in $(\langle A'_{min} \rangle_b^A, \langle ID'_{min} \rangle_b^A)$ according to the functionality of SSP. Then, $(\langle A_{min} \rangle_b^A, \langle ID_{min} \rangle_b^A)$ will be reset as the smaller pair $(\langle A'_{min} \rangle_b^A, \langle ID'_{min} \rangle_b^A)$ for the next comparison. At the end of the loop, $(\langle A_{min} \rangle_b^A, \langle ID_{min} \rangle_b^A)$ is the exact final minimum DTW_D value and the index. Then, P_0 and P_1 can jointly retrieve the index ID_{min} of the sequence with the minimum distance, which is used to determine the classification result according to the class information associated with the selected sequence.

The computational complexity of SCFP is $O(|\mathscr{C}|)$, where $|\mathscr{C}|$ denotes the number of candidate sequences.

Algorithm 7 Secure classification protocol (SCFP)

Input: $\langle A[0,..,r-1]\rangle_b^A$, $\langle \mathscr{C}[0,...,r-1]\rangle_b^A$
Output: $\langle result\rangle_b^A$
1: $(\langle A_{min}\rangle_b^A, \langle ID_{min}\rangle_b^A) \leftarrow (\langle A[0]\rangle_b^A, \langle \mathscr{C}[0]\rangle_b^A)$.
2: **for** $i = 1$ to $r-1$ **do**
3: $((\langle A'_{min}\rangle_b^A,\ \langle ID'_{min}\rangle_b^A),\ (\langle A'[i]\rangle_b^A,\ \langle \mathscr{C}'[i]\rangle_b^A)) \leftarrow$ **SSP**$((\langle A_{min}\rangle_b^A,\ \langle ID_{min}\rangle_b^A),\ (\langle A[i]\rangle_b^A,\ \langle \mathscr{C}[i]\rangle_b^A))$.
4: $(\langle A_{min}\rangle_b^A, \langle ID_{min}\rangle_b^A) \leftarrow (\langle A'_{min}\rangle_b^A,\ \langle ID'_{min}\rangle_b^A)$.
5: P_0 and P_1 jointly retrieve ID_{min} and then get the final result
$\langle result\rangle_b^A \leftarrow \langle \mathscr{T}[ID_{min}]\rangle_b^A$.

5.6 Result Retrieval Stage

According to our protocol, the final result is shared between two servers. After the end of the protocol, P_b sends the respective results $\langle result\rangle_b^A$ to the user. The user recovers the final result locally, i.e., $result = \langle result\rangle_0^A + \langle result\rangle_1^A$ which is the predicted state of health.

6 Security Analysis

In this section, we first give an overall security analysis of our scheme, then we carry out the formal security proofs of the components of our scheme.

Leakage function of our scheme. According to the protocol execution, the leakage function of SecMTSC is defined as

$$\mathscr{L} = (N, l, |Q|, L, U, \mathscr{C}, ID_{min}), \tag{10}$$

where N denotes the number of time-series samples in the database, l denotes the length of each time series, $|Q|$ denotes the length of the query sequence, L and U denote the lower and upper bounds used in the filtering process, $\mathscr{C}$ represents the candidate index set revealed during SFP, and ID_{min} represents the minimum-distance sequence index reconstructed during SCFP.

Apart from the leakage explicitly defined in $\mathscr{L}$, the cloud servers cannot obtain the original time-series data, classification model parameters, labels, or intermediate computation values.

The leakage information of our scheme. We conduct a specific analysis of information leakage according to the user's query process. Since the query process does not involve the service provider (medical center, etc.), we only need to analyze the information leakage in the process of outsourcing data from the service provider to the two servers.

In the data outsourcing stage, the classification model of the service provider is secretly shared between the two cloud servers P_0 and P_1. Since the two servers are assumed to be non-colluding, the private data cannot be reconstructed by any single server. However, the total number of data items in the database N and the length l of each time series will be known by the cloud servers.

After the user initiating a query, the query sequence Q, the upper bound U, the lower bound L, and the similarity threshold ϕ are secretly shared with the cloud server. Based on the security of secret sharing, the cloud server can only get the length of Q, L, and U. Before returning the query results to the user, the following protocols need to be executed on the cloud server, i.e., the secure filtering protocol (SFP), the secure DTW_D distance protocol (SDTW_DP), and the secure classification protocol (SCFP). Next, we mainly analyze the security of these component protocols, such as SFP, SDTW_D, SCFP, etc.

We assume that the security primitives described in Section 4 are secure under the semi-honest model. It mainly includes F_{DRelu} and F_{MUX} functions, which have been proved in the literature [42]. In addition, *Mul* is also secure under the semi-honest model, which has been proven in [44]. Next, we prove the security of our protocols based on the above assumptions.

Theorem 1 *If F_{DRelu} protocol is secure under the semi-honest adversarial model, the secure comparing protocol(SCMP) is also secure under the same model.*

Proof. In SCMP, on input $\langle x\rangle_b^A$, $\langle y\rangle_b^A$ of P_b, it locally computes $\langle a\rangle_b^A$, which is the secret share of the difference between x and y. It is easily known, this process will not reveal any information of x and y. Then with the value of $\langle a\rangle_b^A$, P_0 and P_1 jointly invoke F_{DRelu} function to get the boolean shared value of the comparison result bit $\langle c\rangle_b^B$. Then, we can have if the F_{DRelu} function is correct and secure under the semi-honest adversarial model, the SCMP is also correct and secure under the same model.

Theorem 2 *If F_{DRelu} and F_{MUX} protocols are secure under the semi-honest adversarial model, the secure minimum (maximum) computing protocol(SMinP/SMaxP) is also secure under the same model.*

Proof. In SMinP/SMaxP, on input $\langle x\rangle_b^A$, $\langle y\rangle_b^A$ of P_b, it locally computes $\langle a\rangle_b^A$, and then jointly invoke F_{DRelu} function to get intermediate shared bit $\langle c\rangle_b^B$. Essentially, the above calculation process is the same as the SCMP. Thus, the information of x and y will not be leaked for now. After that, with $\langle c\rangle_b^B$ and $\langle a\rangle_b^A$, P_0 and P_1 jointly invoke F_{MUX} function and get another intermediate result $\langle t\rangle_b^A$. As well as the F_{MUX} is secure under the semi-honest adversarial model, the security of x and y can still be guaranteed. Afterwards, the shared result of SMinP/SMaxP will be locally computed by P_0 and P_1 independently. Thus, the security of SMinP/SMaxP can be reduced to the security of F_{DRelu} and F_{MUX}.

Theorem 3 *If F_{DRelu} and F_{MUX} protocols are secure under the semi-honest adversarial model, the secure sorting protocol(SSP) is also secure under the same model.*

Proof. The goal of SSP is to securely sort the two tuples, i.e., (x, I_x) and (y, I_y) are sorted as (x', I'_x) and (y', I'_y) in ascending order. During the protocol, P_0 and P_1 will only perform two types of operations, i.e., locally computation and sub protocols invoking. In the steps 1, 2 and 6 of SSP protocol in Algorithm 2, P_b locally computes intermediate or final results without sacrificing any security. In the steps 3, 4, and 5 of SSP protocol, P_0 and P_1 jointly invoke F_{DRelu} and F_{MUX} functions.

Therefore, similar to the SMinP/SMaxP, the security of SSP can also be reduced to the security of F_{DRelu} and F_{MUX}.

Theorem 4 *If Mul protocol is secure under the semi-honest adversarial model, the secure square of the difference protocol (SSDP) is also secure under the same model.*

Proof. In the SSDP, P_b locally calculates the shared difference between the inputs $\langle x\rangle_b^A, \langle y\rangle_b^A$ and then jointly invokes the *Mul* function to compute the shared square of the difference. Therefore, it is obvious that the security of SSDP can be reduced to the security of the *Mul* protocol.

Theorem 5 *If SCMP, SSDP and F_{MUX} protocols are secure under the semi-honest adversarial model, the secure filtering protocol (SFP) is also secure under the same model.*

Proof. The SFP aims to filter out the index set of sequences whose distance LB_{mv} to the query sequence is less than the threshold ϕ. To achieve efficient filtering, the candidate index set generated by SFP is considered as explicitly defined leakage in the protocol execution. In the SFP, P_0 and P_1 jointly process all sequences in the database $\mathscr{T}$. For each sequences $\langle \mathscr{T}[i]\rangle_b^A$, P_0 and P_1 first set LB_{mv} as the shared value of 0. This can be easily initialized by P_0 and P_1 independently. Then for each time slot, an intermediate variable D is also set to be the shared value of 0. During these intermediate computations, no information about the private inputs is leaked to P_0 or P_1. Then in steps 7-12 of SFP, P_0 and P_1 jointly invoke SSDP, SCMP and F_{MUX} protocols multiple times. If we assume SCMP, SSDP and F_{MUX} protocols are secure, then the SFP is also secure for now. After that, in steps 13-15, locally computations are executed by P_b independently. Finally, SCMP is invoked one more time and the result of SCMP is retrieved by P_0 and P_1, which means the distance LB_{mv} of the current sequence $\langle \mathscr{T}[i]\rangle$ between the query sequence is less than the threshold ϕ or not. Therefore, it is easily to know that the Theorem 5 holds.

Theorem 6 *If SMinP and SSDP protocols are secure under the semi-honest adversarial model, the secure DTW_D distance protocol ($SDTW_DP$) is also secure under the same model.*

Proof. This protocol aims to compute the DTW_D distance of the query sequence Q between the candidate sequence $\mathscr{T}[\mathscr{C}[i]]$. In the steps 1-6, P_b locally retrieves $\langle \mathscr{T}[\mathscr{C}[i]]\rangle_b^A$ and repeatedly invokes SSDP protocol to compute the shared matrix $\langle MAT1\rangle_b^A$. Since the two sequences are secret shared and the SSDP protocol is secure under the semi-honest adversarial model, the matrix $\langle MAT1\rangle_b^A$ will not reveal any information of the two sequences. After that, in the steps 7-18, P_b locally initializes the first row and first column of matrix $\langle MAT2\rangle_b^A$ according to $\langle MAT1\rangle_b^A$ and jointly invokes SMinP to calculate other cells of the matrix $\langle MAT2\rangle_b^A$. During this process, only local operations and SMinP are carried out. Therefore, $\langle MAT2\rangle_b^A$ will also not leak any information about the input sequences and the $SDTW_D$P is secure under the semi-honest adversarial model.

[1]https://www.timeseriesclassification.com/description.php?Dataset =AtrialFibrillation
[2]https://www.timeseriesclassification.com/description.php?Dataset =ERing
[3]https://www.timeseriesclassification.com/description.php?Dataset =BasicMotions

Theorem 7 *If SSP protocol is secure under the semi-honest adversarial model, the secure classification protocol (SCFP) is also secure under the same model.*

Proof. The security of SCFP can obviously be reduced to the security of SSP. In this protocol, for r candidate sequences with their indices $\mathscr{C}$ and the corresponding distances A, P_b runs SSP protocol $r-1$ times to get the minimum distance and the corresponding sequence index which are stored in $A[0]$ and $\mathscr{C}[0]$, respectively. The reconstructed nearest-neighbor index ID_{min} is regarded as explicitly defined leakage required for returning the classification result. Therefore, if SSP protocol is secure under the semi-honest adversarial model, then SCFP is also secure under the same model.

7 Performance Evaluation

In this section, we present the experimental results of our proposed scheme. We run experiments using three public datasets and implement them under Cryptflow2 [42] framework to benchmark runtime and communication cost.

7.1 Experimental Setting

To measure the performance of the scheme, we conduct experiments on a workstation with an Intel(R) Core(TM) i9-12900KS CPU @5.5GHz, 128GB memory and a docker container on Linux operating system. To monitor network traffic, we use the nload application under the Linux terminal. Moreover, the evaluation datasets were obtained from the UEA Multivariate Time Series Classification Archive [21]. Specifically, we conduct experiments on three datasets, i.e., AtrialFibrillation[1], ERing[2], and BasicMotions[3]. The description of the three dataset parameters is shown in Table 2. To be compatible with secure computing primitives, we normalize the original datasets and quantify each data point to an integer of 100 or 1000 (depending on the data set). In addition, each experiment is conducted multiple times and the average runtime is reported.

Table 2: The description of the three datasets parameters

Datasets	Parameters		
	n	l	d
AtrialFibrillation	15	640	2
ERing	30	65	4
BasicMotions	40	100	6

We use the three datasets to evaluate the experimental results of local data outsourcing, filtering and classification stage, and query result recovery stage. Meanwhile, the computational cost of our scheme is related to the length of time series in the database, the dimensions of the time series in the database and the filtering rate. To illustrate the effect of these parameters on the running time and communication cost, we use artificial data to evaluate the costs by varying with these parameters.

7.2 Data Outsourcing Stage

In our scheme, this stage is mainly data preprocessing by service provider and users. According to the Section 5.2, the time and computation complexity of data processing are negligible. The service provider is primarily responsible for encrypting and outsourcing its local data to two non-collusion servers. We evaluate the running times of outsourcing three datasets to the servers and find that the computational cost was related to one parameter, i.e., the total number of values contained in database $N = n \times l \times d$. In this experiment, the parameters of the three datasets are respectively 19200, 7800, and 24000 and the corresponding computational time consumption of three datasets are 6066 μs, 3482 μs, and 9203 μs, respectively. We can find that the computational cost of local data outsourcing increases with the increase of N.

7.3 Filtering and Classification Stages

We test the secure filtering and classification phases by implementing three protocols, e.g., SFP, SDTW_DP and SCFP and running protocols on the three datasets, respectively. Specially, we evaluate the total computational cost of P_1 and P_2 and communication overhead between the two servers. We also set the filter rate η to be 73.3%. The detail performance is shown in Table 3. It is easy to know that the heaviest cost in the filtering and classification stages is the SDTW_DP. Therefore, the method we designed to filter out suitable data sequence by leveraging SFP is very effective.

7.4 Result Recovery Stage

The classification result is sent by two servers in arithmetic sharing. The user only needs to reconstruct the secret value $\langle result \rangle_b$ into plaintext locally. This operation is extremely simple and efficient, thus the cost of this stage is negligible.

7.5 The Effect of Different Parameters

Next we evaluate the performance of our scheme under different parameters.

7.5.1 The Time and Communication Varying With η

Table 3 shows that the computation of SDTW_DP is heavier than other processes. The secure filtering protocol can significantly improve the efficiency of the scheme by adjusting the filtering rate η. Thus, we next evaluate the time and communication costs of SDTW_DP under different values of η (η from 80% to 0%), as shown in Figure 2 and Figure 3.

From the two figures, we can see that the computational and communication overheads of SDTW_DP increase as η decreases. The reason is that a larger η leads to a smaller candidate series set, thereby reducing the cost of SDTW_DP. As shown in Fig. 2 and Fig. 3, both the computational and communication overheads on the ERing dataset increase substantially as η decreases from 80% to 20%, demonstrating the effectiveness of the filtering strategy in reducing resource consumption.

At the same time, to validate the efficiency of our filtration strategy, we give the computation time and communication overheads of SDTW_DP without the filtration strategy, i.e., $\eta = 0$. The experimental results show that our filtration strategy is efficient. For example, when $\eta = 80\%$, the computational and communication overheads on AtrialFibrillation are 116.4s and 229.8MB, respectively, while those without filtration are 584.1s and 1149.1MB, respectively. That is, the filtration strategy reduces the cost by approximately 80%.

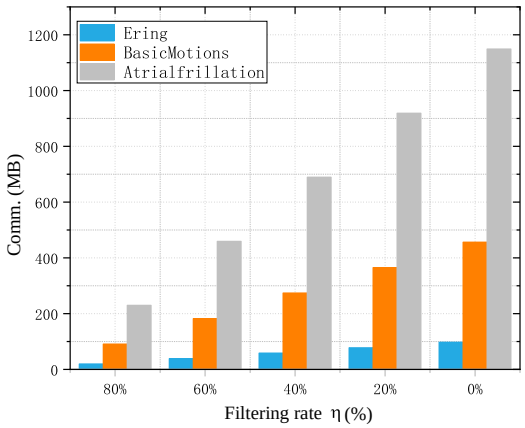

Figure 2: The communication cost of SDTW_DP under different η

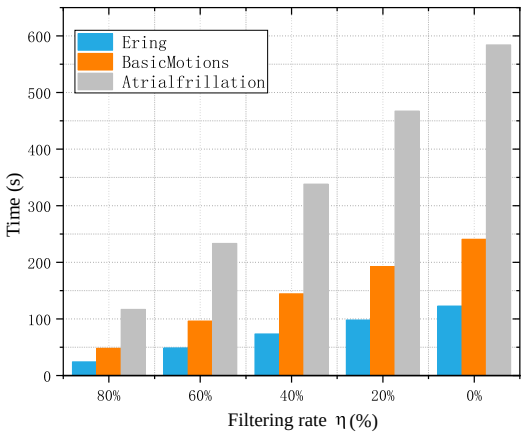

Figure 3: The running time of SDTW_DP under different η

7.5.2 The Time and Communication Varying with d and l

As shown in Subsections 5.2 and 5.3, the dimension d and length l of time series sequence have an impact on the computational and communication costs of LB_{mv} and DTW_D computations in the filtering and classification stage. Thus, we evaluate how d and l affect the costs of these two stages. Specifically, in Figure 4 and Figure 5, we plot the computation time and communication cost of these two stages

Table 3: Performances of our scheme on three data sets

	SFP	SDTW_DP	SCFP
AtrialFibrillation	1.4s/4.01MB	124.6s/306.4MB	0.053s/1.93MB
Ering	0.99s/1.49MB	13.42s/13.05MB	0.12s/4.51MB
BasicMotions	2.31s/6.64MB	19.25s/45.7MB	0.14s/5.15MB

varying with d and l, and set $d = \{1,5,10,15,20\}$, $n = 100$, $l = \{100,200,300,400\}$ and $\eta = 90\%$.

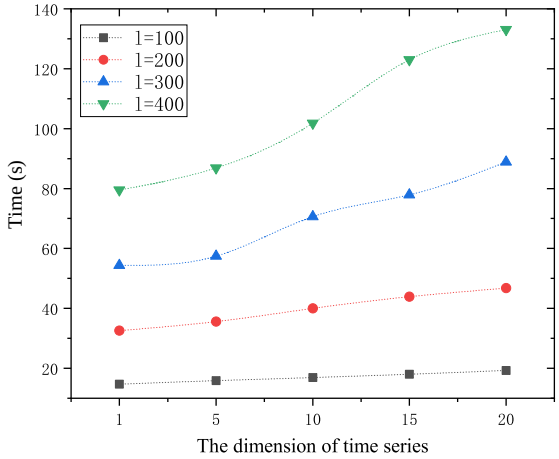

Figure 4: The computation cost of the filtering and classification stage under different d and l

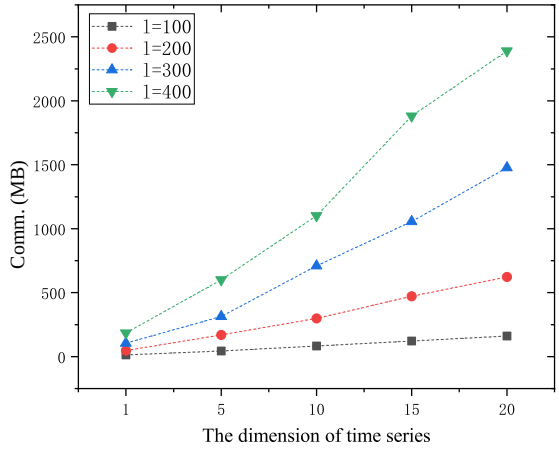

Figure 5: The communication cost of the filtering and classification stage under different d and l

From the two figures, we can see that the computation time and communication cost of filtering and classification stage increase with d and l. Moreover, the increasing of d affects the computation overhead to a lower degree than l. For example, when $l = 200$, the running times of the filtering and classification stages are 35.36s and 46.81s while $d = 5$ and $d = 20$, respectively. Therefore, the dimension d increases by 4 times, but the overall computation time only increases by 1.32 times. On the other hand, when $d = 5$, the running times are 15.85s and 86.86s while $l = 100$ and $l = 400$, respectively. The length l increases by 4 times, but the overall computation time increases by 5.4 times. This is reasonable because the length of the data directly affects the size of the matrix in the $SDTW_DP$. For example, when $l = 10$, the matrix should compute $100 = 10 \times 10$ values. When $l = 100$, the calculation of matrix is $10000 = 100 \times 100$. The amount of the calculated data grows exponentially according to the length. The dimension d also affects the calculation of the first matrix in the DTW_D phase. But it does not affect the size of the matrix, and only affects the number of Euclidean distances which related to d in a linear manner. For example, if $d = 5$ and $l = 10$, 5×10 values should be calculated, and if $d = 10$ and $l = 10$, 10×10 values is calculated.

7.6 Scalability Analysis on a Large-Scale Healthcare Dataset

To further analyze the scalability of SecMTSC in large-scale healthcare time-series scenarios, we conduct a scalability analysis based on the publicly available PhysioNet Challenge 2012 dataset [45]. We select 2,000 records from the training data with publicly available outcome labels for analysis. Each record contains 48 time points and 33 physiological variables, resulting in a dataset size of N=2000, l=48, and d=33.

Since BasicMotions is also a MTSC dataset, we select it as the reference dataset for complexity estimation. Compared with BasicMotions, the PhysioNet subset increases the data scale by approximately 132 times in terms of Nld. Based on the complexity analysis of SecMTSC and the runtime results on BasicMotions, we further estimate the computational overhead in the large-scale setting.

In the estimation, we assume the same filtering rate as the original experiments (η=73.3%) and compare SecMTSC with an unfiltered scheme. The estimated execution times of SFP, SDTWDP, and SCFP in SecMTSC are approximately 304.9 s, 962.5 s, and 7.0 s, respectively, with a total estimated computation time of approximately 1274.4 s. In comparison, the estimated execution times of SDTWDP and SCFP in the unfiltered scheme are approximately 3605.6 s and 26.2 s, respectively, resulting in a total estimated computation time of approximately 3631.8 s. These results indicate that the candidate filtering mechanism can reduce the subsequent secure computation overhead in large-scale healthcare time-series scenarios.

8 Conclusion

This paper proposes SecMTSC, an efficient and privacy-preserving scheme for outsourcing multivariate time series classification based on secure two-party computation. Specifically, we present a series of secure protocols to achieve this goal, such as secure DTW_D distance protocol (SDTW_DP), secure classification protocol and so on. In order to improve the efficiency of the scheme, we additionally designs a secure filtering protocol (SFP) to filter the candidate sequences. The security proofs and the a large number of performance experiments are conducted and the results show that our scheme is secure and has remarkable efficiency and good performance.

Funding

This work was supported by GuangDong Basic and Applied Basic Research Foundation (No. 2024A1515011341), Dongguan Social Development Technology Project (No. 20231800940342).

Author Contributions

Conceptualization, J.W., M.S., and M.C.; methodology, J.W. and M.S.; software, J.W.; validation, J.W., M.S., and J.-N.L.; formal analysis, J.W. and M.S.; investigation, J.W. and M.S.; resources, M.C. and J.-N.L.; data curation, J.W.; writing—original draft preparation, J.W. and M.S.; writing—review and editing, J.W., M.S., M.C., and J.-N.L.; visualization, J.W.;

supervision, M.C. and J.-N.L.; project administration, M.C. and J.-N.L. All authors have read and agreed to the published version of the manuscript.

Conflict of Interest

All the authors declare that they have no conflict of interest.

Data Available

The datasets used in this study are publicly available from the UEA Multivariate Time Series Classification Archive and the PhysioNet/Computing in Cardiology Challenge 2012.

References

[1] Gupta, A., Gupta, H.P., Biswas, B., Dutta, T.: A Fault-Tolerant Early Classification Approach for Human Activities Using Multivariate Time Series. IEEE Transactions on Mobile Computing **20**(5), 1747–1760 (2021) https://doi.org/10.1109/TMC.2020.2973616

[2] Canizo, M., Triguero, I., Conde, A., Onieva, E.: Multi-head CNN–RNN for multi-time series anomaly detection: An industrial case study. Neurocomputing **363**, 246–260 (2019) https://doi.org/10.1016/j.neucom.2019.07.034

[3] León-López, K.M., Mouret, F., Arguello, H., Tourneret, J.-Y.: Anomaly Detection and Classification in Multispectral Time Series Based on Hidden Markov Models. IEEE Transactions on Geoscience and Remote Sensing **60**, 1–11 (2022) https://doi.org/10.1109/TGRS.2021.3101127

[4] Laube, R., Hamilton, H.J.: Wildfire Occurrence Prediction Using Time Series Classification: A Comparative Study. In 2021 IEEE International Conference on Big Data (Big Data), pp. 4178–4182 (2021). https://doi.org/10.1109/BigData52589.2021.9671680

[5] Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., Keogh, E.: Searching and mining trillions of time series subsequences under dynamic time warping. In Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 262–270 (2012). https://doi.org/10.1145/2339530.2339576

[6] Begum, N., Ulanova, L., Wang, J., Keogh, E.J.: Accelerating Dynamic Time Warping Clustering with a Novel Admissible Pruning Strategy. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 49–58 (2015). https://doi.org/10.1145/2783258.2783286

[7] Hamza, R., Yan, Z., Muhammad, K., Bellavista, P., Titouna, F.: A privacy-preserving cryptosystem for IoT E-healthcare. Information Sciences **527**, 493–510 (2020) https://doi.org/10.1016/j.ins.2019.01.070

[8] Deebak, B.D., Al-Turjman, F., Aloqaily, M., Alfandi, O.: An Authentic-Based Privacy Preservation Protocol for Smart e-Healthcare Systems in IoT. IEEE Access **7**, 135632–135649 (2019) https://doi.org/10.1109/ACCESS.2019.2941575

[9] Al-Turjman, F., Zahmatkesh, H., Mostarda, L.: Quantifying Uncertainty in Internet of Medical Things and Big-Data Services Using Intelligence and Deep Learning. IEEE Access **7**, 115749–115759 (2019) https://doi.org/10.1109/ACCESS.2019.2931637

[10] Saha, R., Kumar, G., Rai, M.K., Thomas, R., Lim, S.-J.: Privacy Ensured *e*-Healthcare for Fog-Enhanced IoT Based Applications. IEEE Access **7**, 44536–44543 (2019) https://doi.org/10.1109/ACCESS.2019.2908664

[11] Liu, X., Zheng, Y., Yi, X., Nepal, S.: Privacy-Preserving Collaborative Analytics on Medical Time Series Data. IEEE Transactions on Dependable and Secure Computing **19**(3), 1687–1702 (2022) https://doi.org/10.1109/TDSC.2020.3035592

[12] Zheng, Y., Lu, R., Mamun, M.: Privacy-Preserving Computation Offloading for Time-Series Activities Classification in eHealthcare. In ICC 2020 - 2020 IEEE International Conference on Communications (ICC), pp. 1–6 (2020). https://doi.org/10.1109/ICC40277.2020.9148875

[13] Mercier, D., Lucieri, A., Munir, M., Dengel, A., Ahmed, S.: Evaluating Privacy-Preserving Machine Learning in Critical Infrastructures: A Case Study on Time-Series Classification. IEEE Transactions on Industrial Informatics **18**(11), 7834–7842 (2022) https://doi.org/10.1109/TII.2021.3124476

[14] Tang, Y., Wei, Y., Li, T., Zheng, X., Ji, C.: A hierarchical transformer-based network for multivariate time series classification. Information Systems **132**, 102536 (2025) https://doi.org/10.1016/j.is.2025.102536

[15] Nuyts, L., Perini, L., Davis, J.: TSelect: selecting relevant and non-redundant channels for multivariate time series classification. Data Mining and Knowledge Discovery **39**(6), 76 (2025) https://doi.org/10.1007/s10618-025-01132-4

[16] Oh, Y., Kim, H., Kim, S.: TSSI: Time Series as Screenshot Images for multivariate time series classification using convolutional neural networks. Computers & Industrial Engineering **209**, 111393 (2025) https://doi.org/10.1016/j.cie.2025.111393

[17] Dempster, A., Petitjean, F., Webb, G.I.: ROCKET: exceptionally fast and accurate time series classification using random convolutional kernels. Data Mining and Knowledge Discovery **34**(5), 1454–1495 (2020) https://doi.org/10.1007/S10618-020-00701-Z

[18] Fawaz, H.I., Lucas, B., Forestier, G., Pelletier, C., Schmidt, D.F., Weber, J., Webb, G.I., Idoumghar, L., Muller, P., Petitjean, F.: InceptionTime: Finding AlexNet for time series classification. Data Mining and Knowledge Discovery **34**(6), 1936–1962 (2020) https://doi.org/10.1007/S10618-020-00710-Y

[19] Bagnall, A.J., Flynn, M., Large, J., Lines, J., Middlehurst, M.: A tale of two toolkits, report the third: on the usage and performance of HIVE-COTE v1.0. arXiv preprint arXiv:2004.06069 (2020) https://doi.org/10.48550/arXiv.2004.06069

[20] Bagnall, A.J., Lines, J., Bostrom, A., Large, J., Keogh, E.J.: 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) https://doi.org/10.1007/S10618-016-0483-9

[21] Ruiz, A.P., Flynn, M., Large, J., Middlehurst, M., Bagnall, A.J.: The great multivariate time series classification bake off: a review and experimental evaluation of recent algorithmic advances. Data Mining and Knowledge Discovery **35**(2), 401–449 (2021) https://doi.org/10.1007/S10618-020-00727-3

[22] Ircio, J., Lojo, A., Mori, U., Lozano, J.A.: Mutual information based feature subset selection in multivariate time series classification. Pattern Recognition **108**, 107525 (2020) https://doi.org/10.1016/J.PATCOG.2020.107525

[23] Veizaga, M., Delpha, C., Diallo, D., Bercu, S., Bertin, L.: Voltage Sag Source Classification using Multivariate Time Series and Soft Dynamic Time Warping. In IECON 2022 – 48th Annual Conference of the IEEE Industrial Electronics Society, pp. 1–6 (2022). https://doi.org/10.1109/IECON49645.2022.9968620

[24] Shokoohi-Yekta, M., Hu, B., Jin, H., Wang, J., Keogh, E.J.: Generalizing DTW to the multi-dimensional case requires an adaptive approach. Data Mining and Knowledge Discovery **31**(1), 1–31 (2017) https://doi.org/10.1007/S10618-016-0455-0

[25] Chambon, S., Galtier, M.N., Arnal, P.J., Wainrib, G., Gramfort, A.: A Deep Learning Architecture for Temporal Sleep Stage Classification Using Multivariate and Multimodal Time Series. IEEE Transactions on Neural Systems and Rehabilitation Engineering **26**(4), 758–769 (2018) https://doi.org/10.1109/TNSRE.2018.2813138

[26] Liu, C.-L., Hsaio, W.-H., Tu, Y.-C.: Time Series Classification With Multivariate Convolutional Neural Network. IEEE Transactions on Industrial Electronics **66**(6), 4788–4797 (2019) https://doi.org/10.1109/TIE.2018.2864702

[27] Wilson, D.H., Atkeson, C.G.: Simultaneous Tracking and Activity Recognition (STAR) Using Many Anonymous, Binary Sensors. In International Conference on Pervasive Computing, pp. 62–79 (2005). https://doi.org/10.1007/11428572_5

[28] Zhang, X., Gao, Y., Lin, J., Lu, C.-T.: Tapnet: Multivariate time series classification with attentional prototypical network. In Proceedings of the AAAI Conference on Artificial Intelligence, pp. 6845–6852 (2020). https://doi.org/10.1609/aaai.v34i04.6165

[29] Li, G., Choi, B., Xu, J., Bhowmick, S.S., Chun, K.-P., Wong, G.L.-H.: Shapenet: A shapelet-neural network approach for multivariate time series classification. In Proceedings of the AAAI Conference on Artificial Intelligence, pp. 8375–8383 (2021). https://doi.org/10.1609/aaai.v35i9.17018

[30] Fauvel, K., Lin, T., Masson, V., Fromont, É., Termier, A.: Xcm: An explainable convolutional neural network for multivariate time series classification. Mathematics **9**(23), 3137 (2021) https://doi.org/10.3390/math9233137

[31] Du, M., Wei, Y., Tang, Y., Zheng, X., Wei, S., Ji, C.: ST-Tree with interpretability for multivariate time series classification. Neural Networks **183**, 106951 (2025) https://doi.org/10.1016/j.neunet.2024.106951

[32] Liu, C., Wei, Z., Zhou, L., Shao, Y.: Multidimensional time series classification with multiple attention mechanism. Complex & Intelligent Systems **11**(1), 14 (2025) https://doi.org/10.1007/s40747-024-01630-w

[33] Zheng, Y., Lu, R., Guan, Y., Shao, J., Zhu, H.: Efficient and Privacy-Preserving Similarity Range Query Over Encrypted Time Series Data. IEEE Transactions on Dependable and Secure Computing **19**(4), 2501–2516 (2022) https://doi.org/10.1109/TDSC.2021.3061611

[34] Wang, F., Zhu, H., He, G., Lu, R., Zheng, Y., Li, H.: Efficient and Privacy-Preserving Arbitrary Polygon Range Query Scheme Over Dynamic and Time-Series Location Data. IEEE Transactions on Information Forensics and Security **18**, 3414–3429 (2023) https://doi.org/10.1109/TIFS.2023.3282133

[35] Shi, E., Chan, T.-H., Rieffel, E.G., Chow, R., Song, D.: Privacy-Preserving Aggregation of Time-Series Data. In Proceedings of the Network and Distributed System Security Symposium (2011)

[36] Guan, Y., Lu, R., Zheng, Y., Shao, J., Wei, G.: Achieving Efficient and Privacy-Preserving Max Aggregation Query for Time-Series Data. In 2020 IEEE International Conference on Communications, pp. 1–6 (2020). https://doi.org/10.1109/ICC40277.2020.9149345

[37] Xu, C., Yin, R., Zhu, L., Zhang, C., Zhang, C., Chen, Y., Sharif, K.: Privacy-Preserving and Fault-Tolerant Aggregation of Time-Series Data With a

Semi-Trusted Authority. IEEE Internet of Things Journal **9**(14), 12231–12240 (2022) https://doi.org/10.1109/JIOT.2021.3135049

[38] Rath, T.M., Manmatha, R.: Lower-bounding of dynamic time warping distances for multivariate time series. University of Massachusetts Amherst Technical Report MM **40**, 1–4 (2002)

[39] Shamir, A.: How to Share a Secret. Communications of the ACM **22**(11), 612–613 (1979) https://doi.org/10.1145/359168.359176

[40] Rabin, M.O.: How To Exchange Secrets with Oblivious Transfer. Cryptology ePrint Archive, Paper 2005/187 (2005)

[41] Yao, A.C.-C.: How to generate and exchange secrets. In 27th Annual Symposium on Foundations of Computer Science (sfcs 1986), pp. 162–167 (1986). https://doi.org/10.1109/SFCS.1986.25

[42] Rathee, D., Rathee, M., Kumar, N., Chandran, N., Gupta, D., Rastogi, A., Sharma, R.: CrypTFlow2: Practical 2-Party Secure Inference. In Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security, pp. 325–342 (2020). https://doi.org/10.1145/3372297.3417274

[43] Demmler, D., Schneider, T., Zohner, M.: ABY - A Framework for Efficient Mixed-Protocol Secure Two-Party Computation. In 22nd Annual Network and Distributed System Security Symposium, pp. 1–15 (2015)

[44] Goldreich, O.: Foundations of cryptography: volume 2, basic applications. Cambridge University Press, New York, USA (2009)

[45] Silva, I., Moody, G., Scott, D.J., Celi, L.A., Mark, R.G.: Predicting in-hospital mortality of ICU patients: The PhysioNet/Computing in cardiology challenge 2012. In 2012 Computing in Cardiology, pp. 245–248 (2012)