# Early Prediction on Time Series: A Nearest Neighbor Approach*

**Zhengzheng Xing**
School of Computing Science
Simon Fraser University
zxing@cs.sfu.ca

**Jian Pei**
School of Computing Science
Simon Fraser University
jpei@cs.sfu.ca

**Philip S. Yu**
Department of Computer Science
University of Illinois at Chicago
psyu@cs.uic.edu

## Abstract

In this paper, we formulate the problem of early classification of time series data, which is important in some time-sensitive applications such as health-informatics. We introduce a novel concept of MPL (Minimum Prediction Length) and develop ECTS (Early Classification on Time Series), an effective 1-nearest neighbor classification method. ECTS makes early predictions and at the same time retains the accuracy comparable to that of a 1NN classifier using the full-length time series. Our empirical study using benchmark time series data sets shows that ECTS works well on the real data sets where 1NN classification is effective.

## 1 Introduction

Early classification of time series data is critical in some time-sensitive applications such as health-informatics. For example, a retrospective study of the clinical data of infants admitted to a neonatal intensive care unit [Griffin and Moorman, 2001] found that the infants, who were diagnosed with sepsis disease, had abnormal heart beating time series patterns 24 hours preceding the diagnosis. Monitoring the heart beating time series data and classifying the time series data as early as possible may lead to earlier diagnosis and effective therapy.

However, early classification of time series data is very challenging. An early classifier is expected to meet two requirements. First, an early classifier should be able to affirm the earliest time of reliable classification so that the early predictions can be used for further actions. Second, an early classifier should retain an accuracy comparable to that of a classifier using the full length time series or some user-specified accuracy threshold.

In this paper, we tackle the problem of early classification on time series data. We introduce a novel concept of MPL (Minimum Prediction Length) and develop ECTS ( Early Classification on Time Series), an effective 1-nearest neighbor classification method which makes prediction early and at the same time retains an accuracy comparable to that of a 1NN classifier using the full-length time series. Our empirical study using benchmark time series data sets shows that ECTS works well where 1NN classification is effective.

## 2 Problem Definition and Related Work

A *time series* $s$ is a sequence of pairs $(timestamp, value)$. The data values are ordered in timestamp ascending order. We assume that all timestamps take positive integer values. We denote by $s[i]$ the value of time series $s$ at timestamp $i$.

To keep our discussion simple, in this paper, we assume that all time series in question are of length $L$, i.e., each time series $s$ has a value $s[i]$ at timestamp $1 \leq i \leq L$. $L$ is called the *full length* of the time series. In general, we can use techniques like dynamic time warping [Myers and Rabiner, 1981] to align time series of different lengths.

For a time series $s$ of length $L$, $s[i,j] = s[i]s[i+1]\cdots s[j]$ $(1 \leq i < j \leq L)$ is the *subsequence* at timestamp interval $[i,j]$. Subsequence $s[1,l]$ $(l \leq L)$ is the *length-$l$ prefix* of $s$.

For two time series $s$ and $s'$, $dist(s,s')$ denotes the *distance* between them. In this paper, we use the Euclidean distance $dist(s,s') = \sqrt{\sum_{i=1}^{L}(s[i]-s'[i])^2}$, which is a simple yet effective and popularly adopted choice.

The set of all possible time series of length $L$ is $\mathcal{R}^L$ and is called the *full-length space*, where $\mathcal{R}$ is the set of real numbers. The *prefix space of length-$l$*, denoted by $\mathcal{R}^l$, is the set of length-$l$ prefixes of all possible time series.

In time series classification, a *training set* $T$ contains a set of time series and a set of *class labels* $\mathcal{C}$ such that each time series $t \in T$ carries a class label $t.c \in \mathcal{C}$. The *time series classification problem* is to learn from $T$ a classifier $C : \mathcal{R}^L \rightarrow \mathcal{C}$ such that for any time series $s$, $C$ predicts the class label of $s$ by $C(s)$. The performance of a classifier is often evaluated using a *test set* $T'$, which is a set of time series such that each time series $t' \in T'$ also carries a class label $t'.c \in \mathcal{C}$. The *accuracy* of a classifier $C$ is the percentage that the class labels generated by $C$ match those carried by the time series in the test set, that is, $Accuracy(C,T') = \frac{|\{C(t')=t'.c|t' \in T'\}|}{|T'|}$. Often, we want the classifier $C$ as accurate as possible.

For a time series $s$, an early classifier $C$ can identify an integer $l_0$ and make classification based on $s[1,l_0]$. An early classifier is *serial* [Xing *et al.*, 2008] if $C(s[1,l_0]) = C(s[1,l_0+i])$ for any $i > 0$. In other words, $C$ can classify $s$ based on only the prefix $s[1,l_0]$, and the classification

*The research is supported in part by an NSERC Discovery grant, an NSERC Discovery Accelerator grant, and a Simon Fraser University Trust Endowment Fund.

remains the same by using any longer prefixes. An early classifier is preferred to be serial so that the early classification is reliable and consistent. The minimum length $l_0$ of the prefix based on which $C$ makes the prediction is called the *cost* of the prediction, denoted by $Cost(C, s) = l_0$. Trivially, for any finite time series $s$, $Cost(C, s) \leq |s|$. The *cost* of the prediction is $Cost(C, T') = \frac{1}{|T'|} \sum_{t' \in T'} Cost(C, t')$.

Among many methods that can be used in time series classification, the 1-nearest neighbor (1NN) classifier has been found often accurate in practice [Keogh and Kasetty, 2002; Wei and Keogh, 2006]. The 1NN classification method is parameter-free and does not require feature selection and discretization. Theoretically, Cover and Hart [1967] showed that the error rate of the 1NN classifier is at most twice that of the optimal Bayes probability when an infinite sample set is used.

Due to the effectiveness and the simplicity of the 1NN classifier on time series data, in this paper, we focus on extending the 1NN classifier for early classification on time series data. We use the 1NN classifier on full length time series as the baseline for comparison. Ideally, we want to build a classifier which is as accurate as the baseline method and minimizes the expected prediction cost.

In our previous work [Xing *et al.*, 2008], we formulated the early classification problem on symbolic sequence data. The major idea is to mine symbolic sequential patterns which have high utility in early prediction, and then form classification rules or decision trees using those patterns. Time series are numeric. To use our symbolic methods, time series have to be discretized properly. However, appropriate discretization often heavily relies on good background knowledge. Moreover, affected by the discretization granularity, the discretization-based methods may lose important information in time series data. Our previous study [Xing *et al.*, 2008] shows that the symbolic methods do not work well on numeric time series data. Thus, early prediction on time series data, a type of data prevalent in time-sensitive applications, remains open at large.

To the best our knowledge, [Rodriguez and Alonso, 2002] is the only existing study mentioning early classification on time series data, which refers to classification when only a partial time series is available. It focuses on how to make prediction on partial information but does not attempt to make reliable prediction using minimal partial information.

Generally, most existing sequence classification methods transform a sequence into a set of features and then apply conventional classification methods on the features. For example, Lesh *et al.* [1999] proposed a criterion for selecting features for sequence classification. Nanopoulos *et al.* [2001] extracted statistical features such as mean and deviation from time series, and built a neural network on the features to classify time series. Eads *et al.* [2005] employed grammar-guided feature extraction and proposed a SVM classifier.

Different from feature-based classifiers, instance based classifiers [Keogh and Kasetty, 2002; Wei and Keogh, 2006; Xi *et al.*, 2006], such as the 1NN classifier, make predictions based on the similarities between the time series to be classified and the ones in the training set. The choice of distance metric is critical to the performance of 1NN classifiers. The Euclidean distance is shown [Keogh and Kasetty, 2002; Wei and Keogh, 2006] superior on accuracy comparing to other similarity measurements. Xi *et al.* [2006] showed that, on small data sets, elastic measures such as dynamic time warping (DTW) can be more accurate than the Euclidean distance. However, on large data sets, the accuracy of elastic measures converges with the Euclidean distance. In this paper, we focus on extending the 1NN classifier with the Euclidean distance to achieve early classification. However, our principle can be applied to other instance-based methods using different distance metrics.

| t-id | time series | class |
|---|---|---|
| $t_1$ | $(1,1,1)$ | $c_1$ |
| $t_2$ | $(2,1,2)$ | $c_1$ |
| $t_3$ | $(5,2,1)$ | $c_1$ |
| $t_4$ | $(6,1,2)$ | $c_1$ |
| $t_5$ | $(5,8,7)$ | $c_2$ |
| $t_6$ | $(5,9,9)$ | $c_2$ |
| $t_7$ | $(6,8,9)$ | $c_2$ |

Table 1: A training set $T$ as the running example.

## 3 1NN Early Classification

**Example 1.** *Consider a training set of time series $T$ in Table 1, where each time series, written as a sequence of values in timestamp ascending order, has a length $L = 3$.*

*To classify a time series $s_1 = (1, 2, 1)$, the 1NN classifier using the full length time series finds $t_1$ as the 1NN of $s_1$ in space $\mathcal{R}^3$, and outputs $c_1$, the class label of $t_1$.*

*The prediction on $s_1$ can be made much earlier, since $t_1$ is also the 1NN of prefixes $s_1[1] = (1)$ and $s_1[1, 2] = (1, 2)$. In other words, using the 1NN of the prefixes, we may make early prediction.* ■

For a time series $s$ and a training data set $T$, let $NN^l(s) = \arg\min_{t \in T}\{dist(s[1, l], t[1, l])\}$ be the set of the nearest neighbors of $s$ in $T$ in prefix space $\mathcal{R}^l$. In Example 1, $NN^1(s_1) = \{t_1\}$ and $NN^2(s_1) = \{t_1\}$.

In the full length space $\mathcal{R}^L$, using the 1NN classifier, a time series $s$ is classified by the dominating label in $NN^L(s)$. Consider prefix space $\mathcal{R}^{L-1}$. Let $T[1, L-1] = \{t[1, L-1] | t \in T\}$ be the set of length-$(L-1)$ prefixes of all time series in the training set $T$. If the time series in $NN^L(s)$ are still the nearest neighbors of $s[1, L-1]$ in $T[1, L-1]$, i.e., $NN^L(s) = NN^{L-1}(s)$, then we can use $NN^{L-1}(s)$ to make prediction on the class label of $s$ at timestamp $(L-1)$ without compromise in accuracy. This immediately leads to early prediction, as observed in Example 1. The question is when the data points of $s$ arriving in time ascending order, how we can know starting from which prefix length $l$, the nearest neighbors of $s$ will remain the same.

Generally, the set of length-$l$ prefixes $(1 \leq l \leq L)$ of the time series in $T$ is $T[1, l] = \{t[1, l] | t \in T\}$. For $t \in T$, the *set of reverse nearest neighbors in prefix space $\mathcal{R}^l$* of $t(1, l)$ is $RNN^l(t) = \{t' \in T \mid t \in NN^l(t')\}$. That is, $RNN^l(t)$ is the set of time series in $T$ that treat $t$ as its nearest neighbor. In Example 1, since $RNN^1(t_1) = RNN^2(t_1) = RNN^3(t_1) = \{t_2\}$, $RNN^1(t_2) = RNN^2(t_2) = RNN^3(t_2) = \{t_1\}$.

Suppose training set $T$ is a sufficiently large and uniform sample of the time series to be classified. For a time series $t \in T$, if there exists a timestamp $l < L$ such that $RNN^l(t) = RNN^{l+1}(t) = \cdots = RNN^L(t)$, then we can use $t$ to make prediction early at timestamp $l$ without loss in expected accuracy, since every time series $s$ which uses $t$ in $\mathcal{R}^L$ to predict the class label also likely has $t$ as the 1NN in prefix spaces $\mathcal{R}^l$, $\mathcal{R}^{l+1}$, ..., $\mathcal{R}^{L-1}$. In Example 1, $t_1$ can be used to make prediction at timestamp 1 since $RNN^1(t_1) = RNN^2(t_1) = RNN^3(t_1)$.

**Definition 1 (Minimum prediction length).** *In a training data set T with full length L, for a time series $t \in T$, the* minimum prediction length *(*MPL *for short) of t, $MPL(t) = k$ if for any $l$ ($k \leq l \leq L$), (1) $RNN^l(t) = RNN^L(t) \neq \emptyset$ and (2) $RNN^{k-1}(t) \neq RNN^L(t)$. Specifically, if $RNN^L(t) = \emptyset$, $MPL(t) = L$. We denote by $MPP(t) = t[1, MPL(t)]$ the* minimum prediction prefix *of t.* ■

**Example 2.** *In Example 1, $MPL(t_1) = 1$ and $MPP(t_1) = (1)$. Moreover, $MPL(t_2) = 1$, $MPL(t_3) = MPL(t_4) = 2$. $MPL(t_5) = MPL(t_6) = MPL(t_7) = 3$.* ■

Given a training set $T$, a simple *1NN early classification method* works as follows. We assume that the training set is a sufficiently large and uniform sample of the data to be classified. In the training phase, for each time series $t \in T$, we calculate the minimum prediction length $MPL(t)$. In the classification phase, for a time series $s$ to be classified, the values of $s$ arrive in timestamp ascending order. At timestamp $i$, we return the dominating class label of time series in $NN^i(s)$ that have a MPL at most $i$. If no such a time series is found, we cannot make reliable prediction at the current timestamp, and have to wait for more values of $s$.

**Example 3.** *In Example 1, it is easy to verify that $s_1 = (1, 2, 1)$ is classified as $c_1$ using $t_1$ at timestamp 1. Consider $s_2 = (6, 2, 3)$. $NN^1(s_2) = \{t_4, t_7\}$, but the MPLs of $t_4$ and $t_7$ are greater than 1. Thus, $s_2$ cannot be classified at timestamp 1. $NN^2(s_2) = \{t_3, t_4\}$. The MPLs of $t_3$ and $t_4$ are 2. Thus, $s_2$ can be assigned label $c_1$ at timestamp 2.* ■

The 1NN early classification method has two drawbacks. First, to make early prediction using a time series $t$ in a training set $T$, the RNNs of $t$ must be stable after timestamp $MPL(t)$. This requirement is often too restrictive and limits the capability of finding shorter prediction length. In Example 1, the RNNs of $t_5, t_6, t_7$ are not stable, and the MPLs of them are all 3. We cannot use those time series to classify $s_3 = (5, 8, 8)$ at timestamp 2.

Second, the 1NN early classification method may overfit a training set. The MPP of a time series is obtained by only considering the stability of its RNNs which often consist of a small number of time series. The learned $MPL(t)$ may not be long and robust enough to make accurate classification if the training set is not big enough or is not a uniform sample of the time series to be classified.

## 4 The ECTS Method

To overcome the drawbacks in the 1NN early classification method, we develop the ECTS method which extends the 1NN early classification method by finding the MPL for a cluster of similar time series instead of a single time series.

| Prefix space | Clusters |
|---|---|
| $\mathcal{R}^1$ | $S_1 = \{t_1, t_2\}$, $S_2 = \{t_3, t_4, t_5, t_6, t_7\}$ |
| $\mathcal{R}^2$ | $S_1 = \{t_1, t_2\}$, $S_2 = \{t_3, t_4\}$ $S_3 = \{t_5, t_6, t_7\}$ |
| $\mathcal{R}^3$ | $S_1 = \{t_1, t_2\}$, $S_2 = \{t_3, t_4\}$ $S_3 = \{t_5, t_6, t_7\}$ |

Table 2: The clusters in different prefix spaces.

**Example 4.** *We cluster the time series in the training set T in Table 1 in the full space $\mathcal{R}^3$ and the prefix spaces $\mathcal{R}^2$ and $\mathcal{R}^1$, as shown in Table 2.*

*Cluster $S_1$ is stable in all three spaces, and thus can be used at early classification as early as timestamp 1. Clusters $S_2$ and $S_3$ are stable at timestamps 2 and 3 and thus can be used as early as at timestamp 2.* ■

We need to address two issues. First, we need to use a clustering approach to obtain the clusters in the full-length space. Second, we need to compute the MPLs of clusters.

We adopt single link MLHC [Ding and He, 2005], an agglomerative hierarchical clustering method to cluster the training data set in full length space. It builds a hierarchical clustering tree level by level by merging all mutual nearest neighbor pairs of clusters at each level. Two clusters form a mutual nearest neighbor pair if they consider each other as the nearest neighbor. The distance between two clusters is measured by the minimum among all inter-cluster pair-wise distances. In the output hierarchical clustering tree (i.e., a dendrogram), a cluster represented by a leaf node is called a *leaf-cluster*, and a cluster represented by an internal node is called a *sub-cluster*. The whole training set is represented by the root, and thus is called the *root-cluster*.

A cluster $S$ is called *1NN consistent* [Ding and He, 2004] if for each object (a time series in our case) $o \in S$, the 1NN of $o$ also belongs to $S$. Immediately, we have the following.

**Lemma 1.** *The sub-clusters and the root-cluster generated by single link MLHC are 1NN consistent.* ■

If all time series in a sub-cluster $S$ carry the same label, $S$ is called a *discriminative cluster*. We denote by $S.c$ the common class label of the time series in $S$. A discriminative cluster can be used in classification.

**Example 5.** *$S_1$, $S_2$ and $S_3$ in space $\mathcal{R}^3$ in Table 2 are discriminative clusters and can be used in classification. For example, $s_3 = (5, 8, 8)$ finds $t_5 \in S_3$ as the 1NN in $\mathcal{R}^3$. $S_3.c = c_2$ can be assigned to $s_3$.* ■

To explore the potential of a discriminative cluster $S$ in early classification, we find the earliest prefix space in which $S$ is formed and becomes stable since then. The corresponding prefix length is the minimum prediction length of $S$. In a prefix space, we check if the 1NN consistency property holds for $S$ and the stability of the reverse neighbors of $S$.

For a sub-cluster $S$, in space $\mathcal{R}^l$ ($1 \leq l \leq L$), we define the *reverse nearest neighbors* of $S$ as $RNN^l(S) = \cup_{s \in S} RNN^l(s) \setminus S$. If $S$ is well separated from other clusters, $RNN^l(S)$ is empty. Often, some sub-clusters in a training set may not be well separated from others. In such a case, $RNN^l(S)$ is the boundary area of $S$.

**Definition 2 (MPLs of clusters).** *In a training data set T with full length L, for a discriminative cluster S, $MPL(S) =$*

*$k$ if for any $l \geq k$, (1) $RNN^l(S) = RNN^L(S)$; (2) $S$ is 1NN consistent in space $\mathcal{R}^l$; and (3) for $l = k-1$, (1) and (2) cannot be both satisfied.* ∎

**Example 6.** *For discriminative clusters $S_1$, $S_2$ and $S_3$, $MPL(S_1) = 1$, $MPL(S_2) = 2$ and $MPL(S_3) = 2$. Take $S_3$ as an example, in space $\mathcal{R}^3$ and $\mathcal{R}^2$, $S_3$ is 1NN consistent and $RNN^3(S_3) = RNN^2(S_3) = \emptyset$. In space $\mathcal{R}^1$, $S_3$ is not 1NN consistent. Thus, $MPL(S_3) = 2$.* ∎

In a discriminative cluster $S$, there may be another discriminative cluster $S' \subset S$. $MPL(S)$ may be longer or shorter than $MPL(S')$. Moreover, for a time series $t \in T$, $t$ itself is a leaf-cluster with $MPL(t)$ (Definition 1). Among all the discriminative or leaf clusters containing $t$, we can use the shortest MPL to achieve the earliest prediction using $t$.

As one drawback of the 1NN early classification method, the MPL obtained from a very small cluster may cause over-fitting. To avoid over-fitting, a user can specify a parameter *minimum support* to control the size of a cluster. We calculate the support of a cluster in a way that is aware of the population of the corresponding class.

**Definition 3 (Support).** *In a training set $T$, let $T_c = \{s \in T | s.c = c\}$ be the set of time series that have label $c$. For a discriminative cluster or a leaf cluster $S$ such that $S.c = c$, $Support(S) = \frac{|S|}{|T_c|}$.* ∎

We only use clusters passing the user-specified minimum support threshold for early prediction. The *ECTS* method works in two phases. In the *training phase*, given a training set $T$ with full-length $L$ and a *minimum support* threshold $p_0$, for a time series $t \in T$, let $SS = \{S | t \in S \wedge S \text{ is a discriminative or leaf cluster}\}$ be the set of discriminative or leaf clusters containing $t$. $MPL(t) = \min_{S \in SS, Support(S) \geq p_0} MPL(S)$. The training process is to compute $MPL(t)$ for all $t \in T$. The *classification phase* is the same as the 1NN early prediction method in Section 3.

Section 2 states that we want to build a classifier as accurate as the 1NN classifier using the full-length time series. The following result answers the requirement.

**Theorem 1.** *In a training data set $T$ of full length $L$, assuming for any $t \in T$ and $1 \leq l \leq L$, $NN^l(t)$ contains only one time series*[1]*, ECTS has the same leave-one-out accuracy on $T$ as the 1NN classifier using the full-length time series. Moreover, ECTS is a serial classifier on the time series in $T$.*

*Proof sketch.* Using Lemma 1, we can show that in any leave-one-out test, if ECTS assigns label $c$ to a time series $t \in T$ using prefix $t(1, l)$, the 1NN classifier assigns to $t$ the same class label using any longer prefix $t(1, k)$, where $l < k \leq L$. Limited by space, we omit the details here. ∎

The only remaining question is how to compute the sub-clusters and the MPLs. Given a training set $T$, we first pre-compute the nearest neighbors for each time series in $T$ in all prefix spaces. This pre-computation step takes time $O(|T|^2 \cdot L)$ where $L$ is the full length.

[1]If multiple 1NNs exist, we can select the 1NN of the smallest index .

```
Input: a training data set T;
Output: MPL(t) for each $t \in T$;
Method:
1:  pre-compute 1NNs for each $t \in T$ in all prefix spaces;
2:  compute the MPLs of leaf clusters and update the MPL
    for each $t \in T$;
3:  $n = |T|$, the number of time series in $T$;
4:  while $n > 1$
5:     compute the mutual nearest neighbor pairs;
6:     for each mutual nearest neighbor pair $(S_1, S_2)$
7:        merge $S_1$ and $S_2$ into a parent cluster $S$, $n = n - 1$;
8:        if all time series in $S$ carry the same label then
9:           compute the MPL of $S$;
10:          update the MPL for each time series in $S$;
          end if
       end for
11:    if no new discriminative clusters are generated in this
       round then break;
    end while
```

Figure 1: The algorithm of the training phase in ECTS.

Then, we apply single link MLHC [Ding and He, 2005] to space $\mathcal{R}^L$, which takes time $O(|T|^2)$. For each discriminative sub-cluster $S$, using the pre-computed 1NN information in the prefix spaces, we check the 1NN consistency of $S$ and the stability of $RNN^l(S)$ for $l < L$ in the value $l$ descending order, until $MPL(S)$ is determined. The complexity of this step is $O(|T|^3 \cdot L)$. The algorithm of the training phase is shown in Figure 1.

Is learning different MPLs for different time series necessary? Can we just learn a fixed MPL for all time series in a training set? Let us consider the following simple early classification method called *1NN fixed*. Given a training set $T$ of full length $L$, we calculate the 1NN classification accuracy $p$ in the full space $\mathcal{R}^L$. Then, we check the prefix spaces $\mathcal{R}^{L-1}, \mathcal{R}^{L-2}, \ldots$ until prefix space $\mathcal{R}^k$ such that the accuracies in spaces $\mathcal{R}^{L-1}, \ldots, \mathcal{R}^{k+1}$ are at least $p$, and the accuracy in space $\mathcal{R}^k$ is lower than $p$. We use $(k+1)$ as the MPL for all time series. That is, in classification, we always read the length-$(k+1)$ prefix of a time series $s$ to be classified, and find the 1NNs of $s$ among the length-$(k+1)$ prefixes of the time series in $T$ to classify $s$.

Interestingly, 1NN fixed is a simplified special case of ECTS in 2-class situations under the assumption that each class forms one discriminative cluster in the full length space $\mathcal{R}^L$. However, since 1NN fixed does not consider the different early classification capabilities of the clusters in the hierarchy, it may use longer prefixes for classifications. Furthermore, when there are multiple classes or multiple large discriminative clusters, the 1NN fixed method may not work well since it requires the overall accuracy to be high and cannot identify clusters which are separated from other clusters earlier than the overall accuracy is satisfied.

## 5 Experimental Results

The UCR time series archive [Keogh *et al.*, 2006] provides 23 time series data sets which are widely used to evaluate time

| Dataset | | **ECTS** | Early 1NN | 1NN Fixed | **Full 1NN** | SCR |
|---|---|---|---|---|---|---|
| **Wafer** | Accuracy | 99.08%($-0.47\%$ ) | 99.16%(-0.39 %) | 99.32%(-0.23 %) | 99.55% | 93% (-6.58 %) |
| 2 classes | Ave. len. | 67.39 (44.34%) | 122.05 (80.30%) | 110 (72.37%) | 152 | 55.36 (36.42%) |
| 1000 training samples | Train. time | 152.90 sec | 3.22 sec | 1.58 sec | 0 sec | 164.59 sec |
| 6174 testing samples | Class. time | 0.053 sec | 0.103 sec | 0.004 sec | 0.004 sec | 0.204 sec |
| **Gun-Point** | Accuracy | 86.67% ($-5.1\%$) | 87.3%(-4.41 %) | 91.3%(-0.03 %) | 91.33% | 62.67%($-31.36\%$) |
| 2 classes | Ave. len. | 70.387 (46.92%) | 107.24 (71.49%) | 140 (92.67%) | 150 | 116.17 (77.45%) |
| 50 training samples | Train. time | 0.39 sec | 0.09 sec | <0.01 sec | 0 sec | 0.86 sec |
| 150 testing samples | Class. time | 0.002 sec | 0.004 sec | 0.002 sec | 0.002 sec | 0.11 sec |
| **Two Patterns** | Accuracy | 86.48% ($-4.97\%$) | 87.25%(-4.12 %) | 90.72%( -0.8%) | 91% | 94.75% (+4.12% ) |
| 4 classes | Ave. len. | 111.097 (86.79%) | 113.24 (88.5%) | 124 (96.88%) | 128 | 86.13(67.29 %) |
| 1000 training samples | Train. time | 48.76 sec | 2.18 sec | 1.34 sec | 0 sec | 65.23 sec |
| 4000 testing samples | Class. time | 0.101 sec | 0.077 sec | 0.003 sec | 0.003 sec | 0.125 sec |
| **ECG** | Accuracy | 89% ($+1.17\%$) | 89%(+1.17 %) | 89%(+1.17 %) | 88% | 73%($-17.05\%$) |
| 2 classes | Ave. len. | 74.04 (77.13%) | 83.12 (86.58%) | 92 (92.71%) | 96 | 37.5 (39.06%) |
| 100 training samples | Train. time | 0.83 sec | 0.1 sec | 0.02 sec | 0 sec | 4.22 sec |
| 100 testing samples | Class. time | 0.004 sec | 0.004 sec | 0.001 sec | 0.001 sec | 0.062 sec |
| **Synthetic Control** | Accuracy | 89% ($+1.17\%$) | 88.33%(+0.38 %) | 88%(+0 %) | 88% | 58.33%($-33.72\%$) |
| 6 classes | Ave. len. | 53.98 (89.97%) | 55.09 (91.82%) | 60 (100%) | 60 | 29.39 (50%) |
| 300 training samples | Train. time | 4.64 sec | 0.12 sec | 0.06 sec | 0 sec | 21.09 sec |
| 300 testing samples | Class. time | 0.004 sec | 0.004 sec | 0.001 sec | 0.001 sec | 0.02 sec |
| **OliveOil** | Accuracy | 90% ($+3.8\%$) | 90%(+3.8 %) | 83.33%(-3.92%) | 86.7% | 36.7%($-57.68\%$) |
| 4 classes | Ave. len. | 497.83 (82%) | 526 (92.28%) | 406 (71.23%) | 570 | 500 (87.72%) |
| 30 training samples | Train. time | 0.22 sec | 0.08 sec | 0.02 sec | 0 sec | 2.03 sec |
| 30 testing samples | Class. time | 0.058 sec | 0.043 sec | 0.006 sec | 0.006 sec | 0.016 sec |
| **CBF** | Accuracy | 85.2% ($+0\%$) | 86.89%(+1.98 %) | 83.2%(-2.35%) | 85.2% | 55.22% (-35.18%) |
| 3 classes | Ave. len. | 91.73 (71.5%) | 103.20 (80.63%) | 54 (42.19%) | 128 | 46 (35.93 %) |
| 30 training samples | Train. time | 0.09 sec | 0.02 sec | <0.01 sec | 0 sec | 0.24 sec |
| 900 testing samples | Class. time | 0.001 sec | 0.001 sec | 0.001 sec | 0.001 sec | 0.015 sec |

Table 3: Results on seven datasets from UCR Time Series Archive

series clustering and classification algorithms. In each data set, the time series have a fixed length. Each data set contains a training set and a testing set. The 1NN classification accuracies using the Euclidean distance on the testing sets are provided as well [Keogh *et al.*, 2006].

Table 3 lists the results on all the 7 data sets in the archive where the full-length 1NN classifier using Euclidean distance achieves an accuracy of at least $85\%$. The 1NN classifier can be regarded effective on those data sets. The seven data sets cover cases of 2-class and more-than-two-class cases.

Table 3 compares $5$ methods. We use the 1NN classifier using the full length (denoted by full 1NN) as the baseline. In addition to ETCS, we also report the results of the 1NN early classification method (Section 3, denoted by 1NN Early), the 1NN fixed method introduced at the end of Section 4, and the SCR method [Xing *et al.*, 2008]. ECTS uses only an optional parameter, *minimum support*, to avoid overfitting. The results of ECTS in Table 3 are obtained by setting *minimum support*$=0$. All the experiments were conducted using a PC computer with an AMD 2.2GHz CPU and 1GB main memory. The algorithms were implemented in C++ using Microsoft Visual Studio 2005.

On data sets Wafer and Gun-point, the average prediction lengths ($Cost(C, T')$ defined in Section 2) of ECTS are shorter than half of the full lengths. On the other five data sets, the average prediction lengths of ECTS are from $71\%$ to $89\%$ of the full lengths.

Except for data sets Gun-point and Two-patterns, ECTS has an accuracy almost the same or even slightly better than that obtained by the full-length 1NN method. On Gun-point and Two-patterns, ECTS is about $5\%$ lower in accuracy. As explained before, when *minimal support*$=0$, ECTS may over fit a training set and thus may slightly lose accuracy.

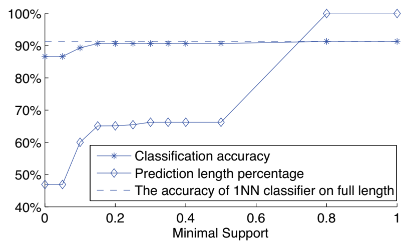

Figure 2: Accuracy and ave. length vs. minimum support.

By increasing *minimum support*, overfitting can be reduced. Figure 2 shows the effect on the Gun-point data set. When *minimum support* increases, the accuracy of ECTS approaches the accuracy of the full length 1NN classifier quickly. As the tradeoff, the average prediction length increases, too. Similar results are observed on the Two-patterns data set. Limited by space, we omit the details here.

The above results clearly show that ECTS can achieve early classification and can retain an accuracy comparable to that of the full length 1NN classifier. Using parameter *minimum support*, ECTS can reduce overfitting effectively.

On all the 7 data sets, ECTS and the early 1NN method achieve very similar accuracies. ECTS always achieves a shorter average prediction length. This result confirms that finding MPLs through clusters helps to obtain shorter MPLs.

On data sets Wafer, ECG and Synthetic-control, ECTS using *minimum support*$= 0$ achieves a shorter average prediction length than the 1NN fixed method. The two methods have similar accuracies. Especially, for the 6 classes synthetic control data set, the 1NN fixed method has to use the full length. ECTS uses a shorter prediction length. The saving mainly comes from the class cyclic, which is classified using an average length of 45. To handle multi-class situations, the 1NN fixed method cannot classify one class earlier if the other classes are mixed together in the prefix spaces.

On data set Gun-point, by setting *minimum support*$= 15\%$ to reduce overfitting, ECTS obtains an accuracy comparable to the 1NN fixed method, but uses a remarkably shorter average prediction length. Similar results are observed on data set Two-patterns.

On data sets OliveOil and CBF, interestingly, the 1NN fixed method is less accurate than ECTS but can obtain shorter average prediction length. For example, on data set OliveOil, ECTS obtains an accuracy of $90\%$ and 1NN fixed method makes only $83.33\%$. The 1NN fixed method obtains an average prediction length of 406 and ECTS gives a length of $497.63$. By analyzing the training set of 30 time series, we find that, training samples 7 and 9 are the cause of the dropping accuracy in the 1NN fixed method, which means the MPLs (406) of those two samples learned by the 1NN fixed method are not long enough to make accurate classification. In ECTS, the learned MPLs vary from 117 to 570 for the training samples. For samples 7 and 9, the learned MPLs are 567 and 570, respectively. Why does ECTS learn longer MPLs for samples 7 and 9? In the full length space, training sample 9 has an empty RNN set. The RNN set of training sample 7 consists of samples from two classes. Those RNN sets suggest that samples 7 and 9 are likely on the decision boundary. In contrast to the 1NN fixed method, ECTS can find longer MPLs for samples likely on the decision boundary to reduce the possible misclassification led by such a sample.

We also compare ECTS with our previous symbolic method SCR [Xing *et al.*, 2008]. Since SCR can only handle discrete values, $k$-means ($k = 3$) is used to discretize values in the time series into 3 values. SCR requires a parameter, expected classification accuracy, which is set to the full length 1NN accuracy. The other parameter of SCR, *minimal support*, is set to 0. Although SCR sometimes uses a shorter average prediction length, the accuracies are far away from the expected values. Comparing to SCR, ECTS makes early classification reliable in accuracy.

In terms of efficiency, ECTS is faster than SCR in training. The early 1NN method, the 1NN fixed method and the full 1NN method are substantially faster than ECTS and SCR in training due to their very simple model construction tasks. All the methods have very fast classification runtime.

## 6 Conclusions

In this paper, we develop ECTS, a method for early classification on time series data. ECTS makes early classification and retains an accuracy comparable to that of the full length 1NN method. As future work, it is interesting to extend ECTS for streaming data.

## References

[Cover and Hart, 1967] T. Cover and P. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[Ding and He, 2004] C. Ding and X. He. K-nearest-neighbor consistency in data clustering: incorporating local information into global optimization. In *Proc. the 2004 ACM symp. on Applied computing*, pages 584–589, 2004.

[Ding and He, 2005] C. Ding and X. He. Cluster aggregate inequality and multi-level hierarchical clustering. In *Proc. PKDD 2005*, pages 224–231.

[Eads *et al.*, 2005] D. Eads, K. Glocer, S. Perkins, and J. Theiler. Grammar-guided feature extraction for time series classification. In *NIPS '05*, 2005.

[Griffin and Moorman, 2001] M. P. Griffin and J. R. Moorman. Toward the early diagnosis of neonatal sepsis and sepsis-like inllness using noval heart rate analysis. *PEDIATRICS*, 107(1):97–104, 2001.

[Keogh and Kasetty, 2002] E. Keogh and S. Kasetty. On the need for time series data mining benchmarks: a survey and empirical demonstration. In *KDD '02: Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 102–111, 2002.

[Keogh *et al.*, 2006] E. Keogh, X. Xi, L. Wei and C. A. Ratanamahatana. The UCR time series classification/clustering page, 2006. `http://www.cs.ucr.edu/~eamonn/time_series_data/`.

[Lesh *et al.*, 1999] N. Lesh, M. J. Zaki, and M. Ogihara. Mining features for sequence classification. In *Proc. KDD 1999*, pages 342–346.

[Myers and Rabiner, 1981] C. S. Myers and L. R. Rabiner. A comparative study of several dynamic time-warping algorithms for connected word recognition. *The Bell System Technical Journal*, 60(7):1389–1409, 1981.

[Nanopoulos *et al.*, 2001] A. Nanopoulos, R. Alcock and Y. Manolopoulos. Feature-based classification of time-series data. *Int'l Journal of Computer Research*, 10:49–61, 2001.

[Rodriguez and Alonso, 2002] C. J. González and J. J. Rodríguez. Boosting interval-based literals: Variable length and early classification. In *ECAI'02 Workshop on Knowledge Discovery from (Spatio-) Temporal Data*, pages 51–62, 2002.

[Wei and Keogh, 2006] L. Wei and E. Keogh. Semi-supervised time series classification. In *Proc. KDD 2006*, pages 748–753.

[Xi *et al.*, 2006] X. Xi and E. Keogh, C. Shelton, L. Wei and C. A. Ratanamahatana. Fast time series classification using numerosity reduction. In *Proc. ICML 2006*, pages 1033–1040.

[Xing *et al.*, 2008] Z. Xing, J. Pei, G. Dong, and P. S. Yu. Mining sequence classifiers for early prediction. In *Proc. of the 2008 SIAM Int'l Conf. on Data Mining*, pages 644–655, 2008.