Integrating Sequential Pattern Mining Techniques and Support Vector Machines for Sequence Classification

Chieh-Yuan Tsai and Yu-Yu Yao

***Abstract*—Sequence classification problem can be found in many real world applications such as protein function prediction, text classification, and so on. Support Vector Machines (SVMs) have been wildly used to deal with sequence classification problem, since SVMs can deal with the nonlinear data and possess high efficiency in classification. However, the most difficult part in SVMs is to design an appropriate kernel function. Therefore, a pairwise sequence similarity kernel is proposed which takes sequential patterns instead of taking k-mers as reference sequences and evaluates the similarity scores between reference sequences and sequence data by the proposed map function. To obtain sequential patterns, three different sequential pattern mining methods are used to extract frequent sequential patterns, frequent closed sequential patterns, and frequent maximal sequential patterns from sequence databases. The three sequential patterns are then evaluated to know which one could achieve higher classification accuracy. A map function, which is similar to edit distance concept, is used in the proposed kernel to calculate the similarity score. Next, the sequence SVM classifier is built according to the proposed pairwise sequence similarity kernel. A set of artificial datasets are employed to test the proposed SVM classification model. The experiment results indicate the proposed SVM classification model using pairwise sequence similarity kernel is efficient and accurate.**

Keywords—sequential pattern mining; sequence classification; kernel method; SVM classifiers

I. INTRODUCTION

A plenty of classifiers such as Neural Network (NN), Naïve Bayes classification, Decision Tree (DT), and Support Vector Machine (SVM) have been proposed. Among them, SVM is one of the most popular methods. Originally, SVM was developed from Vapnik-Chervonenkis (VC) theory and structural risk minimization (SRM) principle [1, 2]. In order to get the best generalization ability and keep resistant to over fitting, SVM attempts to seek out two things. One is to minimize the training set error, and another one is to maximize the margin. Moreover, SVM can provide global minimum and avoid being trapped in local minimum due to the use of convex quadratic programming. SVMs can deal with the nonlinear data and possess high efficiency in the process of classification.

This work was partially supported by the National Science Council of Taiwan (No. 102-2221-E-155-041-MY3).

Chieh-Yuan Tsai is with Department of Industrial Engineering and Management, and Innovation Center for Big Data and Digital Convergence, Yuan Ze University, Taoyuan City, Taiwan (corresponding author: +886-3-4638800 Ext. 2512; fax: +886-3-463-8907; e-mail: cytsai@saturn.yzu.edu.tw).

Yu-Yu Yao, was with Department of Industrial Engineering and Management, Yuan Ze University, Taoyuan City, Taiwan. He is now with Hon-Hai Precision Industry Co.

In SVM, the selection of an appropriate kernel function significantly affect the performance of its operation [3, 4]. Kernel functions enable them to operate in a high-dimensional implicit feature space without ever computing the coordinates of the data in that space. When dealing with numerical data, polynomial kernel and radial-basis function (RBF) kernel are most popular ones. However, many applications , such as text categorization, gene organization rules, and protein fold recognition, are not numerical data but sequence data. This make some popular kernels unsuitable for sequence applications.

[5] proposed Fisher kernel for discrete symbol sequences using a global discrete hidden Markov model (DHMM). [6, 7] presented the spectrum kernel and mismatch kernel where these two kernels used the number of common k-mers as reference sequences. [8] developed the string subsequence kernel that maps a string into a feature vector whose dimension is equal to the number of all possible subsequences of a particular length. This kernel, similar to spectrum kernel and mismatch kernel, considers two sequences as very similar if they share many common substrings. [9, 10, 11] considered the frequency log likelihood ratio based on the probabilities of occurrence of k-mers to be reference sequences. The term frequency log likelihood ratio kernel, which is similar to spectrum kernel and subsequence kernel, maps a discrete symbol sequence onto a feature space.

Those previously mentioned kernels are based on k-mers because k-mers provide a simple way to construct the reference sequence. However, when the number of symbols is large, the number of dimensional feature space increased significantly. For example, if the number of symbols is 12 and k is 3, the number of dimensional feature space is 12^3=1728. Among 1728-dimensional feature space, many worthless dimensional feature spaces are involved. To solve this problem, this paper proposes a method that takes frequent sequential patterns as reference sequences so that the computing cost can be significantly reduced. Specifically, this research develops an efficient support vector machine classifier for sequence data. The proposed sequence classification method consists of two parts: sequential pattern mining and sequence SVM classifier. In sequential pattern mining part, this research provides efficient and useful algorithms to generate three sequential patterns, which are frequent sequential patterns, frequent closed sequential patterns, and frequent maximal sequential patterns, are derived as the reference sequences. The main reason is the amounts of the three sequential patterns might be very different from mining processes. In sequence SVM classifier, the pairwise sequence similarity (PSS) kernel is developed. Map function, which is edit distance algorithm, is

used to compute the similarity scores between sequences data and reference sequences in database. Finally, the sequence data is transformed as a pairwise sequence similarity score vector as input to train the sequence SVM classifier by the proposed kernel method.

II. Sequential Pattern Mining

Sequential pattern mining algorithms address the problem of discovering the existent maximal frequent sequences in a given database. The problem was first introduced by Agrawal and Srikant, where the basic concepts involved in pattern detection were established [12]. In the recent years, several sequential pattern mining algorithms were proposed, but not all assume the same conditions. Some basic definitions are needed, in order to formally introduce the problem. An itemset is a non-empty subset of elements from the item collection, called items. If the data is time dependent, an itemset corresponds to the set of items transacted in a particular instant by a particular entity. The itemset composed of items a and b is denoted by (ab). A sequence is an ordered list of itemsets. A sequence is maximal if it is not contained in any other sequence. A sequence with k items is called a k-sequence. The number of elements (itemsets) in a sequence s is the length of the sequence and is denoted by |s|. The *i*th itemset in the sequence is represented by s_i. and < > denotes the empty sequence. The result of the concatenation of two sequences x and y is a new sequence denoted by xy. The set of considered sequences is usually designated by database (DB), and the number of sequences by database size |DB|. A sequence $a=<a_1a_2...a_n>$ is contained in another sequence $b=<b_1b_2...b_m>$, or a is a subsequence of *b*, if there exist integers $i_1<i_2<\ldots<i_n$ such that $a_1 \subseteq b_{i1}$, $a_2 \subseteq b_{i2}$, …, $a_n \subseteq b_{in}$.

A subsequence s' of s is denoted by $s' \subseteq s$, and by $s' \subset s$ if s' is a proper subsequence of s, i.e. if s' is a subsequence of s but is not equal to *s*. It is usual to assume that the items in an itemset of a sequence are in lexicographic order. This assumption facilitates the design of sequential pattern mining algorithms, avoiding the repetition of some operations (such as the generation of repeated sequences). In this manner, prefixes and suffixes have specific meanings. They are special cases of subsequences: the sequence without the first element in the first itemset of the sequence and without the last element of the last itemset of the sequence, respectively. Finally, the sequential pattern-mining problem may be stated in its entirety. Given a database D of sequences, and some user-specified minimum support threshold σ and constraint c, a sequence is frequent if it is contained in at least σ sequences in the database, satisfying the constraint c.

III. The Proposed Method

The framework of proposed sequence classification method is depicted in Fig. 1. A sequence database *SD* is divided into *n* sub-databases according to the class label of each sequence. Let $SD^c = \{<S_j, c>\}$ be the *c*th sub-database where S_j represents the *j*th sequence, *c* represents the class label for sequence S_j where *c*=1, 2… *n*. For each sub-database SD^c, a sequential pattern mining algorithm is applied for so that a set of sequential patterns SP^c can be generated. Then, a pair of subsets $\{SP^i, SP^j\}$ and their original sub-database pair $\{SD^i, SD^j\}$ are used to build a sequence SVM classifier where $i, j \in \{1, 2\ldots n\}$, $i \neq j$. That is, there are $n(n\text{-}1)/2$ SVM classifiers in total. For example, if $n = 3$, three classifiers are constructed using data of $<\{SD^1, SD^2\}, \{SP^1, SP^2\}>$, $<\{SD^2, SD^3\}, \{SP^2, SP^3\}>$, and $<\{SD^1, SD^3\}, \{SP^1, SP^3\}>$ respectively. When an unknown sequence is inputted, this sequence will be classified by each classifier. Finally, a majority voting scheme is applied based on the class decision of each SVM classifier.

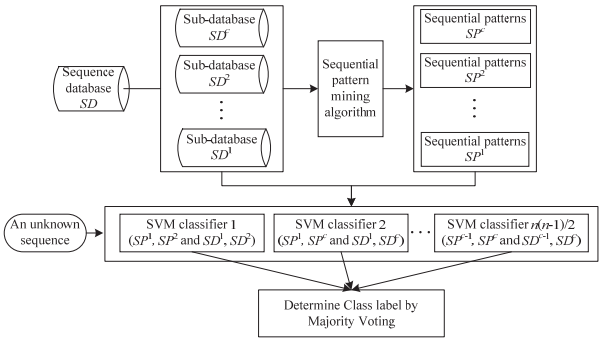


Fig. 1. Framework of the proposed sequence classification method.

A. Sequential pattern mining algorithms

In this study, three sequential patterns are extracted from the same sub-database. They are frequent sequential patterns, frequent closed sequential patterns, and frequent maximal sequential patterns. This research applies PrefixSpan algorithm [13] for generating frequent sequential patterns, CloSpan algorithm [14] for generating frequent closed sequential patterns, and VMSP algorithm [15] for frequent maximal sequential patterns. The three sequential patterns are then evaluated later to know which one could achieve higher classification accuracy.

To make the following discussion easier, a set of frequent sequential patterns is denoted as $FSP^c = \{ fsp_1^c, fsp_2^c, \ldots, fsp_{M_c}^c \}$, a set of frequent closed sequential patterns is denoted as $FCSP^c = \{ fcsp_1^c, fcsp_2^c, \ldots, fcsp_{N_c}^c \}$, and a set of frequent maximal sequential patterns is denoted as $FMSP^c = \{ fmsp_1^c, fmsp_2^c, \ldots, fmsp_{O_c}^c \}$.

B. Sequence Support Vector Machine Classifier

Fig. 2 shows the process of constructing a sequence SVM classifier. There are two main parts in this classifier. The first part is the pairwise sequence similarity kernel. The patterns derived previously are used as reference sequence vectors in the kernel. Therefore, there are three different reference sequence vectors *RS*. Then, all sequences in the database will conduct feature map based on the three reference sequence vectors. The map function *MF* used in this study is edit distance algorithm. The purpose of conducting feature map is to map the sequence into high dimensional feature space. After mapping by edit distance algorithm, pairwise sequence similarity score vectors are obtained.

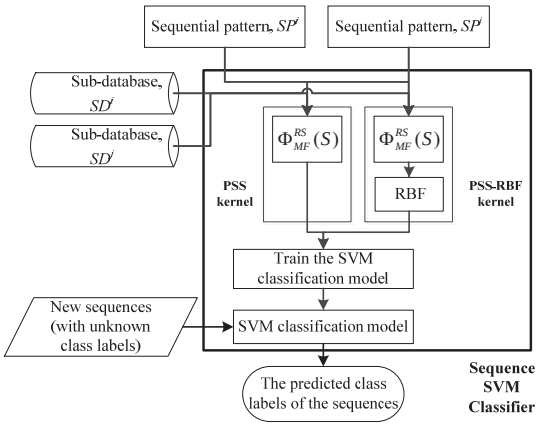


Fig. 2. The proposed sequence SVM classifier with pairwise sequence similarity kernel

The pair of FSP^i and FSP^j is considered as reference sequences for frequent sequential patterns. That is, the reference sequence vectors can be represented as $RS_{fsp} = < fsp^i_1, fsp^i_2, ..., fsp^i_{M_i}, fsp^j_1, fsp^j_2, fsp^j_{M_j} >$. Similarly, the reference sequence vectors for frequent closed sequential patterns and frequent maximum sequential patterns respectively can be expressed as $RS_{fcsp} = < fcsp^i_1, fcsp^i_2, ..., fcsp^i_{N_i}, fcsp^j_1, fcsp^j_2, fcsp^j_{N_j} >$, and $RS_{fmsp} = < fmsp^i_1, fmsp^i_2, ..., fmsp^i_{O_i}, fmsp^j_1, fmsp^j_2, fmsp^j_{O_j} >$. To make the following explanation easier, the above reference sequence vector is formulated as:

$$RS = < rs_1, rs_2, \cdots, rs_w > \quad (1)$$

where rs_x is one of reference sequences.

In this study, a map function which edit distance algorithm is applied for, is denoted as *MF*. The definition of edit distance is the minimal penalty cost between a sequence, *S*, and a reference sequence, rs_x, assessed by sequence similarity. The similarity score of edit distance between two sequences are computed, and an edit distance required to transform rs_x to sequence *S*. Four operations of edit distance which are "substitution", "insertion", "deletion", and "no change". One itemset of rs_x is replaced by the corresponding element in *S* is called "substitution." "Insertion" is that one element of sequence *S* is inserted into one rs_x. Thus, the length of rs_x will increase by one element. One element of sequence *S* is deleted is called "deletion." Therefore, the length of rs_x decreases by one element. "No change" is that itemset of rs_x is the same as the corresponding element in *S*.

Let $\varphi^{RS}_{MF}(S, rs_x)$ be the similarity score function that evaluates the score between sequence *S* and rs_x. The pairwise sequence similarity score vector for sequence *S* is given by

$$\Phi^{RS}_{MF}(S) = < \varphi^{RS}_{MF}(S, rs_1), \varphi^{RS}_{MF}(S, rs_2), ..., \varphi^{RS}_{MF}(S, rs_W) > \quad (2)$$

Alternatively, the pairwise sequence similarity score vector for sequence *S* can be represented as

$$\Phi^{RS}_{MF}(S) = \left\langle \varphi^{RS}_{MF}(S, rs_x) \right\rangle^{w}_{x=1} \quad (3)$$

In addition, to know the performance of the proposed PSS kernel, another version of pairwise sequence similarity with Radial Basis Function (RBF) kernel, denoted as PSS-RFB kernel, is compared. RBF is a popular kernel function which it is commonly used in SVM classification. Two sequences *u* and *v*, represented as feature vectors, is defined as $K(u,v) = exp(-\gamma||x_i - x_j||^2)$, $\gamma > 0$ where $||x_i - x_j||^2$ may be viewed as the squared Euclidean distance between the two feature vectors. γ is the free parameter that implicitly defines the non-linear mapping from input space to some high-dimensional feature space.

The second part is to construct the sequence SVM classifier. The sequence SVM classifier is built according to the proposed pairwise sequence similarity kernel, denoted as PSS kernel. Through the proposed sequence classification framework, the class label of a new sequence will be predicted precisely.

C. Support Vector Machine for Binary and Multiple Classification

The SVM were derived intrinsically for binary classification. There are two main approaches for multi-class SVM classification. One is directly considering all data in one optimization formulation, and another one is to construct binary schemes which break down the multi-class problem into a number of smaller binary problems [16]. The former that the number of variables are used to build and solve the optimization problem for nonlinear SVM, is a positive function of the number of classes [17]. Therefore, it is computationally more expensive to solve a multi-class problem with the same number of data than binary schemes. To avoid the expensive cost of computation, the binary schemes are decided and used in this multi-class problem.

The typical implementation methods of multi-class SVM are one-against-all (OAA) and one-against-one (OAO). In the OAA scheme, the number of binary classifiers is equal to the number of classes. OAA makes the binary decisions between each class and all the other classes. In the OAO scheme, the training of $n(n-1)/2$ binary classifiers is required, where each class separates into two opposite classes in a pairwise way. A voting scheme is used for deciding the final class in OAO approach, and the final class belongs to the one with maximum number of votes. OAO is employed in this research, because OAO is more practical. [16] observed the training process of OAO is quicker than OAA; moreover, [18] and [19] claimed OAO is more accurate than the OAA strategy. That is the main reason why this research uses the OAO approach with nonlinear SVM.

IV. Experiments

A. Datasets

The following experiments are conducted using C# programming language under the environment of Intel(R) Core(TM) i5 2300 CPU with 4.0 GB RAM. Four artificial

datasets are introduced. The number of sequence, the length of sequence, the characters of sequence and total number of classes, sequence composition are generated by a sequence generator for artificial datasets. The detail of the four artificial datasets are summarized and shown in Table I.

TABLE I. THE DETAIL OF ARTIFICIAL DATASET

Dataset	No. of sequences	Len. of sequences	Characters used in sequences	No. of class labels	Sequence composition
*D*1	300	[4, 8]	12	3	[70%,20%,10%]
*D*2	30000	[8, 12]	12	3	[70%,20%,10%]
*D*3	300	[4, 8]	12	3	[60%,20%,20%]
*D*4	30000	[8, 12]	12	3	[60%,20%,20%]

Let's take dataset *D*1 as an example. The sequence in artificial datasets is composed of itemsets. The set of item is assumed as I = {A, B, C, D, E, F, G, H, I, J, K, L}. These 12 items are grouped into three parts, each part contains three items. For instance, {A, B, C} belongs to the first part, {D, E, F} belongs to the second part, and {G, H, I} belongs to the third part. There are three classes (c = 3), and the total sequences are 300 in *D*1 case. In addition, each class has 100 sequences. For each class 80% of sequences are used for the training dataset (*TR*) and 20% of sequences are for the testing dataset (*TE*). The length of sequences is randomly assigned as the range from 4 to 8. For the sequences in class 1, 70% of items are from the first part , 20% of items are from the second part, and the remaining 10% items are from the third part. With the same idea, for the sequences in class 2, 10% of items are from the first part , 70% of items are from the second part, and the remaining 20% items are from the third part. Similarity, for the sequences in class 3, 20% of items are from the first part , 10% of items are from the second part, and the remaining 70% items are from the third part. Table II shows part of sequences of training dataset in *D*1.

TABLE II. THE TRAINING DATA IN D1

SD^c	*D*1		
	sid	S_i	c_i
SD^1	1	<A, D, B, L>	1
	2	<A, C, B, I, C, G, H>	1
	3	<A, E, B, E, C, D, B>	1
	⋮	⋮	⋮
	79	<C, D, C, A>	1
	80	<A, G, B, E, C>	1
SD^2	81	<D, A, C, E, F, A, E>	2
	82	<E, A, D, E, F, L>	2
	83	<F, B, G, H, E, F, D, C>	2
	⋮	⋮	⋮
	159	<E, F, A, D, E, L, D, G>	2
	160	<F, E, F, H, B, L, D, E>	2
SD^3	161	<G, D, K, C, H, B, H>	3
	162	<H, F, A, G, I, C, G, H>	3
	163	<I, A, F, H, G>	3
	⋮	⋮	⋮
	239	<I, B, L, G, H, E, I, H>	3
	240	<H, A, I, G>	3

B. A Case Study

First, the sequences of training data in each class are used to derive. Frequent sequential patterns for class c are generated, denoted as FSP^c where c = 1, 2, and 3 by PrefixSpan algorithm. Table III shows the set of frequent sequential patterns for each class when minimum support is set as 0.3. In addition, frequent closed sequential patterns $FCSP^c$ and frequent maximal sequential patterns $FMSP^c$ are mined using CloSpan algorithm and VMSP algorithm respectively.

TABLE III. FREQUENT SEQUENTIAL PATTERNS IN EACH CLASS

Frequent sequential patterns		
FSP^1	FSP^2	FSP^3
<A>	<D>	<G>
<A, B>	<D, E>	<G, F>
<A, B, C>	<D, F>	<G, G>
<A, B, D>	<D, E, G>	<G, H>
<A, B, D, C>	<D, G>	<H>
<A, C>	<E>	<J>
<A, D>	<E, G>	<K>
<A, D, C>	<F>	<L>
<A, E>	<A>	<A>
<B>	<B>	<B>
<B, C>	<C>	<C>
<B, D>	<G>	<D>
<B, D, C>	<G, F>	<E>
<C>	<G, H>	<F>
<D>	<H>	
<D, C>	<L>	
<D, E>		
<E>		
<G>		

Then, the process of computing the similarity measure between sequential patterns and sequences in *D*1 using edit distance algorithm is performed. Similar process is applied to the testing data in *D*1. Therefore, three similarity score vectors for sequences and frequent sequential patterns, sequences and frequent closed sequential patterns, and sequences and frequent maximal sequential patterns.

Three similarity score vectors which has been generated are used to establish three SVM classifiers. When building SVM classification models, PSS kernel without RBF (called PSS) and PSS with RBF kernel (called PSS-RBF) are evaluated and compared. The values of gamma and cost in RBF are set as 1. Table IV shows the training and testing accuracy of classification model when the value of minimum support is 0.3. For training data, the accuracy of using *FSP* and *FCSP* are better than the accuracy of using *FMSP* when either PSS kernel or PSS-RBF kernel is applied. However, for testing data, the accuracy of using *FMSP* is better than the one of using *FSP* and *FCSP*.

TABLE IV. THE ACCURACY OF THE CLASSIFICATION MODEL WHEN *MIN_SUP* IS 0.3

	PSS kernel		PSS-RBF kernel	
	Training	Testing	Training	Testing
FSP	70.833%	66.667%	97.5%	50%
FCSP	70.833%	66.667%	97.5%	46.667%
FMSP	70%	70%	97.083%	51.667%

Table V illustrates the computational time for training the classification model and calculating accuracy for different classification models. Since the number of *FSP* is largest, computational time for the model using *FSP* will be longer than the models using *FCSP* or *FMSP*. On the other hand, the number of *FMSP* is fewest so that the computational time for training the classification model and calculating accuracy is the fastest. In addition, the computational time for the three models using PSS-RBF kernel is longer than that the computational time for the three models using PSS kernel.

TABLE V. THE COMPUTATIONAL TIME WHEN *MIN_SUP* IS 0.3

	PSS kernel	PSS-RBF kernel
FSP	0.0315	0.0392
FCSP	0.0312	0.0369
FMSP	0.0222	0.0267

C. Experimental Designs

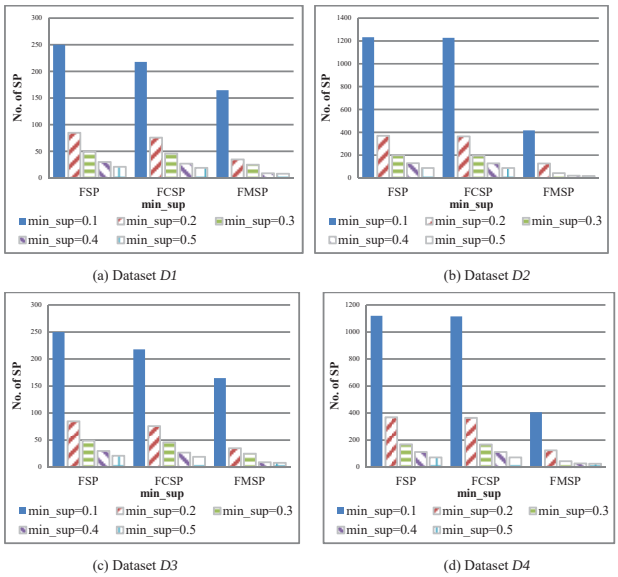
1) Numbers of Sequential Patterns

For dataset *D*1, the numbers of derived frequent sequential patterns (*FSP*), frequent closed sequential patterns (*FCSP*), and frequent maximal sequential patterns (*FMSP*) are summarized in Table VI where minimum support is adjusted from 0.1 to 0.5. *c*1, *c*2 and *c*3 in Table VI represent the number of sequential patterns in each class. It is clear that if minimum support is larger, the number of sequential patterns is less. Moreover, the number of frequent sequential patterns is larger than the numbers of other two types of sequential patterns.

TABLE VI. THE NUMBER OF SEQUENTIAL PATTERNS UNDER DIFFERENT MINIMUM SUPPORT IN *D*1

min_sup	*FSP*(*c*1,*c*2,*c*3)	*FCSP*(*c*1,*c*2,*c*3)	*FMSP*(*c*1,*c*2,*c*3)
0.1	250(82,81,87)	218(67,72,79)	165(31,62,72)
0.2	85(29,27,29)	76(25,23,28)	35(9,11,15)
0.3	49(19,16,14)	46(16,16,14)	25(4,9,12)
0.4	30(17,8,5)	27(14,8,5)	9(3,2,4)
0.5	21(12,6,3)	19(10,6,3)	8(3,3,2)

Fig. 3 shows the numbers of *FSP*, *FCSP*, *FMSP* for D1, D2, D3, and D4. In *D*2, when the value of minimum support is 0.1, the number of generated frequent sequential patterns and frequent closed sequential patterns are over 1000. On the contrary, while the value of minimum support is set as 0.5, the number of three types of sequential patterns are all less than 100. Furthermore, the number of frequent sequential patterns and frequent closed sequential patterns are identical when the value of minimum support is 0.5. In *D*4, the number of frequent sequential patterns and frequent closed sequential patterns in class 2 and class 3 are totally the same when minimum support is from 0.1 to 0.5. Moreover, when the value of minimum support is 0.4 and 0.5, the total number of frequent sequential patterns and frequent closed sequential patterns are identical. It makes the classification models build using the two sequential patterns are almost the same.



(a) Dataset *D1* (b) Dataset *D2*

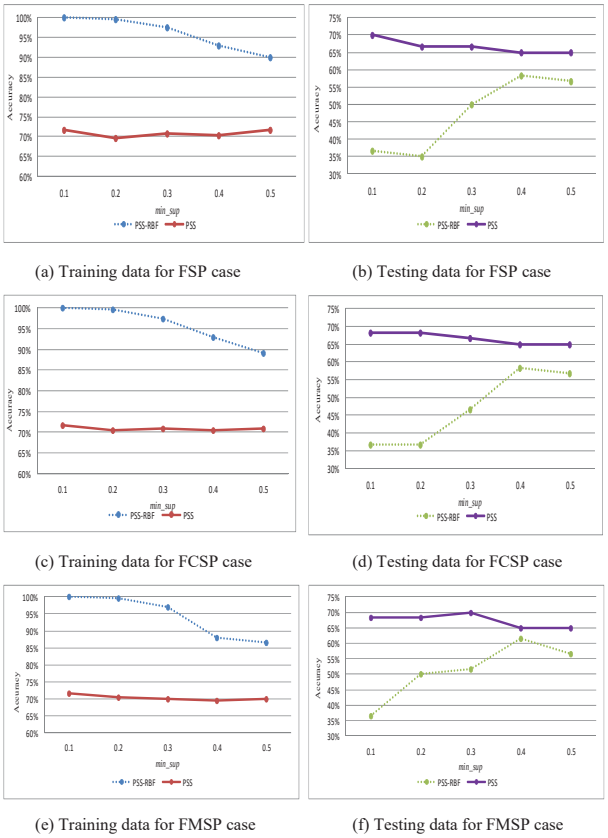
(c) Dataset *D3* (d) Dataset *D4*

Fig. 3. The numbers of sequential patterns for datasets *D*1 to *D4*.

According to the above figures, the total numbers of sequential patterns in *D*1 and *D*3 are close since the numbers of sequences in *D*1 and *D*3 are all 300. Similarly, the total number of sequential patterns in *D*2 and *D*4 are close since the numbers of sequences in *D*2 and *D*4 are all 30000. This makes the total numbers of sequence patterns in *D*2 and *D*4 are greater than the ones in *D*1 and *D*3. In addition, the sequence composition in *D*1 and *D*3 are 70%, 20% and 10% and 60%, 20% and 20% respectively. Similarly, the sequence composition in *D*2 and *D*4 are 70%, 20% and 10% and 60%, 20% and 20% respectively. However, it is observed that the number of sequential patterns will not be affected by the characteristics of the sequence composition as mentioned in Table I.

2) Comparison between the Models Using PSS and PSS-RBF Kernels

Since the experiment result for *D*1 to *D*4 show the similar trend, only the comparison result of *D*1 are reported. Fig. 4 shows the comparison result when *FSP*, *FCSP* and *FMSP* are applied respectively. As shown in Fig. 4(a), for training data of *D*1, the accuracy of the model using PSS-RBF kernel is always higher than the model using PSS kernel for *FSP* case. However, for testing data of *D*1, the accuracy of model with using PSS-RBF kernel is lower than the one using PSS kernel in Fig. 4(b). The reason is that the RBF function makes the model overtrained so that the training accuracy of model is higher and the testing accuracy of model is lower. Similarly, the accuracy of model with *FCSP* and *FMSP* cases using PSS and PSS-RBF kernels are shown in Fig. 4(c) to 4(f).



(a) Training data for FSP case (b) Testing data for FSP case

(c) Training data for FCSP case (d) Testing data for FCSP case

(e) Training data for FMSP case (f) Testing data for FMSP case

Fig. 4. The testing accuracy of model for different patterns using PSS and PSS-RBF kernels under different *min_sup*

The computational time of training classification model and computing accuracy using *FSP* is shown in Fig. 5. It is clearly that when minimum support is smaller, the computational time is longer. Furthermore, the computational time of using PSS kernel is less than using PSS-RBF kernel. The figure shows that SVM classification model using PSS kernel employs less time to achieve the higher accuracy. Therefore, the SVM classification model using PSS kernel is more efficient than the one using PSS-RBF kernel. Similar findings can be found for the model using *FCSP* and *FMSP*.

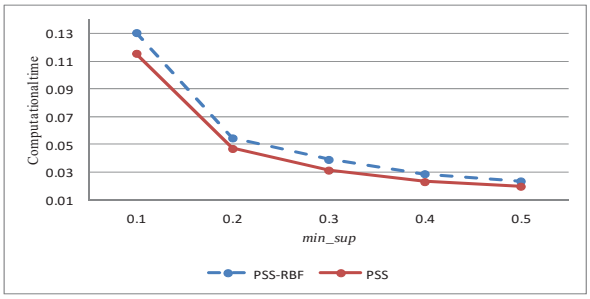


Fig. 5. The computational time of models using PSS-RBF and PSS kernels for *FSP* case

3) Comparison between the Models using PSS kernel and Spectrum Kernel

In this section, the classification model using spectrum kernel is used to compare with the model using the proposed PSS kernel. In spectrum kernel, *k*-mers where *k* is set as 2 and 3 is used as reference sequences in this study. When *k* is set as 2 (2-mers), the number of reference sequence is 12^2=144 since the number of the total items is 12. Similarly, if *k* is 3 (3-mers), the number of reference sequence is 12^3=1728.

Table VII shows the accuracy comparison for the models using PSS and spectrum kernels for *D*1. For training data, the accuracy of the proposed model is higher than the one using 2-mers and 3-mers. Similarly, the accuracy of the proposed model is higher than the one using 2-mers and 3-mers. Compared to spectrum kernel, the model using PSS can generate the highest testing accuracy and good training accuracy at the same time. When the value of minimum support is 0.1, the number of reference sequences for the model using *FSP* is 250. The number of reference sequences using PSS is more than the ones 2-mers, so the computational time using PSS is longer than the ones using 2-mers. On the contrary, the number of reference sequences using PSS is less than the ones using 3-mers, so the computational time using PSS is shorter than the ones using 3-mers.

TABLE VII. ACCURACY COMPARISON OF MODELS USING PSS AND SPECTRUM KERNELS IN *D*1

	Training data	Testing data	Number of reference sequences	Computational time
PSS	71.667%	70%	250	0.1157
2-mers	65.833%	60%	144	0.0623
3-mers	70.833%	56.667%	1728	0.5558

Table VIII shows the accuracy comparison of models using PSS and spectrum kernels for *D*2. The table clearly exhibits the accuracy of models using PSS is better than the ones using other kernels. Although the accuracy of the model using PSS is slightly higher than the one using 3-mers, the computational time for the model using PSS is significantly less than using the one using 3-mers. It represents the proposed PSS kernel is very efficient to achieve the highest accuracy for training and testing data in *D*2.

TABLE VIII. ACCURACY COMPARISON OF MODELS USING PSS AND SPECTRUM KERNEL IN *D*2

	Training data	Testing data	Number of reference sequences	Computational time
PSS	67.317%	66.467%	18	98.3888
2-mers	65.158%	64.767%	144	249.6072
3-mers	66.608%	66.383%	1728	1677.1940

V. CONCLUSIONS

The sequence databases are commonly seen in our daily life, such as protein sequences, biological sequences, transaction sequences and so on. SVMs have been are widely applied for sequence classification due to they provide better

accuracy and more elastic performance. In SVMs, kernel method is one of the most important key to achieve high classification accuracy. Previous researchers proposed many kernel methods to sequence classification. Among them, the kernel using *k*-mers such as spectrum kernel and mismatch kernel is popular to resolve the sequence classification since *k*-mers provide a simple way to construct the reference sequence. However, if the number of symbols is too large, many worthless features in the kernel using *k*-mers will be involved

Instead of using *k*-mers, this paper proposed a pairwise sequence similarity (PSS) kernel that takes three sequential patterns which are *FSP*, *FCSP* and *FMSP* as reference sequences. Each pairwise sequence similarity score vector is computed by edit distance algorithm between sequences and reference sequence vector before building the SVM classifiers. Therefore, the major contributions of this research includes the proposal of a new PSS kernel using sequential patterns as reference sequences instead of the traditional *k*-mers. Moreover, the SVM classification model using PSS kernel can get the higher accuracy and more efficient classification.

The following suggestion is made to improve this method further. Currently, PSS score vectors are computed by EDA. Further study can use different sequence alignment approach algorithms, such as Needleman-Wunsch algorithm, Smith–Waterman algorithm and so on for evaluating the similarity between sequences and patterns. In addition, the cost parameter in EDA can be modified. In this research, the cost for deletion, exchange, and insertion are all set as 1. Those cost might affect the comparison result. In order to increase the classification accuracy, each sequence can be added a weight to represent the importance of each patterns. The higher weight reveals the pattern is more valuable. Finally, the proposed method can be applied not only to protein sequences or biological sequences but also to transaction sequences, travel route sequences, handwritten analysis and so on.

REFERENCES

[1] B. E. Boser, I. M. Guyon, and V. N. Vapnik, "A training algorithm for optimal margin classifiers," Proceedings of the 5th Annual Workshop on Computational Learning Theory, pp. 144-152, 1992.

[2] V. N. Vapnik, Statistical Learning Theory, 1998.

[3] B. Vanschoenwinkel and B. Manderick, "Appropriate kernel functions for support vector machine learning with sequences of symbolic data," Lecture Notes in Computer Science, vol. 3635, pp. 256-280, 2005.

[4] T. Howley and M. G. Madden, "The genetic kernel support vector machine: description and evaluation," Artificial Intelligence Review, vol. 24, no. 3-4, pp. 379-395, 2005.

[5] T. Jaakkola, M. Diekhans, and D. Haussler, "A discriminative framework for de¬tecting remote protein homologies," Journal of Computational Biology, vol. 7, no. 1-2, pp. 95-114, 2000.

[6] C. S. Leslie, E. Eskin, and W. S. Noble, "The spectrum kernel: a string kernel for SVM protein classification," Proceedings of Pacific Symposium on Biocomputing, pp. 566-575, 2002.

[7] C. S. Leslie, E. Eskin, A. Cohen, J. Weston, and W. S. Noble, "Mismatch string kernels for discriminative protein classification," Bioinformatics, vol. 20, no. 4, pp. 467-476, 2004.

[8] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins, "Text classification using string kernels," The Journal of Machine Learning Research, vol. 2, pp. 419-444, 2002.

[9] W. M. Campbell, J. R. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek, "High-level speaker verification with support vector machines," Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 1, pp. 1-73, 2004.

[10] W. M. Campbell, J. P. Campbell, D. A. Reynolds, D. A. Jones, and T. R. Leek, "Phonetic speaker recognition with support vector machines," Advances in Neural Information Processing Systems, vol. 16, pp. 1377-1384, 2004.

[11] W. M. Campbell, J. P. Campbell, T. P. Gleason, D. A. Reynolds, and W. Shen, "Speaker verification using support vector machines and high-level features," IEEE Transactions on Audio, Speech, and Language Processing, vol. 15, no. 7, pp. 2085-2094, 2007.

[12] R. Agrawal and R. Srikant, "Mining sequential patterns," Proceedings of the 11th International Conference on Data Mining, pp. 3-14, 1995.

[13] J. Han, J. Pei, B. Mortazavi-Asl, H. Pinto, Q. Chen, U. Dayal, and M. C. Hsu, "Prefixspan: mining sequential patterns efficiently by prefix-projected pattern growth," Proceedings of the 17th International Conference on Data Engineering, pp. 215-224, 2001.

[14] X. Yan, J. Han, and R. Afshar, "CloSpan: mining closed sequential patterns in large datasets," Proceedings of the SIAM International Conference on Data Mining, pp. 166-177, 2003.

[15] P. Fournier-Viger, C. W. Wu, A. Gomariz, and V. S. Tseng, "VMSP: efficient vertical mining of maximal sequential patterns," Advances in Artificial Intelligence, pp. 83-94, 2014.

[16] C. W. Hsu and C. J. Lin, "A comparison of methods for multi-class support vector machines," IEEE transactions on Neural Networks, vol. 13, no. 2, pp. 415-425, 2002.

[17] R. K. Eichelberger and V. S. Sheng, "Does one-against-all or one-against-one improve the performance of multiclass classifications?" 27th AAAI Conference on Artificial Intelligence, pp. 1609-1610, 2013.

[18] E. L. Allwein, R. E. Schapire, and Y. Singer, "Reducing multiclass to binary: a unifying approach for margin classifiers," Journal of Machine Learning Research, vol. 1, pp. 113-141, 2001.

[19] J. Milgram, M. Cherier, and R. Sabourin, "One-against-one or one-against-all: which one is better for handwriting recognition with SVMs?," Proceedings of the 10th International Workshop on Frontiers in Handwriting Recognition, 2006.