Instance Reduction for Time Series Classification by Exploiting Representative Characteristics using k-means

Vo Thanh Vinh, Hien T. Nguyen$^{(\boxtimes)}$, and Tin T. Tran

Faculty of Information Technology, Ton Duc Thang University, Ho Chi Minh City, Vietnam
{vothanhvinh, hien, trantrungtin}@tdt.edu.vn

Abstract. 1-Nearest Neighbor has been endorsed an efficient method for time series classification as it outperforms more advanced classification algorithms in most cases. However, the time and space efficiency of this method depend on the number of instances in a training set. In order to improve its running time and space using, one can apply an approach called instance reduction. This approach reduces the training set size by choosing the best instances and using only them during classification of new instances. However, the high-dimensional characteristic of time series is a challenge for many mining tasks including instance reduction. Due to this challenge, only two instance reduction methods have been proposed: Naïve Rank Reduction and INSIGHT. In this work, we propose a new approach to instance reduction using K-means. Our method can select a reasonable set of instances so that it can preserve the representative characteristics and keep the generalization of the original training set. We conduct experiments to evaluate our method on 46 public datasets from the UCR Classification Archive. The experimental results show that the proposed method achieves state-of-the-art performance on these datasets.

Keywords: Time series · K-means · 1-Nearest Neighbor · Instance reduction

1 Introduction

Time series data can be found in many areas of our life such as finance, economy, or medicine. Mining data in these areas includes many different tasks such as motif discovery, anomaly detection, rule extraction, clustering, or classification. In which, classification is definitely important as it is the background of pattern recognition. For example, it can be applied in signature verification, handwriting recognition, analysis of brainwaves, or electrocardiograph signals. One of challenging properties of time series data is its high dimensionality. Some classification models such as 1-Nearest Neighbor (1-NN), Artificial Neural Network, or Bayesian Network have been applied successfully in time series. Among them, the simple approach 1-NN has been considered as very hard to be beaten [1, 3]. Nevertheless, time and space efficiency of 1-NN is still need to be improved. To this end, one can apply these directions:

1. Using some indexing structures such as R-Tree [17], M-Tree [16], TS-Tree [18].
2. Reducing dimensions of time series [19, 20].
3. Speeding up computational time of distance measure [21, 22].
4. Reducing the number of instances in the training set [1, 2].

Following the fourth approach, Xi et al. [1] proposed Naïve Rank Reduction and Buza et al. [2] proposed Instance Selection based on Graph-coverage and Hubness for Time series (INSIGHT). Although these methods can help reduce instances, they have three drawbacks: (i) do not preserve the training set's distribution, (ii) may produce an imbalance training set, and (iii) must be provided the reduced training set size.

The contribution of our work can be summarized as follows. Firstly, we propose a new approach to instance reduction in time series classification using K-means. Secondly, our method can decide whenever to stop eliminating instances from the training set, in the meaning, if we continue to remove more instance, it would lead to poor results. The experiments were conducted over popular time series datasets from the UCR Classification Archive [7]. Experimental results show that the proposed method outperforms Naïve Rank Reduction and INSIGHT in most cases.

2 Backgrounds and Related Works

2.1 Time Series

A time series T is a sequence of real numbers collected at regular intervals over a period of time: $T = t_1, t_2,\ldots, t_n$. It can be considered as a n-dimensional object in a metric space.

2.2 1-Nearest Neighbor

The 1-Nearest Neighbor (1-NN) assigns a new data object to the same class of its nearest object in the training set. 1-NN has been considered hard to be beaten in classification of time series data among other methods such as Artificial Neural Network, Bayesian Network [1, 3]. In this work, we use 1-NN to classify time series data and evaluate the error rate of this classifier on the reduced training set obtained by our instance reduction method.

2.3 Instance Reduction

The main goal of instance reduction is: (i) to improve the running time of classification algorithms while losing accuracy as little as possible and (ii) to fit data, especially training data, into limited memory devices. Many instance reduction methods were proposed to speed up nearest neighbor classification of conventional data such as in [8–15]. However, it is hard to find an efficient classification method for time series data because of its high dimensionality characteristic. So far, there has been only two works related to instance reduction for time series data as above mentioned.

2.4 Naïve Rank Reduction

Xi et al. in 2006 [1] proposed a method called Naïve Rank Reduction for time series data which uses a ranking function developed from the ideas in the most referenced paper by Wilson and Martinez (1997) [6]. Naïve Rank Reduction method first assigns rank to each instance x in the training set by identifying its contribution to the classification of other instances. The instances are ranked by the following formula:

$$rank(x) = \sum_j \begin{cases} 1 & \text{if } class(x) = class(x_j) \\ -2 & \text{otherwise} \end{cases} \tag{1}$$

where x_j is the instance having x as its nearest neighbor. This ranking function attempts to keep the instances that contribute to correctly classifying other instances. The ranking function presented in Eq. (1) may produce the same rank for some instances. In order to break this tie, the authors used a second ranking function as presented in Eq. (2).

$$priority(x) = \sum_j \frac{1}{d(x, x_j)^2} \tag{2}$$

where x_j is the instance having x as its nearest neighbor and $d(x, x_j)$ is the distance between instances x and x_j. The idea behind Eq. (2) is that the farther the instance from its nearest neighbor, the lower priority it should have. Because this instance may be noisy or unrepresentative instance, so it should be eliminated first.

2.5 INSIGHT Method for Instances Selection

Buza et al. in 2011 [2] proposed a method called Instance Selection based on Graph-coverage and Hubness for Time series (INSIGHT), which ranks the time series in the training set by using one of their three score functions. These ranking functions attempt to exploit the hubness property of time series [2]. This property states that for data with high dimensionality, some objects tend to become nearest neighbors much more frequently than others. The term hubness is a phenomenon that the distribution of $f_N^k(x)$ significantly skewed to the right, where $f_N^k(x)$ denote the *k-occurrence* of an instance x in a training set D, which is the number of instances in D having x as their k nearest neighbor. The score functions of INSIGHT were defined as follows:

- $f_G(x) = f_G^1(x)$: Good 1-*occurrence* score.
- $f_R(x) = f_G^1(x) / \left(f_N^1(x) + 1\right)$: Relative score.
- $f_{Xi}(x) = f_G^1(x) - 2f_B^1(x)$: Xi's score (base on ranking criterion of Xi et al. [1]).

In which, $f_G^1(x)$ is the number of instances in the training set having x as their 1-nearest neighbor and x has the same label with them. $f_B^1(x)$ is the number of instances in the training set having x as their 1-nearest neighbor and label of x is different from their label.

2.6 Notes on Naïve Rank Reduction and INSIGHT

Naïve Rank Reduction was used in a method called Fast Time Series Classification Using Numerosity Reduction (FastAWARD) proposed by Xi et al. [1]. This method iteratively removes time series in a training set. In each iteration, ranks of all the time series in the training set is calculated and the lowest ranked time series is discarded. Thus, each iteration kept a particular number time series. Xi et al. claim that the best value of Sakoe-Chiba band [4, 5] for DTW distance depends on the number of time series in the reduced training set. Therefore, they include a final step in order to calculate the warping window size for each training set size. The INSIGHT method just ranks instances in the training set and reduces the training set only once. This method does not include the second step to find the value for Sakoe-Chiba band r as in FastAWARD. Instead, the warping-window size is set to 5 % of the time series length.

Our proposed method is different from the FastAWARD and INSIGHT in that we do not use a ranking function. Moreover, our method does not focus on finding Sakoe-Chiba band r since this additional step is not important but computationally expensive. In addition, our method can easily include this step. All things considered, we only compare our method with the Naïve Rank Reduction and INSIGHT.

2.7 Weaknesses of Naïve Rank Reduction and INSIGHT

The main idea of Naïve Rank Reduction and INSIGHT bases on ranking functions, which attempt to exploit the original training set to discover how much the instances impact in the classification process. Then, the instances are arranged from least to most impact. The instances contributing most to the classifier are selected first. This idea has three drawbacks as follows:

- It does not maintain the representative characteristics of the training set.
- It may lead to imbalance in the number of instances of each class in the reduced training set.
- It is parametric which means that we must provide the reduced training set size.

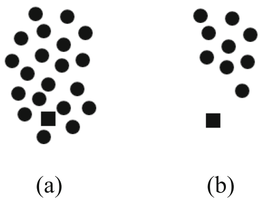


Fig. 1. Instance reduction using ranking strategy might lead to the representative characteristics of a training set are not maintained. *Circle*: training instances, *Square*: a new instance that need to be classified. (a) Original training set, (b) Reduced training set

With regards to the first drawback, because the instances that have most impact in the classification process are kept, the training set's distribution does not maintain as original. This means that representative characteristics of original training set are not preserved in reduced training set. As shown in Fig. 1, the original training set and the reduced training set have different distribution. Therefore, a new instance (square) in real world might be wrongly classified because the instances in the original training set that look like this new instance were removed.

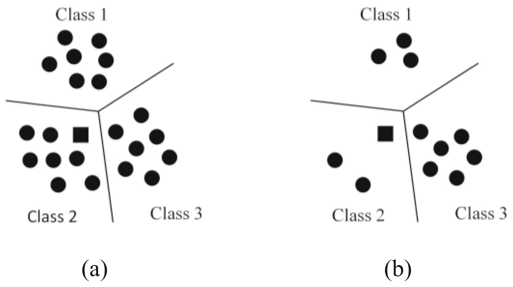


Fig. 2. There is some bias in the number of selected instances in each class. *Circle*: training instances, *Square*: a new instance that need to be classified. (a) Original training set with three classes has equal number of instances, (b) Reduced training set with more instances in class 1, meanwhile class 2 and class 3 has much less instances

For the second drawback, the ranking function may bias some classes while reducing instances. As a result, some classes may have much more instances than the others. We illustrate this issue in Fig. 2. The original training set has three classes; each contains 7 instances (circle). Meanwhile, the reduced training set is unbalanced, which has three instances for Class 1, two instance for Class 2, and six instances for Class 3. As a result, the new instance (square), which is classified as Class 2 with the original training set, will be classified as Class 3 with the reduced training set.

3 Proposed Method

3.1 The Framework

In order to overcome the drawbacks mentioned in Subsect. 2.7, we propose a framework to reduce training set size while maintaining distribution of each class. The outline of our method is described in Fig. 3. In order to keep $p\%$ instances in the training set, we run K-means on each class with the number of cluster $K = p$ x *number of instances in the class* to get K centroids. Then, the K centroids are used as training instances. All the instances kept in each class are combined together to get the reduced training set.

As we can see from Fig. 3, the framework uses K-means to partition n instances into K sets $C = \{C_1, C_2,\ldots, C_K\}$ so as to minimize the sum of distance of each instance in the cluster to the K centroids as in Eq. (3).

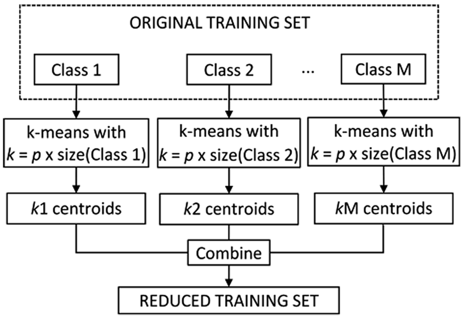


Fig. 3. The framework of instance reduction in a training set with *M* classes

$$\arg\min_{C} \sum_{i=1}^{K} \sum_{x \in C_i} dist(x, \mu_i) \quad (3)$$

where μ_i is the centroids of data instances in C_i. As a result, the instances with similar characteristics are grouped into one cluster and they are all similar to their centroid. Therefore, we can keep the centroids only as they can represent the whole cluster. Figure 4 illustrates effect of our framework, which can help keep representative instances while the number of instances in each class is balance as original. While the component of the framework, K-means, is well-known, we emphasize that the framework as a whole is new for the instance reduction of time series classification.

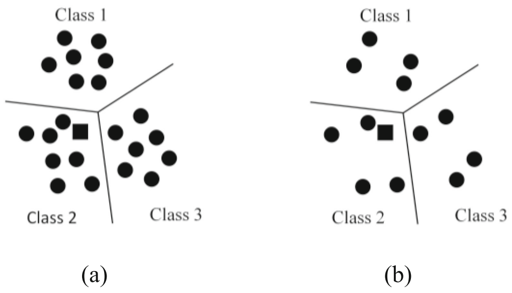


Fig. 4. *Circle*: training instances, *Square*: a new instance that need to be classified. (a) Original training set with three classes has equal number of instances, (b) Reduced training set with representative instances

3.2 Choosing Appropriate Percentage of Instances

One question in instance reduction problem is how many instances should be kept in the training set. In general, the bigger the training set size, the more accurate the classifier. Although accuracy rate should be considered as a major factor in deciding which training set is better, training set size should also be considered as a penalty.

If a training set has more instances than the others, we should penalize it more. For example, given a training set with 100 instances and accuracy rate for classification is 0.85. Now we remove this training set to 20 instances with its new accuracy rate for classification is 0.8. The accuracy rate decreases only 5 % but it can help to speedup 5 times. In this situation, it is more likely that we prefer the second choice. Akaike Information Criterion (AIC) and Bayesian Information Criterion (BIC) are the two popular model selections. The ideas behind these two methods are the same except that BIC gives more penalties to the model with more parameters. The formulas of AIC and BIC are as follows:

$$\begin{aligned} AIC &= -2\log L + 2k \\ BIC &= -2\log L + \log(N) \times k \end{aligned} \tag{4}$$

where k is the number of parameters in the model, log$L$ is a log-likelihood function, and N is the number of observed data. We can understand that AIC and BIC evaluate each model to find the best one bases on two factors: likelihood value and the number of parameters of the model. In other words, they attempt to balance likelihood and model's size. In this work, we adopt the above two formulas in Eq. (4) to select an appropriate point at which we should not remove anymore instances. Note that we only apply the idea of balancing between likelihood value and the number of parameters of a model. So we emphasize that our method is not AIC or BIC. The formulas are redefined as follows:

$$\begin{aligned} f1\_score &= 2 \times percentage\_error + 2 \times k \\ f2\_score &= 2 \times percentage\_error + \log(N) \times k \end{aligned} \tag{5}$$

Where:

- *percentage_error*: the percentage of wrongly classified instances.
- k: the number used to penalize reduced training set, which is defined based on the percentage of selected instances as follows:

Percentage	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90
k	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18

- N: total number of instance in original training set.

In general, the percentage of selected instances with the lowest value of these two criterions is the best one. In Eq. (5), we map the percentage of selected instance to the number of parameters. For example, if we select 10 % instances as reduced training set, the value of k is 2. Now why do we need this mapping? The answer is that if we do not map but use the percentage of selected instance, then it will penalize a lot, i.e. the value of k will have a high weight in selecting appropriate training set, and thus the accuracy rate does not play an essential part in deciding the percentage at which we choose to reduce the training set to.

Figure 5 illustrates how to identify the number of instances for reduced training set. In which we run our framework in Subsect. 3.2 with different number of instances, i.e. 10 %, 15 %, 20 %,…, 90 % of original training set. Then, we use these reduced training sets to classify original one (100 % instances) to get the percentage of wrongly classified instances (*percentage_error*). Finally, Eq. (5) is applied to find out the best point to reduce the training set to.

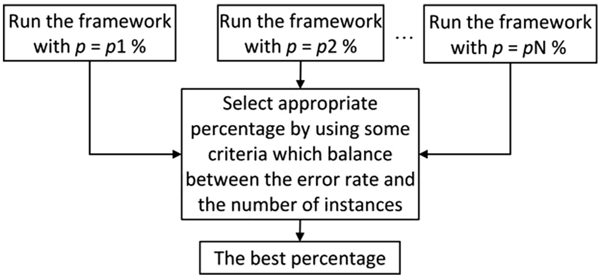

Run the framework with p = p1 %
Run the framework with p = p2 %
…
Run the framework with p = pN %
Select appropriate percentage by using some criteria which balance between the error rate and the number of instances
The best percentage

Fig. 5. Appropriate percentage selection

4 Experiments

This section shows two main experiments. Firstly, we compare accuracy rate of our method with those of Naïve Rank Reduction, and INSIGHT. Accuracy rate is the ratio of the number of correctly classified instances on tested data to the total number of tested instances. In general, the higher accuracy rate, the better method. Secondly, we find the percentage at which we should not remove any more instance. 10-fold cross-validation was applied over 46 datasets from the UCR Time Series Classification Archive [7]. Each datasets has from 2 to 50 classes. The length of each time series in each datasets is from 60 to 1882. Euclidean Distance was used as similarity measurement. The ranking function used in INSIGHT method is *Good* 1-*occurrence score* as this is the function used in the experiment of Buza et al. [2].

4.1 Comparing to Previous Methods

The two scatter graphs in Fig. 6 show comparison of the proposed method with those of Naïve Rank Reduction and INSIGHT. Each circle point in the scatter graphs represents a dataset. As we can see from the figures, most circle points deviate on the lower triangle, which means that the proposed method is better than the two previous ones. In detail, the proposed method outperforms INSIGHT in 41 datasets, draws in 1 datasets, and loses in 4 datasets. In comparison to Naïve Rank Reduction, the proposed method outperforms it in 42 datasets, draws in 1 datasets, and loses in only 3 datasets. Moreover, Table 1 shows accuracy rate of the three approaches on each dataset.

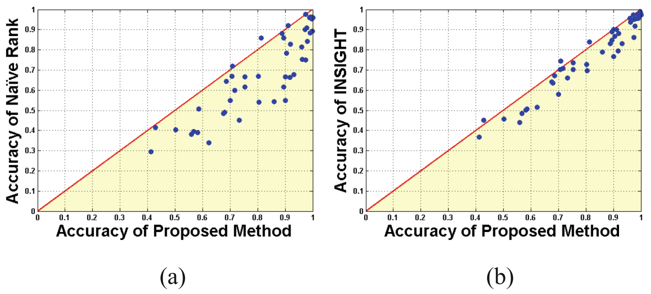


Fig. 6. Comparing the accuracy rates of the proposed method with those of Naïve Rank Reduction (a), INSIGHT (b) on 46 datasets

Table 1. Accuracy rate ± standard deviation in 46 datasets after reducing to 20 % of original training set

Datasets	Naïve Rank	INSIGHT	Proposed Method
50words	0.451 ± 0.07	0.66 ± 0.047	0.732 ± 0.043
Adiac	0.341 ± 0.052	0.515 ± 0.039	0.623 ± 0.041
Beef	0.38 ± 0.166	0.44 ± 0.12	0.56 ± 0.12
Car	0.6 ± 0.128	0.708 ± 0.085	0.717 ± 0.076
CBF	0.892 ± 0.036	0.983 ± 0.012	0.999 ± 0.003
ChlorineConcentration	0.858 ± 0.018	0.84 ± 0.009	0.813 ± 0.019
CinC_ECG_torso	0.961 ± 0.015	0.978 ± 0.016	0.997 ± 0.006
Coffee	0.975 ± 0.075	0.95 ± 0.1	0.975 ± 0.075
Cricket_X	0.39 ± 0.053	0.501 ± 0.035	0.582 ± 0.054
Cricket_Y	0.403 ± 0.028	0.457 ± 0.063	0.503 ± 0.048
Cricket_Z	0.394 ± 0.041	0.485 ± 0.058	0.567 ± 0.072
DiatomSizeReduction	0.96 ± 0.036	0.973 ± 0.036	1 ± 0
ECG200	0.784 ± 0.076	0.868 ± 0.072	0.905 ± 0.046
ECGFiveDays	0.964 ± 0.017	0.957 ± 0.016	0.991 ± 0.012
FaceAll	0.665 ± 0.034	0.794 ± 0.041	0.917 ± 0.022
FaceFour	0.667 ± 0.131	0.767 ± 0.105	0.9 ± 0.105
FacesUCR	0.677 ± 0.045	0.831 ± 0.015	0.931 ± 0.012
FISH	0.669 ± 0.09	0.729 ± 0.05	0.803 ± 0.041
Gun_Point	0.86 ± 0.073	0.89 ± 0.066	0.895 ± 0.052
Haptics	0.416 ± 0.065	0.451 ± 0.072	0.429 ± 0.076
InlineSkate	0.295 ± 0.056	0.366 ± 0.059	0.413 ± 0.07
ItalyPowerDemand	0.902 ± 0.024	0.974 ± 0.017	0.972 ± 0.01
Lighting2	0.718 ± 0.143	0.745 ± 0.151	0.709 ± 0.134
Lighting7	0.55 ± 0.081	0.58 ± 0.108	0.7 ± 0.148
MALLAT	0.843 ± 0.075	0.955 ± 0.012	0.98 ± 0.01
MedicalImages	0.615 ± 0.044	0.704 ± 0.04	0.753 ± 0.034

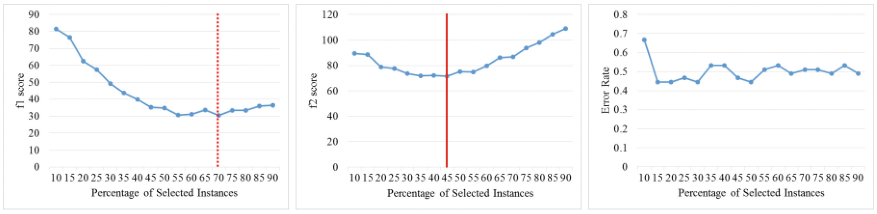
(*continued*)

Table 1. (*continued*)

Datasets	Naïve Rank	INSIGHT	Proposed Method
MoteStrain	0.919 ± 0.025	0.902 ± 0.03	0.912 ± 0.028
NonInvasiveFatalECG_Thorax1	0.544 ± 0.046	0.789 ± 0.008	0.86 ± 0.01
NonInvasiveFatalECG_Thorax2	0.616 ± 0.037	0.848 ± 0.01	0.896 ± 0.011
OliveOil	0.55 ± 0.1	0.9 ± 0.122	0.9 ± 0.122
OSULeaf	0.507 ± 0.066	0.507 ± 0.068	0.585 ± 0.039
Plane	0.752 ± 0.092	0.938 ± 0.043	0.962 ± 0.029
SonyAIBORobotSurfaceII	0.908 ± 0.03	0.919 ± 0.031	0.978 ± 0.017
SonyAIBORobotSurface	0.884 ± 0.042	0.956 ± 0.023	0.99 ± 0.011
SwedishLeaf	0.54 ± 0.031	0.697 ± 0.027	0.805 ± 0.036
Symbols	0.813 ± 0.071	0.957 ± 0.016	0.961 ± 0.01
synthetic_control	0.88 ± 0.035	0.83 ± 0.018	0.888 ± 0.025
Trace	0.485 ± 0.098	0.64 ± 0.092	0.675 ± 0.098
TwoLeadECG	0.959 ± 0.016	0.973 ± 0.013	0.99 ± 0.01
Two_Patterns	0.751 ± 0.028	0.863 ± 0.016	0.974 ± 0.006
uWaveGestureLibrary_X	0.665 ± 0.027	0.735 ± 0.011	0.753 ± 0.02
uWaveGestureLibrary_Y	0.668 ± 0.018	0.704 ± 0.019	0.706 ± 0.017
uWaveGestureLibrary_Z	0.644 ± 0.03	0.67 ± 0.014	0.686 ± 0.015
wafer	0.955 ± 0.01	0.991 ± 0.005	0.998 ± 0.002
WordsSynonyms	0.49 ± 0.056	0.635 ± 0.045	0.68 ± 0.046
yoga	0.829 ± 0.026	0.881 ± 0.021	0.92 ± 0.017

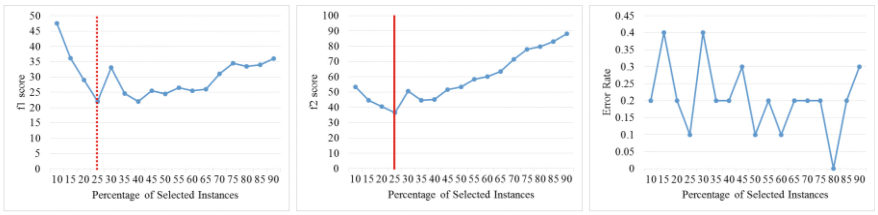
4.2 Experiments on Choosing Appropriate Percentage

Figure 7(a) and (b) illustrate *f*1*_score* and *f*2*_score* respectively on Haptics dataset. As we can see, *f*1*_score* selects 70 % as the percentage that we should reduce the training set to and *f*2*_score* selects at 45 %. Comparing these points to the figures on Fig. 7(c), which shows the error rates when we use the reduced training sets to classify instances on testing set, these reductions are acceptable because the error rates at these points do not increase too much. In detail, the training set with 70 % chosen by *f*1*_score* has the



(a) (b) (c)

Fig. 7. Percentage of instances selected in Haptics, (a) *f*1*_score* selects at 35 %, (b) *f*2*_score* selects at 15 %, (c) error rates at each percentage of instances kept as training set



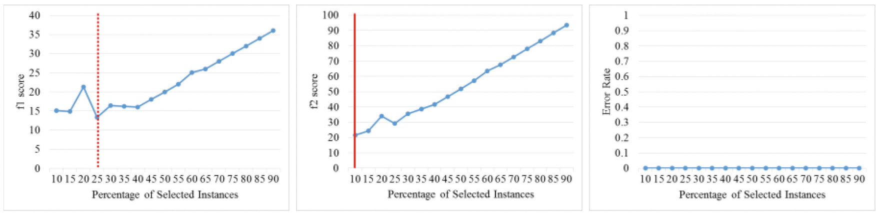
(a) (b) (c)

Fig. 8. Percentage of instances selected in Lightning 7 dataset, (a) *f1_score* selects at 25 %, (b) *f2_score* selects at 25 %, (c) error rates at each percentage of instances kept as training set

error rate of 0.51. The training set with 45 % chosen by *f2_score* has the error rate of 0.46. Meanwhile, the training set of 90 % has the error rate of 0.48.

Figure 8(a) and (b) illustrate *f1_score* and *f2_score* respectively on Lightning 7 dataset. Both *f1_score* and *f2_score* select at 25 % as the percentage that we should reduce the training set to. Compare these points to the corresponding error rates on Fig. 8(c), which use the reduced training sets to classify instances on testing set, these reductions are acceptable because the error rates at these points even decreased. In detail, the training set with 25 % instances has the error rate of 0.1. Meanwhile, the training set with 90 % instances has the error rate of 0.3.

Figure 9(a) and (b) illustrate *f1_score* and *f2_score* respectively on Gun Point dataset. *f1_score* selects at 25 % as the percentage that we should reduce the training set to and *f2_score* selects at 10 %. Comparing these points to the error rate on Fig. 9 (c) when we use the reduced training sets to classify instances on testing set, these reductions are reasonable because the error rate is 0.0 at all the percentage of reduction.



(a) (b) (c)

Fig. 9. Percentage of instances selected in Gun Point dataset, (a) *f1_score* selects at 25 %, (b) *f2_score* selects at 10 %, (c) error rates at each percentage of instances kept as training set

5 Conclusions

Reducing training set size is vitally important in improving speed and space using of 1-NN, an efficient method for time series classification. In this work, our method reduces training set size by using K-means to determine its representative characteristics. Then, only representative instances are kept as training data. In comparison with previous methods, our method has a different viewpoint as it does not use a ranking function, which can lead to some weaknesses mentioned in this paper. In addition, the proposed method can identify an appropriate percentage which we should choose to reduce the training set to. The experimental results show that the proposed method outperforms Naïve Rank Reduction and INSIGHT.

References

1. Xi, X., Keogh, E., Shelton, C., Wei, L., Ratanamahatana, C.A.: Fast time series classification using numerosity reduction. In: Proceedings of the 23rd International Conference on Machine learning, ICML 2006, pp. 1033–1040 (2006)
2. Buza, K., Nanopoulos, A., Schmidt-Thieme, L.: INSIGHT: efficient and effective instance selection for time-series classification. In: Huang, J.Z., Cao, L., Srivastava, J. (eds.) PAKDD 2011. LNCS (LNAI), vol. 6635, pp. 149–160. Springer, Heidelberg (2011). doi:10.1007/978-3-642-20847-8_13
3. Ding, H., Trajcevski, G., Scheuermann, P., Wang, X., Keogh, E.: Querying and mining of time series data: experimental comparison of representations and distance measures. Proc. VLDB Endowment **1**(2), 1542–1552 (2008)
4. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowl. Inf. Syst. **7**(3), 358–386 (2005)
5. Berndt, D., Clifford, J.: Using dynamic time warping to find patterns in time series. In: Proceedings of AAAI Workshop on Knowledge Discovery in Databases, KDD 1994, Seattle, Washington, USA, pp. 359–370 (1994)
6. Wilson, D.R., Martinez, T.R.: Instance pruning techniques. In: Proceedings of ICML 1997, Morgan Kaufmann, pp. 403–411 (1997)
7. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A. and Batista, G.: The UCR Time Series Classification Archive (2015). www.cs.ucr.edu/~eamonn/time_series_data/
8. Brighton, H., Mellish, C.: Advances in instance selection for instance-based learning algorithms. Data Min. Knowl. Discov. **6**, 153–172 (2002)
9. Garcia, S., Derrac, J., Cano, J.R., Herrera, F.: Prototype selection for nearest neighbor classification taxonomy and empirical study. IEEE Trans. Pattern Anal. Mach. Intell. **34**(3), 417–435 (2012)
10. Jankowski, N., Grochowski, M.: Comparison of instances seletion algorithms I. algorithms survey. In: Rutkowski, L., Siekmann, Jörg, H., Tadeusiewicz, R., Zadeh, Lotfi, A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 598–603. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24844-6_90
11. Grochowski, M., Jankowski, N.: Comparison of instance selection algorithms II. results and comments. In: Rutkowski, L., Siekmann, J.H., Tadeusiewicz, R., Zadeh, L.A. (eds.) ICAISC 2004. LNCS (LNAI), vol. 3070, pp. 580–585. Springer, Heidelberg (2004). doi:10.1007/978-3-540-24844-6_87
12. Triguero, I., Derrac, J., Garcia, S., Herrera, F.: A Taxonomy and experimental study on prototype generation for nearest neighbor classification. IEEE Trans. Syst. Man Cybern. Part C Appl. Rev. **41**(1), 86–100 (2012)
13. Wilson, D.L.: Asymptotic properties of nearest neighbor rules using edited data. IEEE Trans. Syst. Man Cybern. **2**(3), 408–421 (1972)

14. Wilson, D.R., Martinez, T.R.: Instance pruning techniques. In: Proceedings of ICML1997, Morgan Kaufmann, pp. 403–411 (1997)
15. Wilson, D.R., Martinez, T.R.: Reduction techniques for instance-based learning algorithms. Mach. Learn. **38**, 257–286 (2000)
16. Ciaccia, P., Patella, M., Zezula, P.: M-tree: an efficient access method for similarity search in metric spaces. In: VLDB 1997 Proceedings of the 23rd International Conference on Very Large Data Bases, pp. 426–435 (1997)
17. Guttman, A.: R-trees: a dynamic index structure for spatial searching. In: SIGMOD 1984 Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 47–57 (1984)
18. Assent, I., Krieger, R., Afschari, F., Seidl, T.: The TS-tree: efficient time series search and retrieval. In: Proceedings of the 11th International Conference on Extending database technology: Advances in database technology (EDBT 2008), pp. 252–263 (2008)
19. Keogh, E., Chakrabarti, K., Pazzani, P., Mehrotra, S.: Dimensionality reduction for fast similarity search in large time series databases. Knowl. Inf. Syst. **3**, 263–286 (2001)
20. Keogh, E., Pazzani, M.: An enhanced representation of time series which allows fast and accurate classification, clustering and relevance feedback. In: Proceedings of the 4th International Conference on Knowledge Discovery and Data Mining, pp. 239–241 (1998)
21. Rakthanmanon, T., Campana, B., Mueen, A., Batista, G., Westover, B., Zhu, Q., Zakaria, J., Keogh, E.: Searching and mining trillions of time series subsequences under dynamic time warping. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2012), pp. 262–270 (2012)
22. Keogh, E., Ratanamahatana, C.A.: Exact indexing of dynamic time warping. Knowl. Inf. Syst. **7**, 358–386 (2005)