applied sciences

MDPI

Article

C-MWCAR: Classification Based on Multiple Weighted Class Association Rules

Gui Li [1], Fan Liu [1,2,*], Cheng Wu [1,3], Yuan Yao [1], Guangxin Wu [1], Zhu Wang [2] and Yanchun Zhang [4,5]

[1] Nanjing Research Institute of Electronics Technology, Nanjing 210039, China
[2] School of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China
[3] School of Electronic Engineering, Xidian Univeristy, Xi'an 710126, China
[4] Cyberspace Institute of Advanced Technology, Guangzhou University, Guangzhou 510006, China
[5] New Cyber Research Department, Pengcheng Laboratory, Shenzhen 518055, China
* Correspondence: liufant800@mail.nwpu.edu.cn

Abstract: Classification is a very important task in data mining and pattern analysis, which have been widely used to solve various real-world problems. To obtain better classification performance, in this paper, we propose a novel classification framework based on multiple weighted class association rules (C-MWCAR), whose key idea is to transform the association among features into a set of class association rules (CARs), then classify unknown instances based on the CARs obtained. Concretely, C-MWCAR consists of a dictionary order-based CAR mining algorithm (DOCMA), a branch-based CAR selection algorithm (BCSA), and a multiple weighted CARs-based classifier (MWCC). Specifically, DOCMA mines the complete set of CARs, from which BCSA further selects a representative and concise set of CARs based on the distribution, coverage, and redundancy of the mined CARs. When classifying an unknown instance, MWCC picks out a set of CARs that are most similar to the given instance and computes the weighted importance of those CARs. Finally, the class label of the given instance will be determined by the similarities between the instance and the CARs and the weighted importance of the CARs. Furthermore, we apply the proposed C-MWCAR to a real-world classification task, i.e., hypertension diagnosis, based on a real dataset of 128 subjects. Experimental results indicate that C-MWCAR outperforms four baseline methods and achieves 93.3%, 93.8%, and 92.7% in terms of accuracy, sensitivity, and specificity, respectively.

Keywords: classification; class association rule; interpretable classifier; hypertension diagnosis

Citation: Li, G.; Liu, F.; Wu, C.; Yao, Y.; Wu, G.; Wang, Z.; Zhang, Y. C-MWCAR: Classification Based on Multiple Weighted Class Association Rules. *Appl. Sci.* **2023**, *13*, 8082. https://doi.org/10.3390/app13148082

Academic Editor: Jose Machado

Received: 7 May 2023
Revised: 11 June 2023
Accepted: 5 July 2023
Published: 11 July 2023

1. Introduction

As an important technique in data mining and pattern analysis, classification has been widely applied to varieties of practical scenarios such as disease prediction [1], anomaly detection [2], object recognition [3], etc. In past years, many kinds of classifiers were designed to solve different application problems. For example, Wu et al. [4] utilized Naïve Bayes to model the channel state information (CSI) of a Wi-Fi signal, so as to classify different indoor locations. Paul et al. [5] designed a decision tree-based method to classify different cognitive tasks based on a wireless electrocardiogram (ECG) signal. Liu et al. [6] proposed a hybrid neural network classification framework to differentiate different kinds of arrhythmias. However, despite these methods obtaining good performance, there still are some shortcomings. Concretely, these classifiers usually use heuristic or greedy strategies to build classification rules, which makes them vulnerable to data noise and hence results in limited performance. In addition, some of the classifiers (such as Decision Tree) assume that the extracted features are independent of each other, which obviously ignores the inherent association relationship among features. Therefore, they are not capable of discovering all the interesting and useful rules from a dataset. As for neural network-based classifiers, they can obtain very high accuracy, but their classification results are hard to understand.

To cope with the above problems, a new type of classifiers, called associative classifiers, have been recently proposed, and they include CBA [7], CMAR [8], and so on. Their core idea is to exploit the association relationship among features for classification. In particular, the main steps of the simplest associative classifier are as follows. It first transforms the association relationship among features into a set of class association rules (CARs). A CAR is made up of two parts, i.e., antecedent and consequent. Specifically, the antecedent is comprised of discretized features, while the consequent is a class attribute. Afterwards, it selects a subset of CARs and a default class to build the final classifier. When classifying an instance, it compares the instance with the selected CARs, one by one, and the instance will be classified as the consequent of the CAR that first matches the instance, or the default class if no matching CARs exists. In particular, it has been demonstrated that CARs-based classifiers are usually more accurate than traditional classifiers such as C4.5, Hidden Markov Model (HMM), Inductive Learning Algorithm (ILA), and Naïve Bayes, both theoretically [9,10] and experimentally [7,11]. Moreover, benefiting from the obtained informative CARs, the classification results can be interpreted to some extent.

However, it is not a trivial task when applying associative classifiers to real-world classification tasks for the following reasons. First, extracting a complete set of CARs is very time-consuming [8,12,13], especially when the number of features is high. Second, as the number of extracted CARs is high [7–9,11,14–17], how to select the best subset of CARs to build the most robust and accurate classifier is a real challenge. Third, it is difficult to utilize just one CAR to accurately classify future instances [8]. Moreover, due to the complexity of real-world problems, it is likely that future instances may be unable to match any CARs. In other words, the future instances cannot be assigned a proper class label.

To tackle the problems facing CARs-based classifiers, this study proposes a novel classification framework based on multiple weighted class association rules (C-MWCAR). Its core idea is to classify an instance by selecting a set of CARs that are most similar to the instance and then utilizes their weighted classification abilities to obtain the final classification results. C-MWCAR mainly contains three innovative components, respectively named as dictionary order-based CAR mining algorithm (DOCMA), branch-based CAR selection algorithm (BCSA), and multiple weighted CARs-based classifier (MWCC). Concretely, DOCMA aims to extract the complete set of CARs as fast as possible; BCSA is used to select a representative and concise set of CARs from the extracted CARs based on their distribution, coverage, and redundancy; then, MWCC carefully picks out a few CARs from the obtained concise set of CARs for each unknown instance, and finally uses them to classify each unknown instance as hypertension or normotension. To evaluate the effectiveness of the proposed C-MWCAR, it is applied to a real-world and important classification task, i.e., hypertension diagnosis. The reason why this task is chosen for experiments is because hypertension is a typical cardiovascular disease and affects more than 40% of adults in the world, according to a report from the World Health Organization [18].

The rest of this paper is organized as follows. The background and related literature are reviewed in Section 2. The materials and methods are then described in Section 3. The process of applying the C-MWCAR to the hypertension diagnosis task is described in Section 4, followed by the evaluation of the experimental results in Section 5. Next, the advantages and limitations of the proposed method are discussed in Section 6. Finally, the contribution of this study is concluded in Section 7.

2. Background

This section will briefly introduce the background and the literature relevant to hypertension diagnosis and classifier construction.

Hypertension diagnosis is an important task in the real world. Many studies diagnose hypertension by extracting features from an ECG signal and directly inputting them into a common machine learning algorithm, such as SVM, logistic regression, Decision Tree, and so on. For example, Poddar et al. [19] extracted many HRV-related features from RR intervals sequence based on ECG signal and then fed them into a SVM to identify

hypertensive subjects. Similarly, Ni et al. [20] employed a logistic regression algorithm to discriminate hypertensive and normotensive. The performance of these methods are good; however, the outputted classification results are unable to interpret why the subjects suffer from hypertension, which is unfavorable for doctors to analyze the patient's condition in depth. To solve this issue, Melillo et al. [21] employed a decision tree-based classifier to model the HRV-related features and obtained a set of dendroid decision rules. If a subject matches a rule antecedent, he or she will be classified by the corresponding rule consequent, i.e., the class label. These rules can be used to explain why subjects are hypertensive to some extent. However, the method can only consider just one feature at each tree node when constructing the decision tree, which may lose the useful correlation relationship among features and hence lead to limited classification performance. Hence, it is important to design a novel kind of classifier that can not only diagnose hypertension accurately but can also generate informative classification results to interpret or reflect the subject's physiological state.

The CARs-based classifier is viewed as a very promising technique, and has been successfully applied in many fields, such as identifying heart disease diagnosis [16,17], activity recognition [9], predicting pandemic influenza [14], learning business rules [11], and discriminating protein–protein interaction type [15], etc. The CARs-based classifier considers all the features at the same time and extracts meaningful association rules from them, which significantly prompts the classification performance. Additionally, it can also generate a set of informative CARs that can accurately reflect the subject's physiological state. The first CARs-based classifier is called CBA (classification based on associations) [7] and its construction process is comprised of three steps: mining CARs, selecting a powerful subset of CARs, and building a classifier based on the selected CARs. After that, many follow-up studies were proposed [8,10,12,13], but their main steps remain unchanged. In particular, there are some issues with each step when applying CARs-based classifiers in real-world problems. In the first step, the mining of CARs is usually time-consuming. For example, CBA [7] extracts CARs based on the Apriori algorithm, which needs to iterate the dataset many times and hence results in a huge time overhead. To solve this issue, many new methods have been proposed, including methods based on FP-growth and the ER-tree [8], vertical dataset layout [22], equivalence class rule tree [23], ECR-tree with Obidset [12], lattice [24], and multi-core processor architecture [18]. In the second step, how to select an effective subset of CARs is a real challenge. Most methods select CARs by considering the coverage of the dataset [7–9,11,14–17]. For example, Liu et al. [7] selects CARs that can accurately match at least one instance. Similarly, Li et al. [8] designs a coverage threshold δ to make sure each instance in the training set can be covered by δ rules, aiming to increase the number of selected CARs. Part et al. [15] takes both the significance of CARs and overlapping rules into consideration. However, these CARs' selection methods ignore the redundancy and distribution of the CARs. In the third step, using only one CAR to classify an instance is hard to obtain high performance [7]. Moreover, It Is likely that there may be no matching CARs in some instances. To this end, Li et al. [8] proposes a weighted χ^2-based method that classifies instances by integrating multiple CARs. CAEP [25] selects a set of emerging patterns and fuses their classification power to assign a possibility for each class, where the class with the largest possibility is viewed as the final class label.

To effectively tackle real-world classification tasks such as hypertension diagnosis, this study designs a novel associative classifier named C-MWCAR, which can simultaneously solve the above-mentioned problems. Concretely, it reduces the time overhead of mining CARs by using a dictionary order-based CAR mining algorithm (i.e., DOCMA) to optimize the most time-consuming steps of the Apriori algorithm. Afterwards, it utilizes a branch-based CAR selection algorithm (i.e., BCASA) to guarantee that the selected CARs are more representative and concise. Finally, it classifies instances by building a multiple weighted CARs-based classifier (i.e., MWCC), which can solve the problem of no matching CARs and obtain higher performance at the same time.

3. Materials and Methods

In this section, the basic idea of associative classifiers is introduced and formalized first, then the details of the C-MWCAR are explained comprehensively.

3.1. Formalization of Basic Associative Classifier

In this section, the basic concepts of associative classifiers are formalized. For CARs-based classifiers, the dataset is usually organized as a relational table. Let $D = \{d_1, \ldots, d_i, \ldots, d_N\}$ be the dataset, which contains N instances and each of them is described by $k-1$ features. Let $Y = \{y_1, \ldots, y_i, \ldots, y_M\}$ be a finite set of class labels which contains M known classes. Each instance in D has been labeled with one class of Y. To simplify the narration, we treat the class label as a special feature; thus, each instance has k features. These k features can be discrete or continuous, but they should be adjusted into uniform formats before mining CARs. For a discrete feature, its possible values are directly mapped to consecutive numbers. For a continuous feature, its value range is first discretized into several intervals which are then mapped to consecutive numbers. For example, assume that an instance is characterized by three features (i.e., feature A and B, and class C) and that A and B are discretized into four and three intervals, respectively, while C contains two class labels. In this case, the intervals of A and B and the class labels of C can be orderly mapped to numbers 1 to 4, 1 to 3, and 1 to 2, respectively. Based on the discretization and mappings, an instance can be represented as a set of <*feature, value*> pairs, where the *feature* refers to the feature's name like A, B, or C, while the *value* refers to the corresponding number. In particular, each pair is called an *item*. Let $I = \{item_1, \ldots, item_i, \ldots, item_L\}$ be the set of items derived from the N instances in D, which contains L items. Next, we define *itemSet* as a subset of I where no items derived from the same feature exists. Note that the items in each itemSet follow the same feature order. In particular, an *itemSet* containing k items is called a *k-itemSet*. Furthermore, if a k-itemSet ($k \geq 2$) has one class-derived item, we call it a candidate class association rule (CCAR), which is expressed as follows:

$$ruleItems \rightarrow y \tag{1}$$

where y, i.e., rule consequent, represents the class-derived item, while *ruleItems*, i.e., rule antecedent, denotes the rest items of the *k-itemSet*. A CCAR with *support (sup) s* means that $s\%$ of the instances in D contain the *ruleItems* and are labeled with y. A CCAR has *confidence (conf) c* means that $c\%$ of instances in D that contain the *ruleItems* are labeled with y. Let *Rcount* and *Ccount* represent the number of instances in D that contain the *ruleItems*, and the number of instances in D that contain the *ruleItems* and are labeled with y, respectively. Thus, the *sup* and *conf* of a given CCAR can be computed as follows:

$$sup = (CCount/|D|) \times 100\%, \tag{2}$$

$$conf = (CCount/RCount) \times 100\%, \tag{3}$$

where $|D|$ is the number of the instances in D. A CCAR whose *sup* is bigger than a user-specified support (*userSup*) is called a frequent candidate class association rule (FCCAR), otherwise called an infrequent candidate class association rule (ICCAR). Moreover, an FCCAR whose *conf* is bigger than a user-specified confidence (*userConf*) is called a class association rule (CAR).

To facilitate the understanding, a simple example to indicate the relationship among the CCAR, FCCAR, and CAR is provided. Given a CCAR with the form of:

$$\{\langle A, 1\rangle, \langle B, 3\rangle\} \rightarrow \langle C, 5\rangle, \tag{4}$$

where A and B are attributes, and C is the class label. Actually, the $\{\langle A, 1\rangle, \langle B, 3\rangle\}$ and $\langle C, 5\rangle$ are the *ruleItems* and y in Equation (1), respectively. Assume that the *Rcount* and the *Ccount* of the {<A, 1>, <B, 3>} are 3 and 2, respectively, and the number of instances in D

is 10. Thus, the *sup* of the CCAR is 2/10 = 20%, and the confidence is 2/3 = 66.7%. If the *minSup* is specified as 10%, we say this CCAR is a FCCAR since its *sup* is greater than the *minSup*. Furthermore, if the *minConf* is set to less than 66.7%, this CCAR is actually a CAR, and hence can be employed when building the CARs-based classifiers.

The symbols frequently used in this paper are shown in Table 1.

Table 1. Frequently used symbols and their definitions.

Symbols	Definition
$l_i[j]$	The j_{th} item of the i_{th} itemSet
L_k	The set of all the frequent k-itemSets
L_k^i	The i_{th} element of the L_k
C_k	The set of the candidate k-itemSets
l_k^i	The i_{th} element of the set of the C_k
$CCAR_k$	The set made up of all the CCARs that contain k items
$ICCAR_k$	The set made up of all the ICCARs that contain k items
CAR_k	The set made up of all the CARs that contain k items.
RuleSet	The complete set of the extracted CARs
D	The dataset, with D_i representing the i_{th} instance in D
Y	The class label set, with Y_i representing the i_{th} label in Y
I	The set of the items, with I_i representing the i_{th} items in I

3.2. Implementation of the Proposed C-MWCAR

3.2.1. Workflow of C-MWCAR

As shown in Figure 1, C-MWCAR mainly contains three components, i.e., DOCMA, BCSA, and MWCC. DOCMA extracts the complete set of CARs from the discretized features. With the optimized connection strategy, optimized pruning strategy, and pre-pruning strategy, the speed of CAR extraction is promoted significantly. BCSA then divides the CAR set into several branches, from each of which the most representative CARs are selected. Afterwards, BCSA further select a more concise set of CARs from the above-mentioned representative CARs by simultaneously measuring the coverage and redundancy of the CARs. When classifying an unknown instance, the features of the unknown instance are extracted and then formalized in advance by using the method introduced in Section 3.1, then the MWCC selects the most similar CARs compared with the formalized unknown instance from the concise set of CARs for each class (in this step, the similarity of each of the selected CARs to the unknown instance is computed), and further computes the weighted importance of each of the selected CARs. Finally, the class label of the unknown instance is determined by simultaneously considering the similarities and the weighted importance of the selected similar CARs. The details of the implementation of the proposed DOCMA, BCSA, and MWCC are described in the following sections.

3.2.2. Dictionary Order-Based CAR Mining Algorithm (DOCMA)

The Apriori algorithm [26] is usually employed to extract the association relationship among items due to its simplicity and intelligibility. Thus, in this paper, we exploit the Apriori algorithm to mine the CARs, which will traverse the dataset repeatedly. Roughly, in the 1st pass, it picks out all the frequent 1-itemSets. In the (k + 1)th pass, it first generates candidate (k + 1)-itemSets based on the frequent k-itemSets obtained in the k_{th} pass. Next, it picks out all the CCARs from the obtained candidate (k + 1)-itemSets. At the end of the current pass, CARs are picked out from the obtained CCARs based on the values of the *sup* and *conf*. Finally, the CARs extracted from each pass conjointly make up the final CAR set. However, this procedure needs to repeatedly connect frequent k-itemSets into candidate (k + 1)-itemSets and prune ICCARs from CCARs in each pass, which is very time-consuming and severely limits the practicability of the method. To this end, we propose the DOCMA by designing and proving a set of definitions, theorems, and

corollaries, by which the time cost of the connection of frequent k-itemSets and the pruning of ICCARs from CCARs can be significantly reduced.

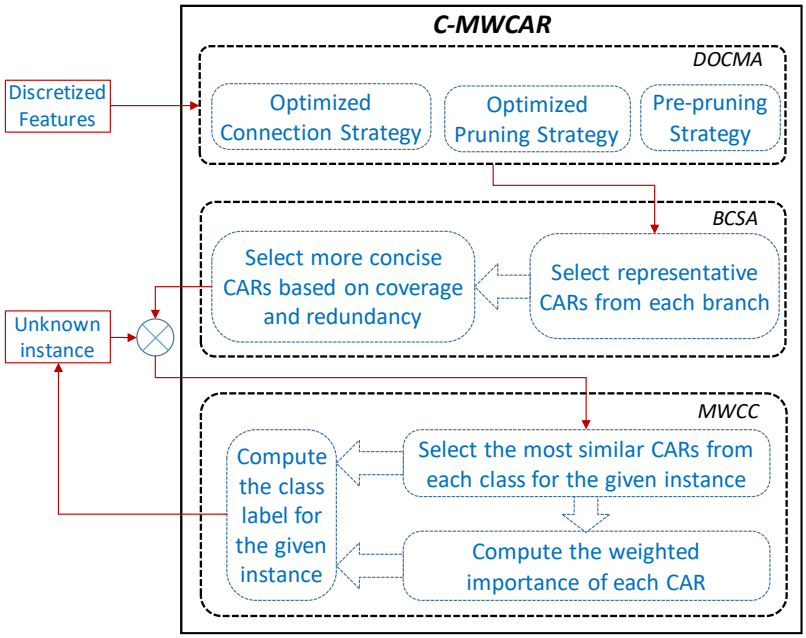

Figure 1. Workflow of the proposed C-MWCAR.

Definition 1. *(The connectivity of two k-itemSets): Given two frequent k-itemSets l_1 and l_2, we say they are connectable only if $(l_1[1] = l_2[1]) \wedge \ldots \wedge (l_1[k-1] = l_2[k-1]) \wedge (l_1[k] \neq l_2[k])$ is valid and $l_1[k]$ and $l_2[k]$ are derived from different features, where $l_i[k]$ is the k_{th} item of the i_{th} frequent k-itemSets. In particular, they produce only one candidate (k + 1)-itemSet, i.e., <l1[1], l2[2], . . . , $l_1[k]$, $l_2[k]$>, as the connection result which is denoted as $l_1 \bowtie l_2$.*

Definition 2. *(The dictionary order of two k-itemSets): Given two k-itemSets l_1 and l_2, they meet the dictionary order with l_1 preceding l_2 if $(l_1[1] = l2[1]) \wedge \ldots \wedge (l_1[I-1] = l_2[I-1]) \wedge (l_1[i] < l_2[i])$, $(1 \leq I \leq k)$ is true, which is denoted as $l_1 < l_2$.*

Theorem 1 . *The k-itemSets in L_k follow the dictionary order.*

Proof. Assume that there are m (k – 1)-itemSets in L_{k-1}. I Apriori algorithm will sequentially judge if each of the binary tuples, i.e., $(l_1l_2), \ldots, (l_1l_m), \ldots, (l_2l_3), \ldots, (l_2l_m), \ldots, (l_{m-1}l_m)$ is connectable and if the corresponding connection result is frequent. Here, mathematical induction is employed to prove Theorem 2. When k = 2, the 1-itemSets in L_1 are already in dictionary order, hence, it is obvious that the above binary tuples are in dictionary order. When k > 2, if the (k – 1)-itemSets in L_{k-1} are already in dictionary order, as only the last items of two connectable (k – 1)-itemSets are different, the final obtained frequent k-itemSets are also in dictionary order. To sum up, Theorem 2 is established. □

Theorem 2. *Given three k-itemSets l_1, l_2, and l_3 $(l_1 < l_2 < l_3)$, if $(l_1[1] = l2[1]) \wedge \ldots \wedge (l_1[I-1] = l_2[I-1]) \wedge (l_1[i] < l_2[i])$, $(1 \leq I \leq k)$ is true, there must exist a j $(1 \leq j \leq i)$ that makes $l_1[j] < l_3[j]$ valid.*

Proof. In the first case, i.e., $l_2[i] \leq l_3[i]$: since $l_1[i] < l_2[i]$, there exists a $j = I$ that makes $l_1[j] < l_3[j]$ valid. In the second case, i.e., $l_2[i] > l_3[i]$: since $l_2 < l_3$, there must exist a j $(1 \leq j \leq i)$ that makes $l_2[j] < l_3[j]$ true. Moreover, considering that $l_1[j] = l_2[j]$ $(1 \leq j \leq i)$, we obtain $l_1[j] < l_3[j]$. To sum up, Theorem 1 is correct. □

Corollary 1. *Given two frequent k-itemSets l_1 and l_2 in L_k, if $l_1 < l_2$ and they are not connectable, then l_1 and any frequent k-itemSet that is behind l_2 are also not connectable.*

Proof. Assume that l_x is a k-itemSet in L_k which is behind l_2, i.e., $l_1 < l_2 < l_x$ is true. Since l_1 and l_2 are not connectable, the largest I making $(l1[1] = l2[1]) \wedge \ldots \wedge (l_1[I-1] = l_2[I-1]) \wedge (l_1[i] < l_2[i])$ true is $k-1$. Meanwhile, in the first $k-1$ items of L_x, there must be a j that makes $l_1[j] < l_x[j]$ true according to Theorem 2. Therefore, l_1 and l_x are not connectable. □

Theorem 3. *In L_k, if the number of frequent k-itemSets that contain a given frequent 1-itemSet is smaller than k, then, the generated candidate (k + 1)-itemSets that contain the frequent 1-itemSet cannot be frequent.*

Proof. According to the characteristics of the Apriori algorithm, we know that for a candidate (k + 1)-itemSet (denoted as l^i_{k+1}), if l^i_{k+1} is frequent and contains a frequent 1-itemSet denoted as l^i_1, then l^i_{k+1} should have k k-subsequences (i.e., k-itemSets) containing l^i_1 and all of them are frequent and in L_k. Therefore, in L_k, if the number of k-itemSets containing a given frequent 1-itemSet is smaller than k, then the candidate (k + 1)-itemSets generated by connecting those k-itemSets cannot be frequent. □

Algorithm 1 shows how the proposed DOCMA works. Based on Corollary 1, the optimized connection strategy is as follows (lines 10–19). When connecting two frequent k-itemSets l_1 and l_2, if l_1 and l_2 are not connectable, then l_1 can give up comparing with all the k-itemSets that are behind l_2. Moreover, the optimized pruning strategy is described as follows (lines 22–30). Given an arbitrary (k – 1)-subsequence of a candidate k-itemSet and a (k – 1)-itemSet in L_{k-1}, denoted as c and l, respectively, if $c < l$, it is impossible for c to be the same as the (k – 1)-itemSets behind l in L_{k-1}. Hence, the candidate k-itemSet cannot be frequent, so that we can prune it now. Based on Theorem 3, we can design a pre-pruning strategy to further reduce the number of connection operations required as follows (*lines 3–9*). We first count the number of k-supersequences (i.e., k-itemSets) of each frequent 1-itemSet in L_k. If the number is smaller than k, we can then delete the corresponding k-itemSets from L_k before connecting frequent k-itemSets, as the generated candidate (k + 1)-itemSets containing the deleted k-itemSets cannot be frequent.

3.2.3. Branch-Based CAR Selection Algorithm (BCSA)

By now, we have obtained a large number of CARs (denoted as *RuleSet*); however, we should not include all of them in the final classifier because they contain much redundant and noisy information. However, we also cannot discard any of the CARs arbitrarily since each has a certain classification ability. Intuitively, for a given CAR, greater confidence and greater support usually mean that the CAR is more reliable and more useful, and hence should be included into the final classifier. Nevertheless, based on extensive experimental results on our dataset, we notice that similar CARs, i.e., CARs that contain many same items, often have similar confidence and support. Therefore, if we directly select CARs based on their confidence and support, the selected CARs are likely to be similar to each other. That is, the selected CARs will belong to the same part of the *RuleSet*, which makes them unrepresentative and redundant. To solve this issue, we propose the BCSA, i.e., Algorithm 2, which consists of two stages. In particular, the first stage is to preliminarily select a set of representative CARs from the different parts of the *RuleSet*, from which the second stage then picks out a smaller but more concise set of CARs.

Algorithm 1: Dictionary order-based CAR mining algorithm (DOCMA)

Input: User-specified support (*userSup*), user-specified confidence (*userConf*).
All the frequent 1-items, i.e., L_1.
Output: The complete set of CARs, which is denoted as *RuleSet*.

1. $RuleSet = \emptyset$; $CCAR_k = \emptyset$; $ICCAR_k = \emptyset$; $CAR_k = \emptyset$; $C_k = \emptyset$; // *initialization*
2. ***for*** (k = 2; $L_{k-1} \neq \emptyset$; k++) ***do***
3. // *delete frequent (k − 1)-itemSets that cannot generate frequent k-itemSets*
4. ***for*** ($I = 1$; $I \leq L_{k-1}$.size(); i++) ***do***
5. *numTemp* = the number of k-supersequence of L_1^i in L_{k-1}
6. ***if*** (numTemp < k − 1) ***then***
7. delete all the k-supersequence of L_1^k from L_{k-1}
8. ***end if***
9. ***end for***
10. // *optimized connection of frequent (k − 1)-itemSets*
11. ***for (I = 1; I < L_{k-1}.size(), i++) do***
12. ***for*** ($j = 2$; $j = L_{k-1}$.size(); j++)***do***
13. ***if*** (L_k^i and L_k^j are connectable) ***then***
14. $C_k = C_k + L_1^k \bowtie L_1^k$
15. ***else***
16. *break;*
17. ***end if***
18. ***end for***
19. ***end for***
20. // select the $CCAR_k$ in current pass
21. $CCAR_k$ = {c ∈ C_k | c has only one item derived from class attribute}
22. // optimized pruning of $ICCAR_k$
23. ***for*** ($I = 1$; $I \leq CCAR_k$.size(); i++) ***do***
24. ***for*** ($j = 1$; $j \leq CCAR_k^i$.(k − 1)-subsequence.size(); j++)***do***
25. ***if*** (the j_{th} (k − 1)-subsequence of $CCAR_k^j$ is not in L_{k-1}) **then**
26. $ICCAR_k = ICCAR_k + CCAR_i$
27. *break;*
28. *end if*
29. *end for*
30. *end for*
31. // remove some *ICCARs* from $CCAR_k$ in advance
32. $FCCAR_k = CCAR_k - ICCAR_k$
33. // *select the CAR_k in current pass based the values of sup and conf*
34. ***for*** ($I = 1$; $i \leq CCAR_k$.size(); i++) ***do***
35. ***if*** ($FCCAR_k^i.sup \geq userSup$ && $CCAR_k^i.conf \geq userConf$) ***then***
36. $CAR_k = CAR_k + FCCAR_k^i$
37. ***end if***
38. ***end for***
39. $RuleSet = RuleSet \cup CARk$
40. ***end for***

In the first stage (lines 1–6), we first divide the *RuleSet* into a set of non-overlapping branches according to the items of each CAR. Here, the branch is defined as a set of CARs that have the same prefixes, i.e., the first few items. In particular, if the first k items of these CARs are same, we say that these CARs are in the same k-branch. In this study, we first divide the *RuleSet* into several k-branches, from each of which we then pick out the CARs that correspond to the maximal confidence or maximal support, respectively, as the representative CARs. In this way, we can guarantee that the selected CARs are made up of the most powerful CARs coming from different parts of the *RuleSet*, so that the selected CARs are of higher diversity and hence are more likely to cover more unknown instances in the future.

Algorithm 2: Branch-based CAR selection algorithm (BCSA)

Input:

- *k:* the number of same prefixes in each branch
- *RuleSet*: the complete set of CARs mined by using DOCMA
- *T*: the dataset.
- The given coverage threshold *C* and the given redundancy threshold *R*.

Output:

- *FinalCARs*: the final CARs set that will be used to build the classifier.

```
1.   branches = RuleSet.divide(k)        // divide RuleSet into k-branches
2.   // select CARs with biggest Conf or biggest Sup from each k-branch
3.     for (i = 1; i ≤ branches.size(); i++) do
4.           rCars = rCars + branches(i).biggestConf()      / /rCars is initially empty
5.           rCars = rCars + branches(i).biggestSup()
6.      end for
7.     sortedCars = sort(rCars)        // sortedCars is sorted representative CARs
8.    FinalCARs = Ø, CV = 0;        // CV is the computed coverage value
9.    for (i = 1; i ≤ sortedCars.size(); i++) do
10.                  Num = 0;
11.                  for (j = 1; j < T.size(); j++) do
12.                       if (T(j) matches sortedCars(i)) then
13.                            Num++;
14.                       end if
15.                  end for
16.                  if (Num > 0 && sortedCars(i).redundancy < R && CV < C) then
17.                     FinalCARs = FinalCARs + sortedCars(i)
18.                  end if
19.                  update(CV); //update the current coverage value
20.                  if (CV ≥ C) then
21.                           break;
22.                  end if
23.        end for
24.        return (FinalCARs);
```

The second stage aims to extract a smaller but more concise set of CARs from the obtained representative CARs for the final classifier. Ideally, the selected CARs should be able to classify most of the instances in the training set. Meanwhile, the redundant information contained in the selected CARs should be as little as possible. To meet these two requirements, we first rank the representative CARs in descending order (line 7). Given two CARs r_i and r_j, r_i is said to have a higher rank than r_j, denoted as $r_i > r_j$, if and only if (1) $r_i.conf > r_j.conf$; (2) $r_i.conf = r_j.conf$ but $r_i.sup > r_j.sup$; or (3) $r_i.conf = r_j.conf$, $r_i.sup = r_j.sup$, but r_i has more items than r_j does. As a result, CARs that have stronger classification ability are sorted in the front. We then design two metrics, i.e., coverage and redundancy, to respectively measure the generalization ability of the CARs that have already been selected into the final CARs set, with the amount of redundant information contained in a given CAR as follows:

$$Coverage = \frac{NC}{ND} \tag{5}$$

$$Redundancy = \frac{NPC}{NCC} \tag{6}$$

where *NC*, *ND*, *NCC*, and *NPC* stand for the number of instances that can be matched by CARs previously selected, the number of instances in the dataset, the number of instances that can be matched by the current CAR, and the number of instances in *NCC* that can be matched by previously selected CARs, respectively. For example, assume that the

dataset contains 20 instances and that there are 3 representative CARs, having been in descending order. Assume that these CARs can respectively match 2, 5, and 6 instances; however, they can only match 6 different instances in total due to redundancy, i.e., 4 of these 6 instances that can be matched by the third CAR have been matched by the first or second CAR. In this case, $NC = 6$, $ND = 20$, $NCC = 6$, and $NPC = 4$; *coverage* = 6/20 = 30%; and *redundancy* = 4/6 = 66.7%. Finally, we sequentially judge whether each CAR in the sorted representative CARs should be included in the final CAR set. As shown in Algorithm 2 (lines 8–24), for a given CAR r_i in the sorted representative CARs, we first go through the dataset to see if r_i can match at least one instance. If so, we then compute the coverage and redundancy. Only when they are respectively smaller than the given coverage threshold C and the given redundancy threshold R will r_i be included in the final CAR set. In this way, the final CAR set used to build the classifier can match as many instances as possible while containing less redundant information.

3.2.4. Multiple Weighted CARs-Based Classifier (MWCC)

When classifying a new instance, most existing CARs-based classifiers usually label it with the consequent (i.e., the class-derived item) of the first CAR in the obtained final CAR set that matches the instance. If there are no matching CARs, it will be assigned a default class [7]. However, the above procedure has two shortcomings when facing actual problems. We find that although the discretized feature values of some new instances cannot match any CARs, they are very similar to some CARs. On this occasion, classifying those instances with the default class will unavoidably lose some useful information, leading to reduced classification performance [8]. Moreover, sometimes one instance can match several CARs; however, these CARs are conflicting CARs. In other words, the class labels of those matching CARs are different from each other. Thus, we will be unable to classify the instance since we do not know which CAR we should use. Here, it takes the diagnosis of hypertension as an example to illustrate the shortcomings. Hypertension is a quite complex physiological status where physiological indexes often change continuously, which makes hypertensive patients usually show much different physiological indexes. Even the same patient's physiological indexes recorded at different times may also be different. Therefore, given a new subject, it is likely that the subject's physiological indexes cannot match any of the CARs obtained already but are similar to some of them. Moreover, hypertensive patients may not show any symptoms at times, which may result in a patient's physiological indexes matching several CARs having conflicting class labels. To tackle these problems, we designed a novel associative classifier named MWCC. Concretely, MWCC first selects a set of CARs that are most similar to the given instance based on the similarities between the CARs and the instance. We then compute the weighted classification abilities of the selected CARs. Finally, we classify the instance by considering the obtained similarities and the weighted classification abilities simultaneously. In this way, the problem of no matching CARs and conflicting CARs can be effectively solved.

To measure the similarity between a CAR and an instance, we design a metric named similarity which is defined as follows:

$$Sim(CAR,\ Instance) = \frac{\sum_{i=1}^{i=N} \frac{(Max_i - Min_i)^2 - (CAR_i - Instance_i)^2}{(Max_i - Min_i)^2}}{N} \tag{7}$$

where N denotes that the given CAR contains N items, CAR_i is the value of the i_{th} item of the CAR, $Instance_i$ is the value of the corresponding item of the instance (that is, CAR_i and $Instance_i$ come from the same feature), and Max_i and Min_i, respectively, denote the maximum and minimum values of the aforementioned item in the dataset. Obviously, the part, i.e., $(Max_i - Min_i)^2$, is fixed when CAR is given, so that the more similar the CAR is to the given instance, the bigger the computed similarity is, and the more likely the instance belongs to the class represented by the CAR. In particular, the denominator, i.e., N, is employed to eliminate the effects caused by different numbers of the items of CARs,

which means that the computed similarity represents the average similarity of each item. For each instance, we select the K most similar CARs from different classes, respectively, for the final classification procedure.

The classification ability of each CAR may differ from each other; hence, we should not treat them equally. Intuitively, if the support of an item changes dramatically from one class to another, the item may have much stronger ability to classify the instances containing this item, and hence should be assigned a higher weight. Given an item, its weight regarding a given class A is marked as $W(item, A)$, which is defined as follows:

$$W(item, A) = freq(item, A) \times \prod_{i=1}^{i=N-1}(1 - freq(item, B_i)) \quad (8)$$

where *freq(item,A)* denotes how frequently the item appears in CARs whose class label is A, N indicates that the dataset contains N classes in total, and B_i represents the i_{th} class except for class A. Obviously, the computed weight tends to be high if the item appears frequently in class A but infrequently in other classes. Based on the weight of an item defined above, the weight of a CAR regarding class A can be defined as follows:

$$W(CAR, A) = \frac{\sum_{i=1}^{i=N} W(item_i, A)}{N} \quad (9)$$

where N denotes the number of items of the given CARs, while $item_i$ represents the i_{th} item of the CAR.

After obtaining the similarities of the K most similar CARs and their weights relating to each class, the final class label of the new instance can be determined as follows:

$$Result = \underset{\omega_j}{argmax}\left(\sum_{i=1}^{i=K} W\left(CAR_i, \omega_j\right) \times Sim\left(CAR_i, \omega_j\right)\right) \quad (10)$$

where ω_j represents the j_{th} class of the dataset and CAR_i is the i_{th} similar CARs regrading to the current class.

4. C-MWCAR Applied to Hypertension Diagnosis

Hypertension is a typical and extremely harmful cardiovascular disease which affects more than 40% of adults in the world, according to a report from the World Health Organization [27]. The traditional clinical method to diagnose hypertension is to measure blood pressure by using a cuff-based mercury sphygmomanometer, which is not suitable for daily use in the home environment. More importantly, hypertension is a quite complex physiological status where blood pressure always fluctuate unpredictably. However, the sphygmomanometer can only give discontinuous measurement results, which is not conducive to accurate analysis of the disease status. Hence, a classifier that can not only produce higher classification performance but can also generate interpretive classification results is suitable for the hypertension diagnosis task. This is the reason why this study specially designs a CARs-based classifier (i.e., C-MWCAR), instead of applying a common machine learning technique such as logistic regression.

This section focus on introducing the process of applying C-MWCAR to hypertension diagnosis, with the experimental setup, results, and evaluation given in Section 5.

4.1. Data Collection and Preprocessing

As a typical cardiovascular disease, in recent studies, hypertension is usually diagnosed by analyzing heart rate variability (HRV) [20,28–32]. HRV is a useful tool to characterize the fluctuation pattern of the heart-beat interval sequence. On the other hand, as a non-intrusive and easily accessible physiological signal, a ballistocardiogram (BCG) signal [33,34] can accurately reflect the micro body vibrations caused by heart beats. Therefore, the BCG signal has been widely used to extract heart-beat interval sequence and further analyze HRV. Hence, this study also utilizes the BCG signal to diagnose hypertension.

In this study, an off-the-shelf BCG acquisition system named RS-611 [35] is employed to collect the BCG signal. As shown in Figure 2, the main components of RS-611 are a micro-movement sensitive mattress (MSM) with two hydraulic pressure sensors placed in the upper part and lower part of the MSM, respectively, an analog-digital (AD) converter, and a terminal PC. When a subject lies on the MSM, the pressure changes caused by heart beats can be recorded in real time, which is called the BCG signal. Compared with other wearable data acquisition devices such as ECG Holter [36] and smart watch [37], RS-611 is completely unobtrusive and the users do not need to attach any electrodes or devices on their body.

Figure 2. The micro-movement sensitive mattress-based BCG signal acquisition system (RS-611).

This study recruited 128 staff members of our university as subjects. First, their blood pressure was measured by a trained nurse using a standard mercury sphygmomanometer. In particular, blood pressure was measured in the left arm twice after five minutes of rest, with the subjects in seated position, according to the guidelines of the American Society of Hypertension [15]. For each subject, blood pressure was recorded in four separate sessions within three weeks, and the averaged values were then used to represent the final systolic and diastolic blood pressure values. Subjects with blood pressure values greater than 140/90 mmHg were viewed as hypertensive patients, while the rest were regarded as normotensive subjects. For each subject, one night of BCG was collected when they were sleeping, since the physiological indexes are relatively stable and not affected by other factors during sleep. The statistical results of the BCG dataset are summarized in Table 2.

Table 2. Statistics of the experimental BCG dataset (mean ± standard deviation).

Subject Information	Hypertensive	Normotensive
Number	61	67
Sex (Male/Female)	33/38	35/32
Age (years	55.6 ± 7.9	53.2 ± 9.2
Heart Rate (bpm)	77.1 ± 9.2	73.6 ± 8.3
Body Mass Index (kg/m^2)	24.3 ± 3.6	23.7 ± 3.3
Systolic blood pressure (mmHg)	155.6 ± 11.2	112.1 ± 15.7
Diastolic Blood Pressure (mmHg)	103.6 ± 8.2	74.4 ± 6.3

A segment of the BCG signal is shown in Figure 3a, from which we extract heart-beat interval sequence through the following steps. First, we normalize the BCG signal by means of the Z-score method [33] in order to eliminate the individual error caused by body weights, since different body weights can modulate the amplitude of the BCG to some extent. Next, the frequency band that reflects the heart beats is extracted by designing an elliptic bandpass filter, so as to remove the noise caused by breathing or other reasons. Based on trial and error, the stopband corner frequency and the passband corner frequency are finally set to 13/6 Hz and 5/6 Hz, respectively. The stopband attenuation and the passband ripple are set to 8 and 0.2, respectively. Figure 3b shows the filtered BCG signal. Finally, an overlapping sliding window is used to automatically detect heart beats. In particular, the window size is set to 100 samples by considering the signal sampling rate (100 Hz) and normal heart rate range (60–100 beats per minute). The detected heart beats are shown in Figure 3c, from which the heart-beat interval sequence is obtained.

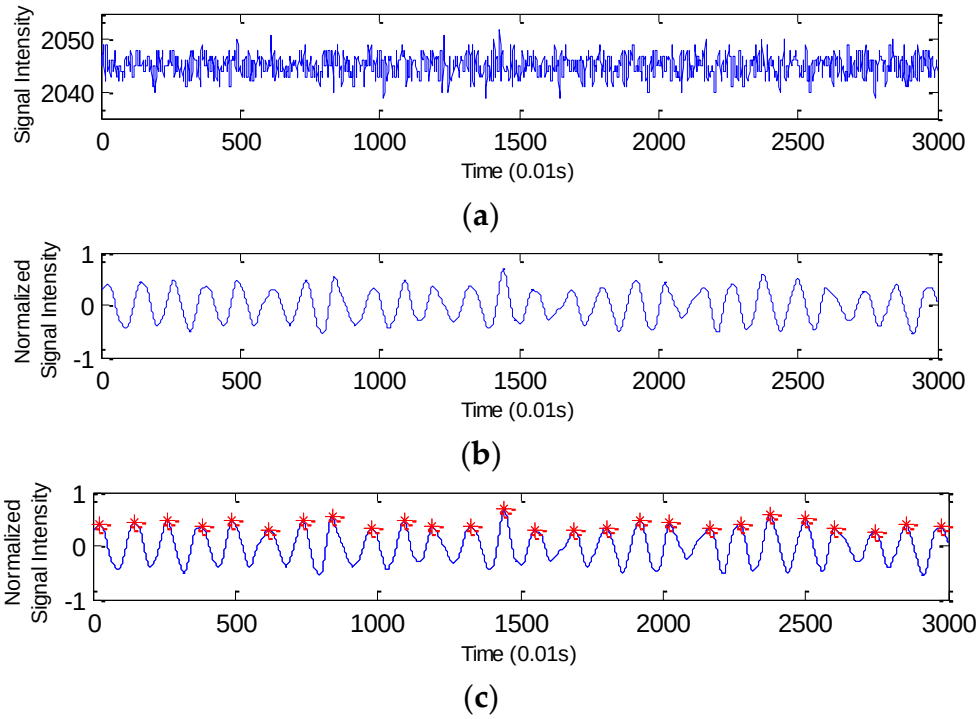

Figure 3. The original BCG, the filtered BCG, and the detected heart beats. (**a**) The original BCG signal; (**b**) The normalized and filtered BCG signal; (**c**) The detected heart beats.

4.2. Feature Extraction and Discretization

In this section, a set of features is first extracted from the obtained heart-beat interval sequence, then the obtained features are discretized and mapped into numbers in order to meet the format for CAR mining.

4.2.1. Feature Extraction

To objectively evaluate the power of the proposed C-MWCAR, commonly used time-frequency domain features and non-linear domain features in the hypertension diagnosis field [20,28–32] are also used in this study, so as to eliminate the effects caused by the features. The definition and description of the extracted features are listed in Table 3 and the statistical analysis results are listed in Table 4.

Table 3. List of the extracted features. The detailed definitions and descriptions of these features can be found in [20,28–32].

Type	Features	Definition	Description
TD [1]	Max	The maximum value of the heart-beat intervals	To describe the distribution of the heat-beat intervals
	Min	The minimum value of the heart-beat intervals	To describe the distribution of the heat-beat intervals
	MEAN	The mean value of the heart-beat intervals	To describe the distribution of the heart-beat intervals
	MEDIAN	The median value of the heart-beat intervals	To describe the distribution of the heart-beat intervals
	SDNN	The SD [4] of the heart-beat intervals	To measure the overall variation of the heart-beat interval sequence
	RMSSD	The root mean square of the heart-beat intervals	To reflect the rapid variation of the heart-beat interval sequence
	PNN50	The percentage of heart-beat intervals longer than 50 ms	To describe the degree of change in the heart-beat interval sequence
FD [2]	vLF	The power in the 0.0033 Hz–0.04 Hz band	To reflect vascular mechanisms caused by negative emotions
	LF	The power in the 0.04 Hz–0.15 Hz band	To reflect sympathetic modulation of heart rate
	HF	The power in the 0.15 Hz band	To reflect parasympathetic activity
	LF/HF	The ratio of power in the LF and HF bands	To reflect the balance of sympathetic nerve and parasympathetic nerve
ND [3]	SampEn	The Sample Entropy value with r = 0.15 × SD	To investigate the complexity of heart-beat interval sequence
	DFA	The short-term coefficient of detrended fluctuation analysis	To investigate the inner correlation of successive heart-beat intervals
	SD_S	The SD of the short-term variability of the Poincare plot	To reflect the short-term variability of the heart-beat intervals
	SD_L	The SD of the long-term variability of the Poincare plot	To reflect the long-term variability of the heart-beat intervals

[1] Time domain features. [2] Frequency domain features. [3] Non-linear domain features.

Table 4. Statistical analysis results of the extracted features.

Type	Features	Normotensive	Hypertensive
Time domain features	Max	1.1704 ± 0.1596	1.0767 ± 0.1804
	Min	0.6800 ± 0.1296	0.7206 ± 0.1152
	MEAN	0.8977 ± 0.1034	0.9010 ± 0.1301
	MEDIAN	0.8860 ± 0.1311	0.9102 ± 0.1405
	SDNN	0.1034 ± 0.0594	0.0539 ± 0.0401
	RMSSD	0.1351 ± 0.0854	0.0698 ± 0.0628
	PNN50	0.9496 ± 0.0331	0.8689 ± 0.0845
Frequency domain features	vLF	5.7261 ± 4.0433	3.6910 ± 3.6003
	LF	3.3063 ± 3.2327	1.1621 ± 1.7799
	HF	6.0722 ± 6.3137	1.9928 ± 3.7635
	LF/HF	0.6995 ± 0.3277	0.9367 ± 0.6504
Non-linear domain features	SampEn	0.63 ± 0.14	0.6831 ± 0.1538
	DFA	0.7 ± 0.2	0.65 ± 0.23
	SD_S	22.99 ± 2.56	18.67 ± 1.76
	SD_L	51.87 ± 6	37.39 ± 3.42

4.2.2. Feature Discretization

Since all the extracted features are continuous variables and cannot be directly used to mine CARs, we utilize equal-width binning strategy to discretize the feature values into several equal-width intervals. In this study, the value range of each feature is finally discretized into five equal-width intervals based on doctors' advice, which represent "very low", "low", "medium", "high", and "very high", respectively. The class attribute is inherently discrete and has two categories, i.e., hypertensive patients and normotensive

subjects. Finally, these discretized features (the class attribute included) are mapped to "very low", "low", "medium", "high", or "very high" based on the method mentioned in Section 3.

Here is an example of feature discretization based on the extracted feature shown in Table 4. The range of feature SampEn is [0.386, 0.915], which is equal-width binned into five sub-ranges, i.e., [0.386, 0.4918), [0.4918, 0.5976), [0.5976, 0.7034), [0.7034, 0.8092), and [0.8092, 0.915], corresponding to "very low", "low", "medium", "high", and "very high", respectively. If a hypertensive subject has a SampEn value of 0.5, then his SampEn value will be mapped into "low". Similarly, a SampEn value of 0.6 will be mapped into "medium", and so on.

4.3. Hypertension Classification

When classifying a new subject, the K most similar CARs are first picked out from the hypertensive group and normotensive group, respectively, by using Formula (7). Afterwards, the class of the subject is determined based on the similarities and weights of the selected CARs by employing Formulas (8)–(10). The optimal value of the parameters in the above process, i.e., the K, etc., will be tuned and discussed in Section 6.

5. Experimental Evaluation

5.1. Experimental Setup

We evaluate the performance of the proposed C-MWCAR on the hypertension diagnosis task based on a real BCG dataset of 128 subjects. In particular, 61 of them are hypertensive and the rest are normotensive. For each subject, one night of BCG is collected while sleeping, since at that time physiological indexes remain relatively stable. To achieve robust evaluation results, 10-fold cross-validation is utilized. Concretely, the dataset is randomly divided into ten portions, and nine of them are used as the training set and the last one is used as the test set. Repeat this process ten times until each portion has already been selected as the test set just once. Finally, ten groups of classification results are obtained, and the averaged values are regarded as the final classification results. The number of subjects in each portion is shown in Table 5.

Table 5. The division of the dataset.

Type	Por.* 1	Pro. 2	Pro. 3	Por. 4	Por. 5	Por. 6	Por. 7	Por. 8	Por. 9	Por. 10	SUM
Hypertensive	6	6	6	6	6	6	6	6	6	7	61
Normotensive	7	7	7	7	7	7	7	7	7	6	68
Subtotal	13	13	13	13	13	13	13	13	13	13	128

* is the abbreviation of "Portion".

When conducting the experiments, *userSup* and *userConf* are initialized to 0.25 and 0.75, respectively. Additionally, the length of the same prefixes (i.e., the k in BCSA), the coverage threshold (i.e., the C in BCSA), the redundancy threshold (i.e., the R in BCSA), and the number of similar CARs (i.e., the K in MWCC) are initialized to 4, 0.6, 0.85, and 3, respectively. The selection process of these parameters is detailedly shown in Section 5.2.3.

Three widely used metrics, i.e., accuracy (ACC), sensitivity (SEN), and specificity (SPE), are also utilized in this study [20].

The experiments are conducted in Matlab 2014a and Eclipse Neon on an Intel Core i5-4590 PC with an 8 GB RAM running Windows 7 operating system.

5.2. Evaluation Results

5.2.1. Classification Performance Analysis

We first quantitatively study the performance improvements contributed by the proposed BCSA and MWCC. As shown in Table 6, we set four methods for comparison, named as "NONE", "BCSA", "MWCC," and "BCSA + MWCC", respectively. In particular, "NONE" means that neither the BCSA nor the MWCC is adopted when building the

classifier, "BCSA" means that the BCSA algorithm is adopted while the MWCC algorithm is not adopted when building the classifier, and "BCSA + MWCC" indicates that both of the methods are adopted when building the classifier. From Table 6, we find that "BCSA" obtains 5.0% of ACC, 5.6% of SEN, and 5.4% of SPE improvements, while "MWCC" obtains 7.1% of ACC, 8.1% of SEN, and 6.3% of SPE improvements in comparison with "NONE", which indicates that selecting a representative and concise set of CARs and making use of multiple weighted CARs are very conducive to achieving more accurate results. Moreover, once BCSA and MWCC are combined, the performance significantly improves and ACC, SEN, and SPE increase to 93.3%, 93.8%, and 92.7%, respectively, which shows the effectiveness of our methods.

Table 6. Classification performance improvements contributed by BCSA and MWCC.

Combinations	ACC	SEN	SPE
NONE	78.8%	79.6%	78.3%
BCSA	83.8%	85.2%	83.7%
MWCC	85.9%	87.7%	84.6%
BCSA + MWCC	93.3%	93.8%	92.7%

Afterwards, we compared the proposed C-MWCAR with four baselines proposed by Poddar et al. [29], Ni et al. [20], Liu et al. [7], and Li et al. [8]. Poddar et al. [29] extracted many HRV features and then directly inputted them into an LibSVM classifier. Ni et al. transformed the extracted HRV features into different feature representations by using feature pooling techniques, and then modeled obtained feature representations by utilizing a L1-regularized logistic regression classifier [17]. The third [7] and fourth [8] methods are two famous CARs-based classifiers, i.e., CBA and CMAR. CBA uses only one CAR to classify a given instance (if there are no matched CARs, the instance will be labeled with a default class), while CMAR employs multiple CARs to determine the final class labels. Note that for fair comparison, these baselines are corrected based on the BCG dataset collected in this study. In particular, Refs. [20,29] extracted their own HRV features, while CBA and CMAR utilized the HRV features extracted in this study. In addition, the best parameter combinations of the baselines were adopted. As shown in Figure 4, C-MWCAR obtained the highest performance, and outperformed the method in [29] by 17.7%, 20.8%, and 17.9% in terms of ACC, SEN, and SPE, respectively, which indicates that extracting the associate relationship among features is useful for classification. The method in [20] showed slightly higher performance than the method in [29] (2.2% of ACC, 2.0% of SEN, and 3.5% of SPE) as it employed more complex feature combinations; however, its performance was much lower than C-MWCAR (13.3% of ACC, 13.8% of SEN, and 11.3% of SPE). The performance of CBA was lower than CMAR by 4.3%, 4.6%, and 4.3%, and much lower than C-MWCAR by 10.2%, 12.1%, and 9.9% in terms of ACC, SEN, and SPE, respectively, which means that utilizing multiple CARs can provide more useful information for classification. In particular, C-MWCAR and CMAR use the same features but the former outperformed the latter by 5.9% of ACC, 7.5% of SEN, and 5.6% of SPE, which may be because C-MWCAR can select the most similar CARs and exploit their weighted classification abilities when classifying each instance.

5.2.2. Efficiency of the Proposed DOCMA

We first investigate the efficiency of the designed DOCMA under different combinations of support and confidence. From Figure 5, we can easily find that the time overhead mainly depends on the support rather than the confidence. Furthermore, it will increase significantly if the support decreases slightly. In particular, it takes nearly 4 h to mine all CARs when the proposed DOCMA is not employed and the support is set to 0.2, which is obviously unacceptable in practice. However, once the proposed DOCMA is applied, the time overhead will decrease by more than half. Concretely, compared with the optimized pruning strategy and the pre-pruning strategy, the optimized connection strategy brings

the largest time savings. In particular, it reduces the time overhead by approximately 44%, 39%, and 47% when the support is set to 0.3, 0.25, and 0.2, respectively. Moreover, if the optimized pruning strategy and the pre-pruning strategy are also employed, an additional 24.4–36.2% time savings can be obtained. In total, C-MWCAR can save 68.3–81.2% of time and the best and worst performance are obtained when support and confidence are set to 0.3 and 0.7, and 0.25 and 0.8, respectively.

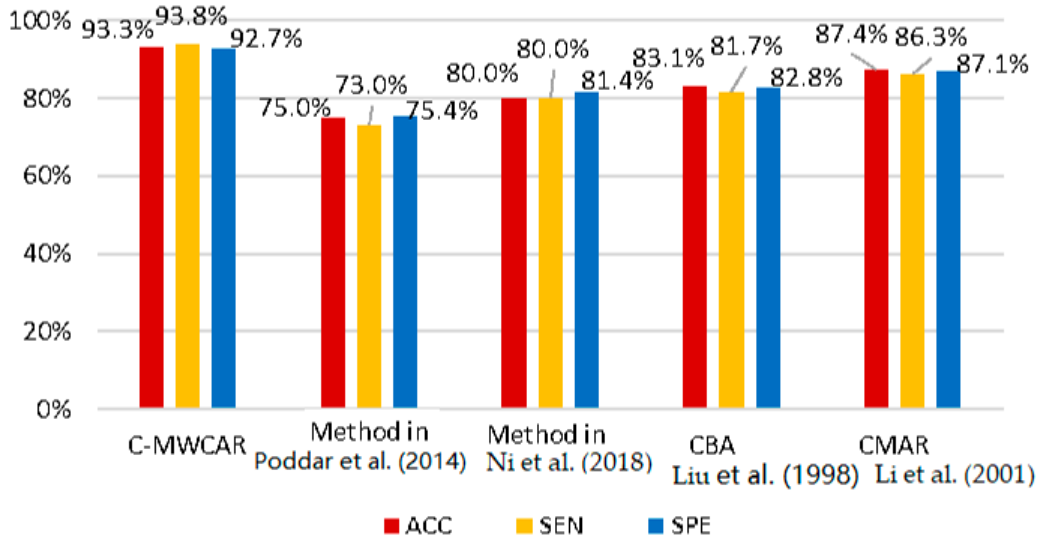

Figure 4. Performance comparison between C-MWCAR and four state-of-the-art baselines [7,8,20,29].

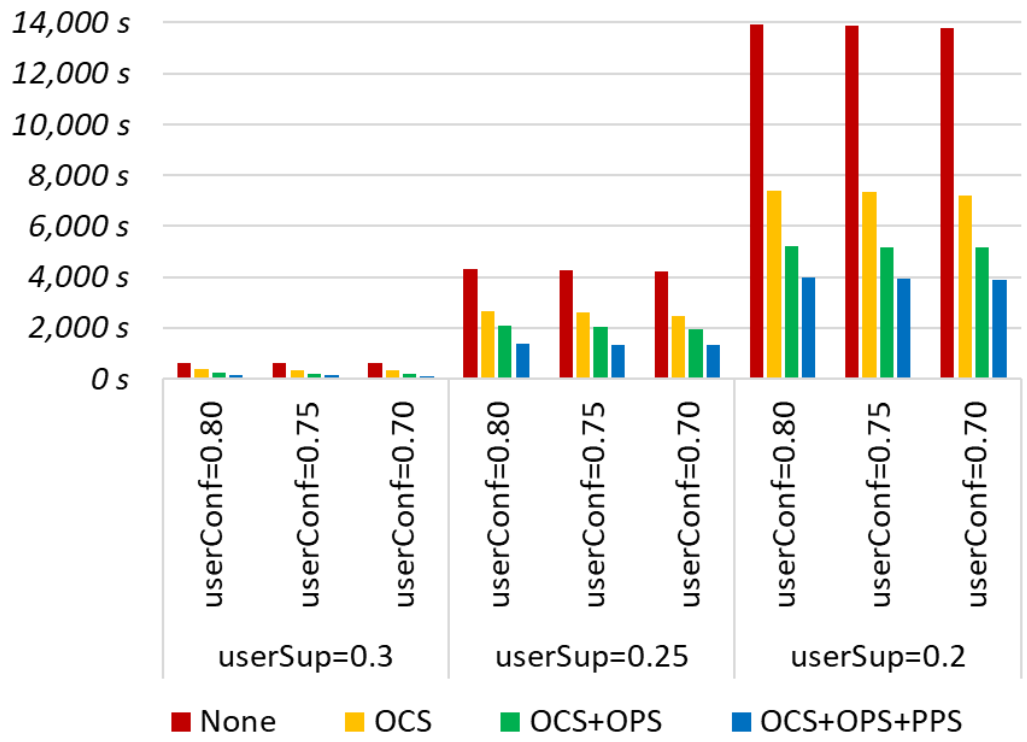

Figure 5. Time overhead of the mining of the CARs under different combinations of support and confidence. In particular, OCS, OPS, and PPS stand for the optimized connection strategy, the optimized pruning strategy, and the pre-pruning strategy described in Section 4.2, respectively. None means that none of those strategies is employed.

Next, we compare DOCMA with three state-of-the-art CAR mining methods, i.e., CAR-Miner [12], ECR-CARM [23], and PMCAR [18], with the confffidence fixed at 0.8. From Table 7, we find that DOCMA spends the least time except for when the support is set to 0.2. Concretely, compared with CAR-Miner and ECR-CARM, DOCMA can save 3.6%, 36.5%, and 69.2%, and 38.1%, 60.7%, and 82.9% of the time when support is set to 0.2, 0.25, and 0.3, respectively. Compared with PMCAR, DOCMA spends 17.7% more time when support is set to 0.2 but saves 15.7% and 47.9% of the time when support is set to 0.25 and 0.3, respectively.

Table 7. Comparison of the efficiency of mining CARs between DOCMA and three state-of-the-art baselines.

Algorithms	Strategies Used to Mine CARs	Time Overhead [1]		
		Sup = 0.2	Sup = 0.25	Sup = 0.3
DOCMA	Use the Apriori-like algorithm with optimized strategies to generate rules	3976.3 s	1369.7 s	129.9 s
CAR-Miner [12]	Mine CARs based on the modified ECR-tree with Obidset	4123.5 s	2156.8 s	421.8 s
ECR-CARM [22]	Use equivalence class rule tree (ECR-tree) to generate candidate rules	6425.8 s	3482.5 s	761.3 s
PMCAR [24]	Mine CARs with parallel and distributed approaches	3379.4 s	1624.4 s	249.5 s

[1] The confidence is fixed at 0.8.

5.2.3. Parameter Optimization

The support and confidence are two key parameters when mining CARs, and directly affect the number of extracted CARs. If the number of CARs is too small, it may miss some useful CARs; however, if the number of CARs is too large, it will incur too much noise and redundancy. The number of mined CARs under different combinations of support and confidence are shown in Figure 6, which shows that the number of the CARs mainly depends on the support rather than the confidence. It soars sharply if the support decreases, while it increases slightly if the confidence decreases. In particular, the number of CARs is neither too small nor too large when the support is set to 0.25. In addition, we notice that the growth of the number of CARs caused by the decrease of confidence when support is set to 0.3 or 0.2 is more moderate than that when support is set to 0.25, which signifies that the distribution of the CARs is much more even with respect to the confidence when support equals 0.25. Therefore, the support and confidence are finally set to 0.25 and 0.75, respectively.

When selecting CARs, the length of the same prefix k, the redundancy threshold R, and the coverage threshold C directly determine the final CARs set, which will further affect the final classification results. Too large an L value may lead to missing some effective CARs, but too small an L value may make the selected CARs almost unrepresentative. Likewise, a big R value makes selected CARs contain too much redundant information, while a small R will leave out useful information. Moreover, if the value of C is too large or too small, it may lead to over-fitting or under-fitting. In addition, in the classification process, the number of similar CARs used to classify instances, i.e., K, is also a key parameter that can directly affect the classification performance. Obviously, too small an N value cannot obtain enough information from the CARs for classification, while too large an N value will introduce unimportant information and hence distort the classification results. As shown in Figure 7, we investigate the performance as k, R, C, and K vary independently.

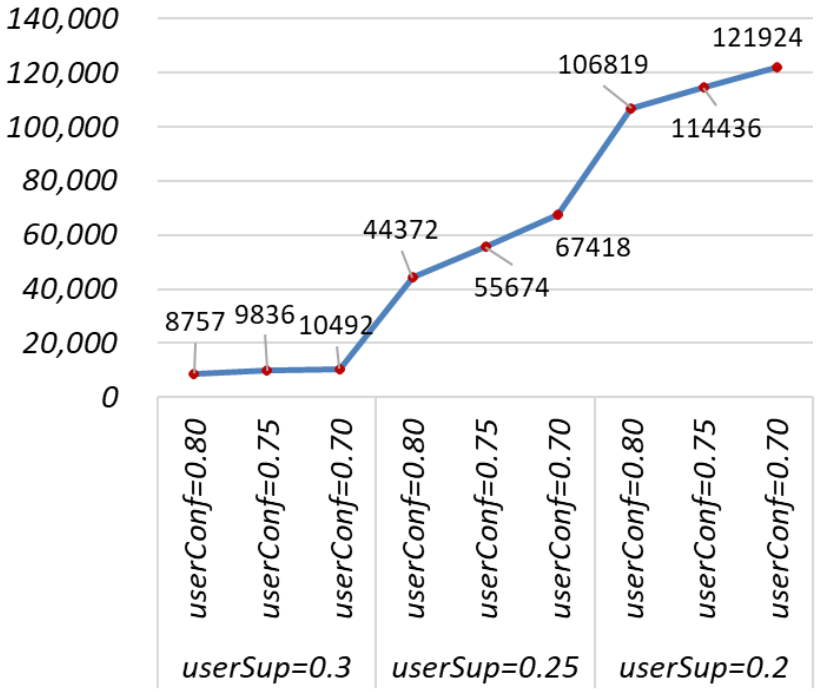

Figure 6. The number of mined CARs under different combinations of support and confidence.

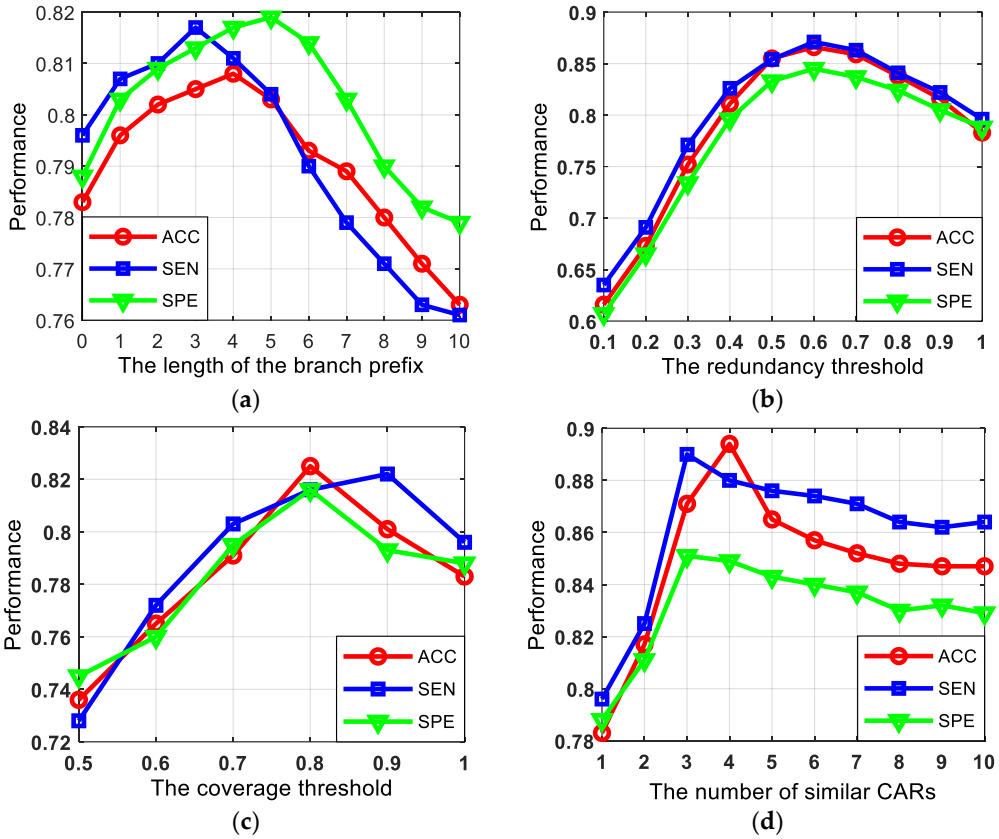

Figure 7. Optimization of the key parameters of the C-MWCAR. (**a**–**d**) reflect the variation of ACC, SEN, and SPE when the length of the branch prefix, the redundancy threshold, the coverage threshold, and the number of similar CARs change, respectively. Note that when studying the effects of a specific parameter, the other parameters are set to default values. The default values of these four parameters are 4, 0.6, 0.85, and 3, respectively.

Figure 7a shows that ACC, SEN, and SPE have the same trend and they all first increase and then decrease. They obtain their maximum when the length of the same prefix is set to 4, 3, and 5, respectively. In order to keep all three metrics high, we make a trade-off and suggest setting the length of the prefix as 4. As shown in Figure 7b, the maximum values of the three metrics are obtained when the redundancy threshold is 0.6. Figure 7c shows that ACC and SPE reach their maximums when the coverage threshold C is set to 0.8. However, for SEN, the optimal coverage threshold is 0.9, where ACC and SPE have decreased considerably. To keep all three metrics relatively high, the coverage threshold C is finally set to 0.85. In Figure 7d, all of the metrics rise sharply at first, and then decline gradually, which can be explained by the fact that utilizing multiple CARs will bring more comprehensive and stable information and hence improve the classification performance; however, too many CARs will lead to over-fitting and thus result in reduced performance. In this study, the number of CARs is set to 3, since the increment of ACC is less than the reduction of SEN when the number of similar CARs varies from 3 to 4.

5.2.4. Visualization of the Mined CARs

We randomly select five CARs from the hypertensive group and normotensive group, respectively, and then visualize them in Figure 8, with the null values of each CAR replaced by the average value of the current item in the current group. From Figure 8, we find that the overall trends of the CARs derived from the two groups are quite different from each other. Concretely, the overall trend of the normotensive group remains relatively high at first, and then gradually decreases until the end, except for the fluctuations at MIN, PNN50, and SD1. As for the hypertensive group, its overall trend is just the opposite. More specifically, the hypertensive group usually has smaller time-frequency domain features except for MIN and PNN50 but bigger non-linear domain features compared with the normotensive group, which is coherent with the results reported in previous medical studies [31,38–40]. Hence, we think that the extracted CARs are conducive for doctors to analyze the patient's condition. For example, if the subject's features converge towards to or are similar to the CARs of the hypertensive group, then the doctors have reasons to believe that the subject is likely to suffer from hypertension in the future, and thus give him or her effective health interventions at once, which may avoid the occurrence or prevent the deterioration of hypertension.

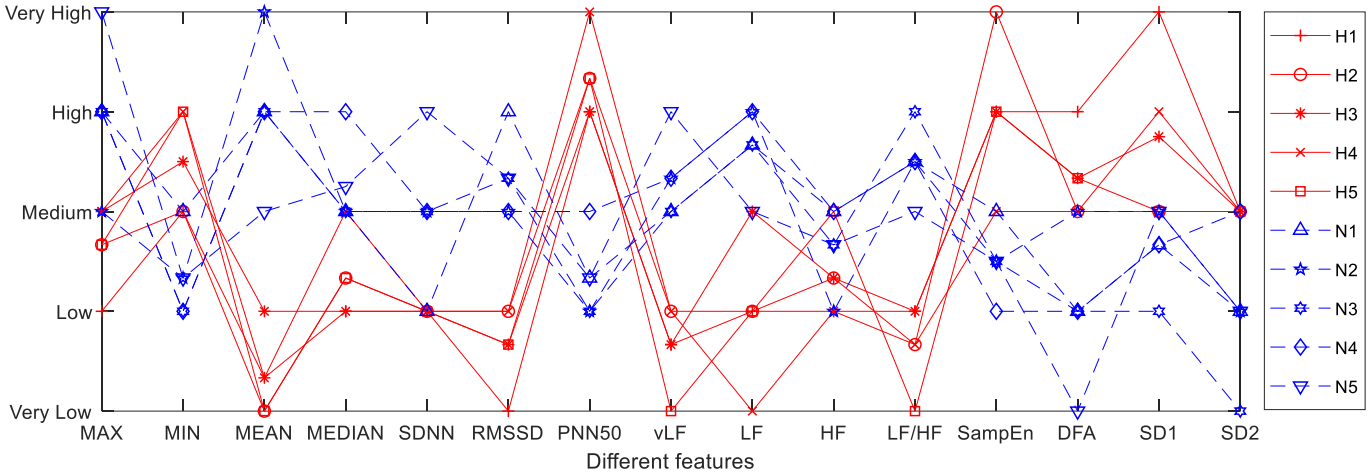

Figure 8. Visualization of the five most powerful CARs in the hypertensive group and normotensive group. The null value in CARs are replaced by the average value of the current group, and the CARs for the hypertensive group and normotensive group are plotted by using the red solid line and blue dotted line, respectively.

6. Discussion

This paper proposed a multiple weighted class association rules-based classifier (i.e., C-MWCAR) and applied it to a real-world classification task. The experimental results demonstrated the effectiveness and interpretability of the proposed method. In this section, the advantages and limitations of C-MWCAR are discussed.

The advantages of C-MWCAR are as follows. Firstly, the effectiveness of C-MWCAR is excellent. In particular, the BCSA of C-MWCAR can select representative CARs from each branch for classification, and the MWCC of C-MWCAR can not only solve the problem of having no matching CARs and the problem of having conflicting CARs but can also make full use of the power of multiple CARs that are the most similar to the subject to be classified. Therefore, C-MWCAR is much more accurate and useful when applied to real-world problems. Secondly, the interpretability of C-MWCAR is impressive. C-MWCAR can generate a set of CARs that can be used to explain why the subject is classified as hypertensive or normotensive to a certain degree. However, traditional classifiers, taking the Decision Tree classifier as an example, can generate a set of IF–THEN rules. Nevertheless, its interpretability is limited, since Decision Tree only takes one feature into consideration when building each tree node, instead of considering all the extracted features simultaneously, which may leave out some valuable association relationship among features. Moreover, the IF–THEN rules heavily rely on continuous feature values rather than discrete feature values, therefore, once the dataset changes slightly, the IF–THEN rules may have to adjust in order to adapt to the changed feature values. However, C-MWCAR deems the features non-independent intrinsically, which accords with the actual situation, so the interpretability of the CARs is more credible.

The limitations of C-MWCAR are as follows. Firstly, the process of feature discretization may be affected by outliers. If there are some outliers for a feature, the feature values of hypertensive subjects or normotensive subjects in terms of the feature may be discretized into the same sub-range rather than different sub-ranges, which makes the feature lose its distinguishing ability. Secondly, the scale of the dataset collected in this study is not large since human-related physiological signals are hard to collect due to various restrictions in practice, which may make the classification performance not stable. To obtain more stable and reliable performance evaluation results, it is suggested to recruit more subjects to collect enough data samples. Thirdly, the equal-width binning strategy used in this study to discretize the features may be not the most powerful. In future, the performance of different feature discretization strategies should be investigated. Fourthly, only the most frequently used features are extracted in this study. To achieve higher classification accuracy, more features should be explored and studied. It is notable that the increased number of features almost only increases the time overhead of the process of CAR extraction but virtually has no effect on the time overhead of applying the classifier to classify the subjects.

7. Conclusions and Future Work

In this paper, we proposed a novel classification framework named C-MWCAR, which is able to obtain better classification performance by utilizing multiple weighted CARs. In particular, C-MWCAR can select representative and concise CARs based on the distribution, coverage, and redundancy of the CARs, and then build an effective classifier by considering the similarities between CARs and the instance and the weighted classification abilities of the CARs simultaneously. In addition, the designed CAR mining algorithm (i.e., DOCMA) can significantly reduce the time overhead used for extracting the complete set of CARs, which makes the application of CARs-based classifiers to real-world problems possible. Experimental results based on a real BCG dataset of 128 subjects demonstrated that the proposed C-MWCAR achieved 93.3%, 93.8%, and 92.7% in terms of ACC, SEN, and SPE, respectively, which outperformed four baselines significantly. In addition, the proposed DOCMA proved to be capable of reducing the time overhead by up to 81.2% when *sup* and *conf* were set to 0.25 and 0.8, respectively. Moreover, the visualization of the CARs denoted

that the mined CARs can explicitly reflect a subject's physiological status; in other words, the classification results were interpretable to some extent.

The contributions of this paper can be summarized as follows.

- To accelerate the extraction of CARs, a dictionary order-based CAR mining algorithm (DOCMA) was designed by optimizing the Apriori algorithm [26]. To be specific, we designed and proved a set of theorems and corollaries, by which the most time-consuming steps, i.e., the connection of frequent $k-1$ itemSets and the pruning of infrequent k itemSets, can be greatly simplified. In addition, it can also pre-prune the set of frequent $k-1$ itemSets, which further reduces the number of connection operations required. Experimental results showed that the proposed DOCMA saved up to 81.2% of time and outperformed three state-of-the-art CAR mining methods.
- In order to deal with the numerous CARs, a branch-based CAR selection algorithm (BCSA) was designed to select the most representative and concise CARs. It first grouped all the CARs into non-overlapping branches, and then selected the most representative CARs from each of them, which made the selected CARs more representative. After that, we designed two metrics, named coverage and redundancy, by which the CARs contained little new information but much redundant information was filtered out, making the final set of selected CARs more concise and effective.
- To obtain higher classification performance, we proposed a multiple weighted CARs-based classifier (MWCC) that classifies an instance by using multiple weighted CARs. Specifically, it first picks out a set of CARs that are most similar to the given instance, then it fuses their weighted classification abilities to compute the final classification result. This strategy can not only solve the problem of no matching CARs but can also achieve more accurate and robust classification results.
- We applied the proposed C-MWCAR to a real classification task, i.e., hypertension diagnosis. Experimental results showed that C-MWCAR outperformed four state-of-the-art baselines, and achieved 93.3%, 93.8%, and 92.7% in terms of accuracy, sensitivity, and specificity, respectively. In addition, we found that the generated CARs can intuitively reflect the patient's physiological status, which signifies that C-MWCAR is interpretable to some extent.

In the future, we will aim to build a new kind of CARs-based classifier that is much simpler but can produce better classification performance, and then test it on more real-world datasets.

Author Contributions: The work described in this article is the collaborative development of all authors. Conceptualization of data processing, G.L. and F.L.; algorithm design, G.L. and F.L.; data measurement, Y.Y., C.W. and Z.W.; analysis, Y.Y., C.W. and Z.W.; writing—original draft preparation, G.W. and Y.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work is partially supported by the National Natural Science Foundation of China (No. 61960206008, 62072375) and the Major Key Project of PCL (Grant No. PCL2022A03, PCL2021A02, PCL2021A09).

Institutional Review Board Statement: Participants provided verbal informed consent; the reason why written consent had not been obtained is that we only invited those who agreed to share their data for scientific research to participate the experiment; the ethics committees have approved this consent procedure. No: 20170078.

Informed Consent Statement: Informed consent was obtained from all subjects involved in this study.

Data Availability Statement: The data can be provided only on the condition of requesting us and we will share it by E-mail due to privacy.

Acknowledgments: The authors would like to thank the support of the laboratory, university, and government.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Liu, F.; Zhou, X.; Wang, Z.; Ni, H.; Wang, T. OSA-weigher: An automated computational framework for identifying obstructive sleep apnea based on event phase segmentation. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *10*, 1937–1954. [CrossRef]
2. Ma, J.; Sun, L.; Wang, H.; Zhang, Y.; Aickelin, U. Supervised anomaly detection in uncertain pseudoperiodic data streams. *ACM Trans. Internet. Technol.* **2016**, *16*, 4. [CrossRef]
3. Ren, S.; He, K.; Girshick, R.; Zhang, X.; Sun, J. Object detection networks on convolutional feature maps. *IEEE Trans. Pattern Anal. Mach. Intell.* **2016**, *39*, 1476–1481. [CrossRef]
4. Wu, Z.; Xu, Q.; Li, J.; Fu, C.; Xuan, Q.; Xiang, Y. Passive indoor localization based on csi and naive bayes classification. *IEEE Trans. Syst. Man Cybern. Syst.* **2017**, *48*, 1566–1577. [CrossRef]
5. Paul, S.K.; Nine, M.S.Q.Z.; Hasan, M.; Amin, M.A. Cognitive Task Classificaiton from Wireless EEG. In Proceedings of the 8th International Conference, BIH 2015, London, UK, 30 August–2 September 2015; pp. 13–22.
6. Liu, F.; Zhou, X.; Cao, J.; Wang, Z.; Wang, H.; Zhang, Y. Arrhythmias classification by integrating stacked bidirectional LSTM and two-dimensional CNN. In Proceedings of the 23rd Pacific-Asia Conference, PAKDD 2019, Macau, China, 14–17 April 2019; pp. 136–149.
7. Liu, B.; Hsu, W.; Ma, Y. Integrating classification and association rule mining. In Proceedings of the Fourth International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 27–31 August 1998; pp. 27–31.
8. Li, W.; Han, J.; Pei, J. CMAR: Accurate and efficient classification based on multiple class-association rules. In Proceedings of the 2001 IEEE International Conference on Data Mining, California, CA, USA, 29 November–2 December 2001; pp. 369–376.
9. Wen, J.; Zhong, M.; Wang, Z. Activity recognition with weighted frequent patterns mining in smart environments. *Expert Syst. Appl.* **2015**, *42*, 6423–6432. [CrossRef]
10. Veloso, A.; Meira, W.; Zaki, M.J. Lazy associative classification. In Proceedings of the Sixth International Conference on Data Mining (ICDM'06), Hong Kong, China, 12–18 December 2006; pp. 645–654.
11. Kliegr, T.; Kuchař, J.; Sottara, D.; Vojíř, S. Learning business rules with association rule classifiers. In *Lecture Notes in Computer Science*, 1st ed.; Springer: Cham, Germany, 2014; Volume 8620, pp. 236–250.
12. Nguyen, L.T.; Vo, B.; Hong, T.-P.; Thanh, H.C. CAR-Miner: An efficient algorithm for mining class-association rules. *Expert Syst. Appl.* **2013**, *40*, 2305–2311. [CrossRef]
13. Huynh-Thi-Le, Q.; Le, T.; Vo, B.; Le, B. An efficient and effective algorithm for mining top-rank-k frequent patterns. *Expert Syst. Appl.* **2015**, *42*, 156–164. [CrossRef]
14. Kargarfard, F.; Sami, A.; Ebrahimie, E. Knowledge discovery and sequence-based prediction of pandemic influenza using an integrated classification and association rule mining (CBA) algorithm. *J. Biomed. Inform.* **2015**, *57*, 181–188. [CrossRef]
15. Park, S.H.; Reyes, J.A.; Gilbert, D.R.; Kim, J.W.; Kim, S. Prediction of protein-protein interaction types using association rule based classification. *BMC Bioinf.* **2009**, *10*, 36. [CrossRef]
16. Soni, J.; Ansari, U.; Sharma, D. Intelligent and effective heart disease prediction system using weighted associative classifiers. *Int. J. Comput. Sci. Eng.* **2011**, *3*, 2385–2392.
17. Soni, S.; Vyas, O. Using associative classifiers for predictive analysis in health care data mining. *Int. J. Comput. Appl.* **2010**, *4*, 33–37. [CrossRef]
18. Nguyen, D.; Vo, B.; Le, B. Efficient strategies for parallel mining class association rules. *Expert Syst. Appl.* **2014**, *41*, 4716–4729. [CrossRef]
19. Poddar, M.G.; Kumar, V.; Sharma, Y.P. Linear-nonlinear heart rate variability analysis and SVM based classification of normal and hypertensive subjects. *J. Electrocardiol.* **2013**, *46*, e25. [CrossRef]
20. Ni, H.; Cho, S.; Mankoff, J. Automated recognition of hypertension through overnight continuous HRV monitoring. *J. Ambient Intell. Hum. Comput.* **2018**, *9*, 2011–2023. [CrossRef]
21. Melillo, P.; Izzo, R.; Orrico, A. Automatic prediction of cardiovascular and cerebrovascular events using heart rate variability analysis. *PLoS ONE* **2015**, *10*, e0118504. [CrossRef]
22. Zhao, M.; Cheng, X.; He, Q. An algorithm of mining class association rules. In Proceedings of the 4th International Symposium on Intelligence Computation and Applications, ISICA 2009, Huangshi, China, 23–25 October 2009; pp. 269–275.
23. Vo, B.; Le, B. A novel classification algorithm based on association rules mining. In Proceedings of the Pacific Rim Knowledge Acquisition Workshop, PKAW 2008, Hanoi, Vietnam, 15–16 December 2008; pp. 61–75.
24. Nguyen, L.T.; Vo, B.; Hong, T.-P.; Thanh, H.C. Classification based on association rules: A lattice-based approach. *Expert Syst. Appl.* **2012**, *39*, 11357–11366. [CrossRef]
25. Dong, G.; Zhang, X.; Wong, L.; Li, J. CAEP: Classification by aggregating emerging patterns. In Proceedings of the Second International Conference, DS'99, Tokyo, Japan, 6–8 December 1999; pp. 30–42.
26. Agrawal, R.; Srikant, R. Fast algorithms for mining association rules. In Proceedings of the 20th VLDB, Santiago de, Chile, Chile, 12–15 September 1994; pp. 487–499.
27. World Health Organization. A global brief on hypertension: Silent killer, global public health crisis. In *World Health Day 2013*; WHO: Geneva, Switzerland, 2013; pp. 1–39.
28. Poddar, M.G.; Kumar, V.; Sharma, Y.P. Automated diagnosis of coronary artery diseased patients by heart rate variability analysis using linear and non-linear methods. *J. Med. Eng. Technol.* **2015**, *39*, 331–341. [CrossRef]

29. Poddar, M.G.; Kumar, V.; Sharma, Y.P. Heart rate variability based classification of normal and hypertension cases by linear-nonlinear method. *Def. Sci. J.* **2014**, *64*, 542–548. [CrossRef]
30. Sommermeyer, D.; Zou, D.; Eder, D.N.; Hedner, J.; Ficker, J.H.; Randerath, W.; Priegnitz, C.; Penzel, T.; Fietze, I.; Sanner, B.; et al. The use of overnight pulse wave analysis for recognition of cardiovascular risk factors and risk: A multicentric evaluation. *J. Hypertens.* **2014**, *32*, 276–285. [CrossRef]
31. Tejera, E.; Areias, M.J.; Rodrigues, A.I.; Nieto-Villar, J.M.; Rebelo, I. Blood pressure and heart rate variability complexity analysis in pregnant women with hypertension. *Hypertens. Pregnancy* **2012**, *31*, 91–106. [CrossRef]
32. Yue, W.-W.; Yin, J.; Chen, B.; Zhang, X.; Wang, G.; Li, H.; Chen, H.; Jia, R.-Y. Analysis of heart rate variability in masked hypertension. *Cell Biochem. Biophys.* **2014**, *70*, 201–204. [CrossRef] [PubMed]
33. Liu, F.; Zhou, X.; Wang, Z. Identifying Obstructive Sleep Apnea by Exploiting Fine-Grained BCG Features Based on Event Phase Segmentation. In Proceedings of the 2016 IEEE 16th International Conference on Bioinformatics and Bioengineering (BIBE), Taichung, Taiwan, 31 October–2 November 2016; pp. 293–300.
34. Inan, O.T.; Migeotte, P.-F.; Park, K.-S.; Etemadi, M.; Tavakolian, K.; Casanella, R.; Zanetti, J.; Tank, J.; Funtova, I.; Prisk, G.K.; et al. Ballistocardiography and seismocardiography: A review of recent advances. *IEEE J. Biomed. Health Inform.* **2014**, *19*, 1414–1427. [CrossRef] [PubMed]
35. Li, W.; Wang, R.; Huang, D. Assessment of Micro-movement Sensitive Mattress Sleep Monitoring System (RS611) in the detection of obstructive sleep apnea hypopnea syndrome. *Chin. J. Gerontol.* **2015**, *35*, 1160–1162.
36. Vollmann, D.; Sossalla, S.; Schroeter, M.R. Renal artery ablation instead of pulmonary vein ablation in a hypertensive patient with symptomatic, drug-resistant, persistent atrial fibrillation. *Clin. Res. Cardiol.* **2013**, *102*, 315–318. [CrossRef]
37. MAP Health Watcher. Available online: https://maphealthwatch.com/ (accessed on 20 December 2018).
38. Singh, J.P.; Larson, M.G.; Tsuji, H.; Evans, J.C.; O'Donnell, C.J.; Levy, D. Reduced heart rate variability and new-onset hypertension: Insights into pathogenesis of hypertension: The Framingham Heart Study. *Hypertension* **1998**, *32*, 293–297. [CrossRef]
39. Shyma, P.; Pal, G.K.; Habeebullah, S.; Shyjus, P. Decreased total power of HRV with increased LF power in early part of pregnancy predicts development PIH in Indian population. *Biomedicine* **2008**, *28*, 104–107.
40. Mussalo, H.; Vanninen, E.; Ikäheimo, R.; Laitinen, T.; Laakso, M.; Länsimies, E.; Hartikainen, J. Heart rate variability and its determinants in patients with severe or mild essential hypertension. *Clin. Physiol. Funct. Imaging* **2001**, *21*, 594–604. [CrossRef]