Journal of King Saud University – Computer and Information Sciences 34 (2022) 134–142

Contents lists available at ScienceDirect

Journal of King Saud University – Computer and Information Sciences

journal homepage: www.sciencedirect.com

Time series classification through visual pattern recognition

Agnieszka Jastrzebska

Faculty of Mathematics and Information Science, Warsaw University of Technology, Warsaw, Poland

ARTICLE INFO

Article history:
Received 29 September 2019
Revised 22 November 2019
Accepted 22 December 2019
Available online 27 December 2019

Keywords:
Time series
Time series representation
Classification
Pattern recognition
Image recognition

ABSTRACT

In this paper, a new approach to time series classification is proposed. It transforms the scalar time series into a two-dimensional space of amplitude (time series values) and a change of amplitude (increment). Subsequently, it uses this representation to plot the data. One figure is produced for each time series. In consequence, the time series classification problem is converted into the visual pattern recognition problem. This transformation allows applying a wide range of algorithms for standard pattern recognition – in this domain, there are more options to choose from than in the domain of time series classification. In this paper, we demonstrated the high effectiveness of the new method in a series of experiments on publicly available time series. We compare our results with several state-of-the-art approaches dedicated to time series classification. The new method is robust and stable. It works for time series of differing lengths and is easy to extend and alter. Even with a baseline variant presented in an empirical study in this paper, it achieves a satisfying classification accuracy. Furthermore, the proposed conversion of raw time series into images that are subjected to feature extraction opens the possibility to apply standard clustering algorithms.

1. Introduction

Time series classification is a popular machine learning problem that finds practical applications in many control systems, including medicine (Molina et al., 2016), human activity monitoring (Ignatov, 2018), (hydrology Montgomery et al., 2018), agriculture (Muro et al., 2018), and more. On an input, we get a training set of time series with class labels. The goal is to construct a model, which will classify a previously unseen time series to one of the available classes present in the training set. The difference between a standard classification problem and a time series classification problem is that in the latter the order of elements is important (Atluri et al., 2018).

The majority of the existing methods for time series classification operate on time series values directly. Considered are straightforward time series comparisons where data is analyzed "as is", such as presented by Mori et al. (2016) and Wang et al. (2016) and comparisons of time series subsequences (Grabocka et al., 2015). The wealth of algorithms dedicated to efficient time series processing allowed to reduce the complexity of time series similarity evaluation by introducing techniques such as dimensionality reduction (O'Reilly et al., 2017) and data segmentation or local searches (Baydogan and Runger, 2016).

Looking at a broad spectrum of classification problems, not only in time series analysis domain, one shall notice that for an automatic pattern recognition, sometimes it is difficult to determine proper features, i.e. mathematically encoded particular properties of a signal, that would precisely discriminate between objects that should belong to different groups (Zhou et al., 2016). This is why machine learning puts a high emphasis on the so-called data representation, which is the data format on which processing algorithms are launched (van Noord and Postma, 2017). In contrast to standard pattern recognition domains, where switching data representation is a common practice (Xu et al., 2017), existing time series classification procedures typically do not introduce very drastic changes to data representation. The only exceptions are methods presented by Hatami et al. (2018), Silva et al. (2013), Wang and Oates (2015) where authors plot time series and convert time series classification problems to image recognition problems. The main issue with the above-mentioned methods is that the applied image-based representations are "heavy" – discussed representations require color or gray-scale plots.

In this paper, we present a new approach to time series classification that relies on representation transformations. In the first

Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

E-mail address: A.Jastrzebska@mini.pw.edu.pl

https://doi.org/10.1016/j.jksuci.2019.12.012
1319-1578/

step, we move univariate time series into a two-dimensional space. Next, we store this representation in the form of images. At this point, the time series classification problem is converted into an image recognition problem. We propose to solve the given image recognition problem in two steps: by extracting numerical features describing images and by training a classifier using those features. The proposed time series processing scenario is novel, and it makes the core contribution of this paper. The image-based time series representation introduced in this paper is monochrome and, thus, less resource-consuming than other existing methods.

The specific objectives of this study are as follows.

- We present new time series classification scenario based on representation transformations.
- We address subsequent steps of the proposed procedure and highlight how they can be extended.
- We demonstrate its effectiveness on time series of different characteristics and sizes.
- We provide a comparative study of the effectiveness of the new method with several state-of-the-art approaches to time series classification.

The remainder of the paper is organized as follows. In Sections 2 and 3, relevant literature positions are discussed. Section 4 presents new method. In Section 5 an empirical study on the new approach is presented. Section 6 concludes the paper and highlights future research directions.

2. Literature review on time series classification

Time series classification has been intensively researched for well over a decade now. Among the most popular methods, we find those related to dynamic time warping. Dynamic time warping, DTW for short, is a method for measuring the similarity between two temporal sequences. It aims at matching two sequences in a way that a cost is minimized. The cost is computed as a sum of absolute differences between the values of each pair of matched indices. Sequences are warped non-linearly, which induces that DTW can be applied to evaluate the similarity of sequences of a different length (Dau et al., 2018).

An interesting idea of how to classify time series has been presented by Stefan et al. (2013). This method, called Move-Split-Merge, is based on a concept that we can transform a given time series into another time series. Steps needed to do this transformation are named "move" (which changes a value of one element), "split" (which converts one element into two consecutive elements), and "merge" (which merges two consecutive elements into one). These operations have costs assigned to them (alike in the DTW) and, thus, the smaller the cost of a transformation, the more similar are the two time series.

A distinct collection of methods aims at computing certain statistical features based on time series, for which we wish to compute similarity. For example, Bagnall and Janacek (2014) utilize the concept of a run length of a time series. The referenced method involves forming a histogram of above and below the mean run lengths, which assures its scalability to big data sets and applicability to data streams. In contrast, Baydogan et al. (2013) apply a bag-of-feature approach. They split a time series into multiple subsequences, for which certain features/patterns are computed, and then compared.

The next group of methods postulates the use of time series transforms. Frequently utilized transforms include shapelets and derivatives. After a time series data set gets transformed, one still needs an algorithm for evaluating similarity. Bagnall et al. (2015) introduce a method based on multiple transforms and a set of elastic distance measures. It is called COTE, which stands for the Collective of Transformation-Based Ensembles. Gorecki and Luczak (2014) apply time series transforms together with the DTW to classify time series.

The literature covers a few image-based methods for time series classification. This group is very small, but we are distinguishing it because the method introduced in this paper belongs to it. The image-based approaches to time series classification aim at encoding time series as images. Thus, a time series classification problem is transformed to an image recognition problem. Wang and Oates (2015) discussed an approach to time series classification based on Gramian Angular Summation/Difference Fields and Markov Transition Fields saved as images. The authors used Tiled Convolutional Neural Networks to perform the task of image recognition. Hatami et al. (2018) and Silva et al. (2013) encoded time series as Recurrence Plots. In both cases, used images must be color or gray-scale.

3. Literature review on pattern classification

The task of classification aims at categorizing unknown patterns to appropriate groups. The procedure is based on quantifiable characteristics obtained from the source signal. Those characteristics, i.e. features, are gathered in a feature vector (of independent variables) and each pattern is described with one vector of features' values. We expect that patterns accounted for the same category are in some sense similar. There are many mathematical models that can be used as classifiers. Let us mention two classification algorithms that we later use the empirical study.

Support Vector Machine (SVM) is a supervised learning method used for classification, regression and outliers detection. The SVM algorithm relies on a construction of hyperplane with a maximal margin that separates patterns of two classes (Cortes and Vapnik (1995)). SVM is effective in high-dimensional spaces, memory efficient, and quite versatile with many kernel functions that can be specified for the decision function.

Random forest is a popular ensemble method. The main principle behind ensemble methods, in general, is that a group of "weak learners" can come together to form a "strong learner". In the random forest algorithm introduced by Breiman (2001), the weak learners are decision trees, which are used to predict class labels. After a relatively large number of trees is generated, they vote for the most popular class. Random forests join few important benefits: (a) they are relatively prone to the influence of outliers, (b) they have an embedded ability of feature selection, (c) they are prone to missing values, and (d) they are prone to overfitting.

The aforementioned algorithms operate on a numerical data table. Thus, if one wants to solve an image recognition problem using those tools, it is necessary to extract numerical features from images. There exist several approaches to feature extraction. In this paper, we utilized a popular approach to feature extraction suitable to monochrome images. We produce vectorial features such as histograms and transition vectors and then extract numerical features from them. This methodology is not only but also described by Luo et al. (2016). At the moment, the trending approach is to apply Convolutional Neural Networks which take as an input training set of images and build a classifier directly on them. In this case, features correspond to pixel values that are present in subsequent layers of a network (Li et al., 2017).

There is no single, universal pattern recognition processing scheme that is suitable for each data set. The choice of a particular method depends on multiple factors, the main one being the properties of a given data set. For instance, in image recognition problems where objects of interest may be located in different regions of a picture or are being rotated, Convolutional Neural Networks

shall perform better (Hu et al., 2015). In contrast, studies report that the image classification procedure split into feature extraction and then standard classification performs better than Convolutional Neural Networks when images are properly segmented (Amri et al., 2018). In the case of the image-based time series representation discussed in this paper, there is no need to apply Convolutional Neural Networks since plots are constructed in an intentional way.

4. Methods

In this section, we discuss the introduced method which translates the time series classification problem into the visual pattern recognition problem using an alternative data representation scheme. We denote a scalar time series as the following sequence:

$$z_1, z_2, z_3, \ldots, z_n \tag{1}$$

where $z_i \in \mathbb{R}$ and n is the number of data points in the time series. The usual assumption holds, that a time series is an ordered sequence of values at equally spaced time intervals. The sequence coded in Eq. (1) is the most elementary representation scheme of any univariate time series. We can transform this sequence into a two-dimensional real-coordinate space of amplitude and an increment (a difference). We elevate the sequence of scalars from the Eq. (1) into a sequence of pairs:

$$(z_2, dz_2), (z_3, dz_3) \ldots, (z_n, dz_n) \tag{2}$$

where $dz_i = z_i - z_{i-1}$. In this form, the lag is arbitrarily set to 1. Numerical differentiation displayed in Eq. (2) is equivalent to the first discrete derivative of the time series. Other variants (i.e. second derivative, third, etc.) may be considered as well as viable variations of the discussed method.

Subsequently, we plot and save each time series plot. It is necessary to use the same scale and graphical parameters for each plot. Only the data scatter should be saved, not the axes or other elements. All images are monochrome. After performing the above-mentioned steps, we arrive at the visual pattern recognition problem, in which the input data are labeled (time series) images.

Since all images are monochrome, a single image can be represented as a mapping $I : \langle 1, h\rangle \times \langle 1, w\rangle \to \{0, 1\}$, where $I(i, j) = 0$ for a white pixel and $I(i, j) = 1$ for a black pixel. h and w correspond to the height and the width of the pattern. A wealth of numerical features can be extracted when we construct vectorial features. The key vectorial features are:

- projections (horizontal and vertical), which are vectors of numbers of black pixels in rows and columns (respectively),
- bottom, top, left, and right margins which denote indices of first/last pixels vertically and horizontally,
- transitions (horizontal and vertical), which tell the number of pairs of two consecutive white and black pixels in rows and columns respectively.

Named vectorial features can be transformed into other vectorial features, namely histograms and cumulative histograms. Subsequently, based on all vectorial features, we compute target numerical features such as minimum or maximum value, the position of minimum/maximum value, etc. (Homenda and Pedrycz, 2018) provide a detailed description of all 150 used features.

Subsequently, we use one of many available standard classification algorithms, which on the input assumes numerical feature vectors and compute class labels. Due to the specifics of the chosen feature extraction procedure, many features may be correlated. Some classification algorithms perform poorly with such data sets. Hence, one shall either select an algorithm that performs some sort of feature selection procedure or applies a feature selection procedure to choose the best features. By classifying images, we indirectly classify time series. We propose to call this method PAINTS, which is an acronym from PAINt Time Series.

The motivation behind encoding a time series in the form of a two-dimensional data set is that an analogous operation (using a differenced time series and original time series) proved to be successful in other studies on time series analysis such as studies presented by Gorecki and Luczak (2014).

The discussed method mimics one of the ways, in which a human could solve a time series classification problem. Cognitive capabilities of an "average" human being do not allow to perform efficient mental computations on long sequences of real numbers. Hence, the standard time series representation form, given in Eq. (1), is not a convenient format. In contrast, human vision is powerful enough, to distinguish the content of two images.

Let us present an example of a real-world time series named StarLightCurves, available under (Bagnall et al., 2019). The data set in the referenced repository contains 1,000 training samples (time series), each of length 1,024. There are three classes (denoted as 1, 2, and 3), which have been determined by an expert. In Fig. 1, we illustrate randomly chosen three time series, one from each class, plotted as a colorful scatter in the two-dimensional space of amplitude and its difference.

We may inspect Fig. 2, in which all samples from the training data set are displayed in one plot. Colors allow distinguishing between classes and they are used only to enhance the clarity.

It must be stressed, that in the experiments presented in this paper, we aimed at establishing a baseline variant of the introduced approach. Hence, we were using monochrome images with relatively low resolution. In the course of experiments, we settled on images of size 150 pixels by 150 pixels and all plots use the same axes scale. In Fig. 3, we display a sample time series from the StarLightCurves data set, which was used to construct a time series classifier for this data set.

We can efficiently recognize written characters and we can also differentiate between plotted time series. On this ground, we proposed to translate the time series classification problem into the visual pattern recognition problem hoping that such a representation would be beneficial in some cases. Naturally, no algorithm is universally good and this one is not either.

The developed method utilizes knowledge about the order of data points only to a rudimentary extent. A single plotted data point encodes information about a change that happens from the moment in time i to the moment in time $i + 1$. This representation and, what follows, the method does not take full advantage of all the available information (as most of the methods described in Section 2 do).

A tempting variant of the proposed method is to use plots of amplitude in time and classify such images. However, this approach is less robust for time series of different lengths. Besides, it has brought very poor results in preliminary experiments.

The question is, do we need the plot or not. One may propose to transform a time series into the two-dimensional space and then label each created tuple with a class label that it was computed for. It is possible to run a classifier on such data directly and use a voting scheme to determine class membership of a sequence. In practice, this solution is infeasible. The transformation generates a lot of data points that have only two attributes. The size of the problem (the number of data samples in the training set) is equal to the number of time series multiplied by the length of the time series (minus one). In preliminary experiments, this approach gave classification accuracy worse than a random guess. Also, this approach is in contradiction with the motivation that the method

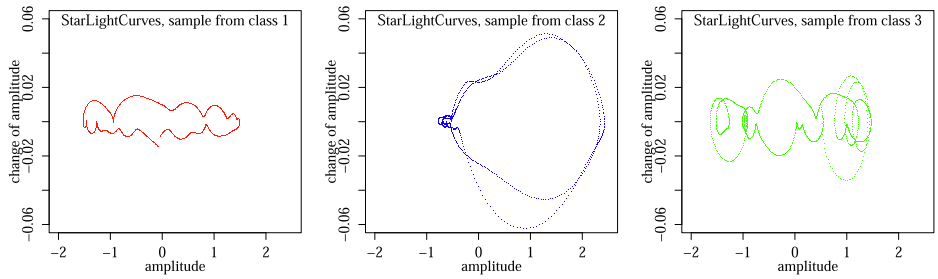


Fig. 1. Selected three time series from the StarLightCurves data set, from class 1, 2, and 3 in the two-dimensional space of an amplitude and a change of amplitude. The plots serve as an illustration only. In the experiments, we used monochrome plots with low resolution. Also, only data scatter must be subjected to classification (while in the plots above one can see scales and other graphical components).

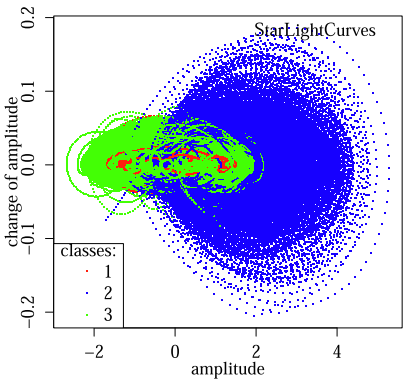


Fig. 2. All training samples from the StarLightCurves data set in the two-dimensional space of amplitude and change of amplitude. Colors mark different classes.

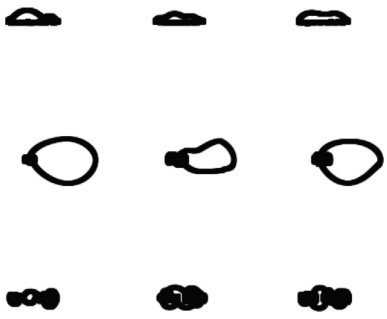


Fig. 3. Selected time series from the StarLightCurves data set, three samples from each class. Class 1 in row 1, class 2 in row 2, class 3 in row 3. Data scatter is plotted in the two-dimensional space of amplitude and change of amplitude. The size of an image is 150 pixels by 150 pixels. The scale is the same, hence a lot of white space. These are the images used to construct a classifier.

should simulate human cognition. Thus, we do need images to classify time series.

The time complexity of the proposed method shall be computed by summing the time complexity of the feature extraction step, feature selection step, and classification step. Feature extraction step run time depends linearly on the size of the input. In particular, it depends on the number of processed images and on their size (width times height in pixels). The time complexity of the feature selection step shall be computed with a particular algorithm in mind. There exist index-based feature ranking algorithms that have a linear time complexity. However, wrapper-based feature selection algorithms require much more time. A run time of a wrapper-based feature selection depends on the search strategy it applies and classifier training that is used as an evaluator. Thus, for the best-first or hill-climbing searches in the worst-case scenario time gets proportional to $b^{(f+1)}$ where b is the number of branches that need to be checked and f is the number of features. Finally, we need to consider the time complexity of classifier training which, again, would depend heavily on the algorithm that was chosen. In the case of the Support Vector Machine algorithm, time complexity depends on the number of instances in the training set, the number of features, the type of kernel function and the regularization parameter. A realistic estimation of Support Vector Machine time complexity that includes its various sub-routines was presented by Abdiansah and Wardoyo (2015). The authors show that for a Radial Basis Function kernel one may expect time complexity proportional to a polynomial of degree two of the number of instances. In contrast, a neural network classifier training would require more time to train. Assuming that a network has n layers and each layer has n neurons, backpropagation (which is slower than forward propagation) requires time proportional to n^5, as it was shown by Livni et al. (2014).

The proposed approach can be also applied to cluster a data set of time series. In this scenario, on the input we obtain multiple time series. Each time series gets transformed into a plot. Subsequently, we extract numerical features and perform the task of clustering based on those features. We would be able to apply standard numerical data processing tools. For example, feature selection or dimensionality reduction could be followed by a range of well-known clustering algorithms such as k-means, DBSCAN or spectral clustering. The procedure produces a standard numerical data table, where columns correspond to features and rows to instances. This enables the application of well-known algorithms which is a great advantage of our approach.

Finally, it should be stressed, that the method is easy to extend and alter. The most apparent modifications concern image recognition tools, which are being researched extensively and one can easily find advanced and efficient solutions.

5. Results and discussions

5.1. Data sets and implementation details

The research community working on time series classification frequently uses the already-mentioned UEA & UCR repository (Bagnall et al., 2019). We selected 30 data sets from this repository to illustrate the capabilities and properties of the designed approach. Table A.2 in the Appendix contains elementary information about the selected data sets.

The implementation of the introduced procedure in this paper is comprised of three components. The first component, written in R (all base functions), was used to read in data sets and produce plots. Plots are monochrome, 150 by 150 pixels. For each data set, in all plots, the same scale on axes was used. We determine a minimum and maximum limits using the entire training set. The second component, written in C (all elementary libraries), was used to read in plots and compute numerical features for each image. The output of this program was a csv file. The third element, written in R, was used to read in csv files, construct classifiers based on training sets, and evaluate quality. Two classifiers were used: Support Vector Machine (SVM) algorithm, available in package e1071 and Random Forest (RF), available in package randomForest. SVM was fitted with a radial basis function kernel. Gamma and cost parameters were tuned individually for each training set via a cross-validation procedure that was employing different parameter sets. SVM training was done using a ten-fold cross-validation procedure. Random Forests were constructed with 500 trees. A feature selection method based on a best-first search was used (implemented in package FSelector). It is a wrapper-based, greedy search feature selection method that evaluates multiple models, constructed using different subsets of features. SVM with fixed parameters was employed as an evaluator.

We use classification accuracy to assess the quality of the proposed method. It is computed as:

$$\text{accuracy} = \frac{\text{correctly classified samples}}{\text{total number of samples}} \tag{3}$$

We applied relevant statistical tests to compare our method with selected state-of-the-art methods. The tests were implemented in R package scmamp.

All classifiers were trained on dedicated training sets. All presented results concern test sets only. All classifier training procedures were repeated ten times.

Classifier training is the only non-deterministic procedure in the experimental schema described in this paper. Hence, it was necessary to validate the results in multiple trials. The remaining steps are fully deterministic.

5.2. Classification results

In Table 1, we present classification accuracy obtained by the proposed method. The table contains average accuracy, achieved in ten repetitions of the classifier training procedure, and the best outcome. Results obtained with the SVM algorithm are compared with the RF algorithm. The table allows comparing accuracy when models were constructed using the full set of features with models trained on subsets of features selected with the best first search method. Results concern 30 tested data sets, ordered alphabetically.

On the full set of features, RF classifiers performed better than SVM classifiers. This is explained by the fact, that the RF algorithm performs an internal feature evaluation procedure when constructing its trees. For the data sets processed in the experiments, RF-based models do not need to be accompanied by a feature selection procedure. In all but three cases, RFs constructed on the full set of features outperformed RFs constructed on thinned sets of features. It shall be reiterated, that the feature selection procedure was a wrapper-based, best-first search with an SVM as an evaluator.

In contrast, SVM performs generally better when we use thinned sets of features. For 14 out of 30 data sets, this strategy produced a higher accuracy, in comparison to SVMs constructed on full sets of features.

In general, the SVM-based classification produced better results. In 16 out of 30 data sets, SVM gave a higher accuracy of prediction than RF.

It is worth to mention, that in many cases, the best result is only slightly different from the average result, achieved in ten repetitions of the experiment.

5.3. Strawberry – results inspection example

In the following paragraphs, we demonstrate an extended validation scenario for the results obtained for one selected set, Strawberry. The data set contains instances that belong to two classes labeled as 1 and 2. We investigate one particular classification experiment executed with the Random Forest algorithm. It was performed based on a subset of features. Quality evaluation was performed on the test set that was made of 613 instances. The confusion matrix of this experiment's results is placed below.

		true labels	
		1	2
predicted labels	1	149	27
	2	70	367

Let us recall that in a confusion matrix as above we collect information about classification efficiency. On the principal diagonal, we see true positives (TP) and true negatives (TN) – data that were correctly classified. On the anti-diagonal, we place false negatives (FN) and false positives (FP) – counts of samples that were not recognized correctly. Based on the content of the confusion matrices, we can compute elementary measures such as accuracy, sensitivity, specificity, precision, and F1-score. Sensitivity is computed as a ratio of true positives to all true positives and false negatives (TP/(TP + FN)). Precision is a ratio of true positives to true positives plus false positives (TP/(TP + FP)). Specificity is computed by dividing true negatives by true negatives plus false positives (TN/(TN + FP)). F1-score is obtained by computing the harmonic mean of precision and sensitivity, which resolves to 2TP/(2TP + FP + FN). In this case, classifier sensitivity is equal to 0.68, precision to 0.85, accuracy to 0.84, specificity to 0.93, and F1-score to 0.75.

Subsequently, we may apply several visual techniques for results inspection. We have used the pROC R package to plot Receiver Operating Characteristic (ROC) curves. The package smooths the plots to obtain good-looking outcomes. A ROC plot presents specificity against sensitivity. Fig. 4a presents the ROC curve for the discussed classifier. Based on the shape of the curve, we may pose a conclusion that samples of both classes are recognized with similar efficiency and there is no tendency to recognize one class better.

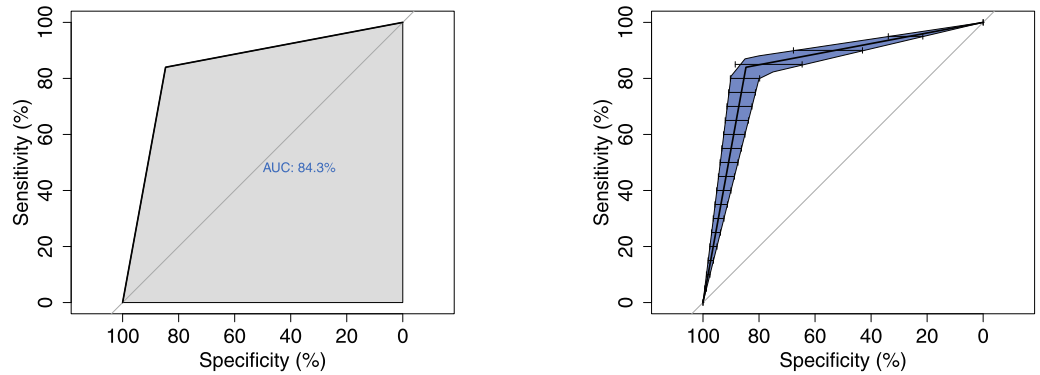
Additional validation of the ROC would be to plot its confidence intervals. A confidence interval for the ROC Curve represents the range at a given point of specificity and its corresponding sensitivity. This is presented in Fig. 4b. Very wide intervals may suggest issues with the model. This is not the case in the displayed example, where we do not see anything alarming.

5.4. Comparison with other methods

Subsequently, we compare results achieved with the proposed procedure with selected literature-based methods. The comparison concerns two kinds of algorithms. Firstly, we study the performance of two standard classifiers: Naive Bayes (NB) and Quinlan's C4.5 decision tree. Secondly, we compare top-tier time series classification methods: a DTW Features, Shapelet Transform (ST), Learned Shapelets (LS), and one variant of the COTE method (HIVE-COTE). DTW Features, proposed by Kate

Table 1
Classification accuracy, expressed in %, provided by the proposed method on a suite of real-world data sets. Results were achieved using the proposed method with SVM and Random Forest (RF) as classifiers. The table contains average results on test sets, based on ten runs of the same experiment. The best result for each case is given.

Data set	Classification accuracy, full set of features				Classif. accuracy, reduced set of features			
	SVM		RF		SVM		RF	
	Avg.	Max	Avg.	Max	Avg.	Max	Avg.	Max
Beef	94.00%	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**	**100%**
BeetleFly	78.00%	**85.00%**	78.00%	80.00%	81.50%	**85.00%**	78.50%	80.00%
BirdChicken	87.00%	**95.00%**	87.00%	90.00%	83.00%	**95.00%**	83.00%	**95.00%**
CBF	86.69%	87.11%	90.24%	**90.89%**	85.32%	89.22%	81.21%	84.56%
CinC_ECG_torso	61.14%	66.81%	67.64%	**68.41%**	61.63%	65.80%	58.53%	66.59%
Coffee	92.86%	92.86%	90.36%	92.86%	**100%**	**100%**	**100%**	**100%**
Computers	63.76%	66.80%	64.12%	65.20%	66.88%	**67.20%**	66.72%	**67.20%**
DiatomSizeReduction	80.23%	**80.72%**	74.35%	75.16%	75.56%	76.14%	72.35%	72.55%
Dist.Phal.OutlineAgeGr.	81.55%	**83.00%**	82.08%	82.75%	81.63%	82.75%	81.75%	82.75%
Dist.PhalanxOutlineCor.	79.23%	79.83%	81.22%	**82.17%**	77.47%	77.67%	77.05%	77.83%
DistalPhalanxTW	74.50%	78.00%	77.70%	**78.25%**	77.85%	**78.25%**	75.85%	76.00%
Herring	60.63%	62.5%	57.66%	60.94%	67.34%	**68.75%**	62.50%	65.63%
Mallat	80.34%	**84.82%**	80.34%	80.34%	71.01%	72.71%	73.10%	73.43%
Mid.Phal.OutlineCorrect	56.67%	60.33%	55.95%	57.67%	67.47%	**69.50%**	66.20%	68.00%
MiddlePhalanxTW	61.58%	62.66%	61.88%	**62.91%**	61.70%	**62.91%**	60.93%	61.90%
ShapeletSim	83.61%	85.56%	96.50%	**97.22%**	**97.22%**	**97.22%**	96.56%	96.67%
SmallKitchenAppliances	67.95%	68.00%	69.63%	**70.40%**	69.23%	69.33%	67.57%	70.13%
SonyAIBORobotSurf.1	84.36%	**87.19%**	81.26%	81.36%	81.26%	81.36%	81.26%	81.36%
SonyAIBORobotSurf.2	89.65%	89.82%	91.03%	**91.50%**	87.82%	89.61%	86.56%	88.77%
StarLightCurves	94.91%	**95.14%**	94.51%	94.74%	94.34%	94.39%	93.88%	93.88%
Strawberry	84.65%	**85.64%**	82.89%	83.52%	84.47%	84.50%	83.52%	83.52%
SwedishLeaf	74.48%	76.48%	78.82%	79.68%	79.33%	**80.64%**	76.59%	77.76%
ToeSegmentation1	72.76%	**75.88%**	73.29%	75.44%	65.70%	66.23%	64.74%	65.79%
ToeSegmentation2	64.69%	**74.62%**	61.00%	63.85%	67.54%	68.46%	67.38%	68.46%
Trace	97.10%	99.00%	**100%**	**100%**	99.00%	99.00%	89.10%	99.00%
Two_Patterns	73.80%	74.38%	74.58%	74.93%	75.13%	**75.20%**	74.92%	74.98%
TwoLeadECG	93.44%	**93.77%**	85.23%	86.57%	88.17%	90.87%	83.84%	84.46%
wafer	97.51%	**97.65%**	96.26%	96.30%	96.78%	97.08%	96.17%	96.28%
Wine	50.19%	51.85%	51.67%	55.56%	63.15%	**68.52%**	57.04%	59.26%
WormsTwoClass	60.22%	61.33%	64.64%	**65.75%**	65.19%	65.19%	63.59%	64.09%



(a) ROC for the analysed classifier.

(b) Confidence intervals of specificity for the ROC.

Fig. 4. Strawberry data set – ROC for random forest classifier.

(2016), combines DTW distances and SAX histograms. Thus, it is a highly efficient fusion of two well-known time series classification approaches. DTW Features was published in 2016. Shapelet Transform is an algorithm based on the so-called shapelets. A shapelet is a time series subsequence, which allows distinguishing between time series belonging to different classes. In the Shapelet Transform algorithm, searching for shapelets is performed as a separate procedure, preceding classification. Learned Shapelets (Grabocka et al., 2014) is a shapelet discovery algorithm that utilizes a heuristic gradient descent shapelet search procedure. HIVE-COTE, introduced by Lines et al. (2018), is an ensemble method, that combines 35 classifiers over four data representations. HIVE-COTE at the moment of writing this article is one of the best-performing algorithms for time series classification.

Fig. 5 allows inspecting classification accuracy of the introduced method, denoted on the plot as PAINTS, versus the other six approaches. The accuracy achieved by the literature approaches is reported in the cited papers and the repository (Bagnall et al., 2019). The figure does not distinguish between numerical results achieved by different literature methods. Instead, we plot the best and the worst outcome in the form of two lines (dashed and solid). The area between the worst and the best accuracy achieved by the literature methods is marked with a lightly dotted background. Results achieved with the PAINTS method are marked with black dots.

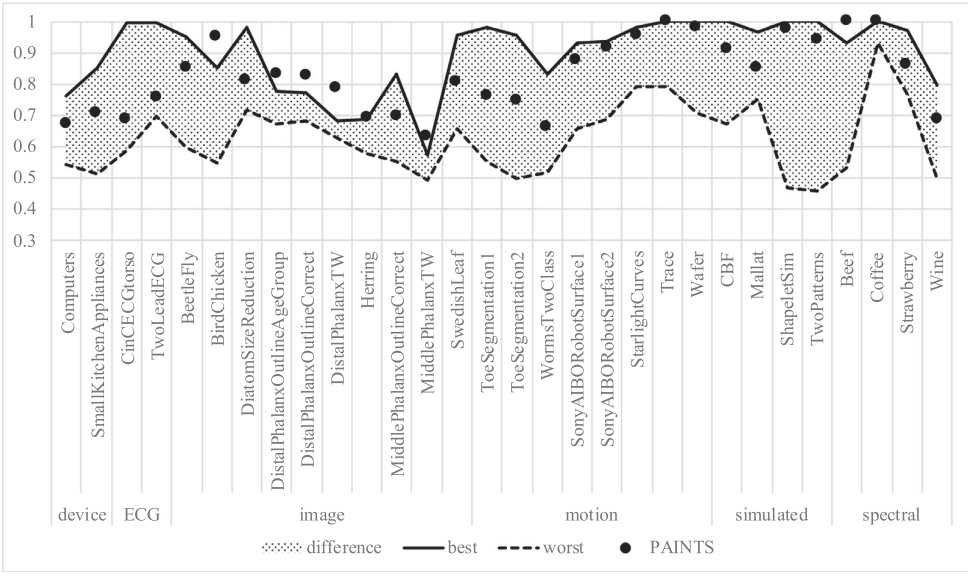


Fig. 5. Classification accuracy achieved by the method introduced in this paper (denoted as PAINTS) versus accuracy of the selected six literature approaches. The data sets are grouped by the domain from which they were collected (device, ECG, image, motion, sensor, simulated, spectral). The order of data sets within one domain is alphabetical.

It must be stressed, that in all cases, the proposed method does not perform worse than the selected state-of-the-art methods. In six cases (Beef, BirdChicken, DistalPhalanxOutlineAgeGroup, DistalPhalanxOutlineCorrect, DistalPhalanxTW, MiddlePhalanxTW), the introduced method outperformed all competitors. In further three cases (Coffee, Herring, Trace), the developed method was exactly as accurate as of the best literature method.

Let us draw attention to the type of processed data sets, which is denoted on the bottom row of the horizontal axis in Fig. 5. It is a very interesting observation, that the proposed method performs best when the time series classification data set originates from images. In this case, we achieved a noticeable advantage over all literature competitors.

We inspect the results of relevant statistical tests suited for experiments on comparing multiple algorithms on multiple problems to validate the soundness of the executed comparisons. At first, we applied two non-parametric methods for algorithm comparisons, the Friedman test and its modification by Iman and Davenport (1980). The obtained p-values were low what suggested that some of the tested algorithms may perform similarly. Thus, we moved to more detailed comparisons. Fig. 6 presents a critical difference diagram. It illustrates a comparison of all used time series classifiers against each other with the Nemenyi test. Groups of classifiers that are not significantly different (at the significance level equal to 0.05) are connected. We see that the two tested standard classifiers (Naive Bayes and C4.5) deliver not significantly different outcomes. According to this test, on the examined data sets, the introduced method produced results coinciding with the results of the DTW Features, Learned Shapelets, and Shapelet Transform algorithms.

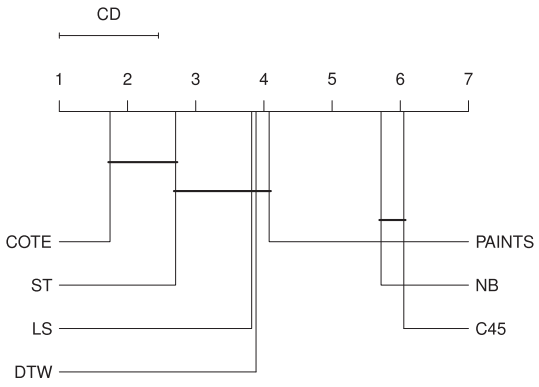


Fig. 6. Critical difference diagram for the classification of 30 data sets using the introduced method (PAINTS), two regular classifiers: Naive Bayes (NB) and C4.5 and four top-level time series classification methods: DTW Features (DTW), Shapelet Transform (ST), Learned Shapelets (LS), HIVE-COTE (COTE).

Garcia and Herrera (2008) suggest applying the Bergmann and Hommel's method of the dynamic correction of p-values. We use this method to perform a more detailed inspection of the pairwise differences between algorithms. The Bergmann-Hommel post hoc procedure uses the Friedman Aligned Ranks Test p-values. It applies a correction based on a list of possible hypotheses for multiple testing. The results of this test are displayed in Fig. 7.

According to the p-values presented in Fig. 7, we can conclude that at the significance level equal to 0.05 there is a significant difference between the introduced method and four out of six tested algorithms. Results achieved with the introduced method coincide with the DTW Features and Learned Shapelets algorithms. There is no significant difference between the two standard classifiers (Naive Bayes and C4.5). Similarly, in the conducted experiment, there was no difference between results achieved with DTW Features and Learned Shapelets.

6. Conclusion

In this paper, a new method for time series classification was introduced. The approach contributes an entirely novel concept, in which the time series classification problem is transformed into the visual pattern recognition problem.

A series of experiments on real-world data sets has shown, that the method performs very well. The lengths of time series do not have to be the same. Besides, it is well-suited for large-scale data sets (with long time series). Interestingly, the method achieves very good results for time series that were extracted from images.

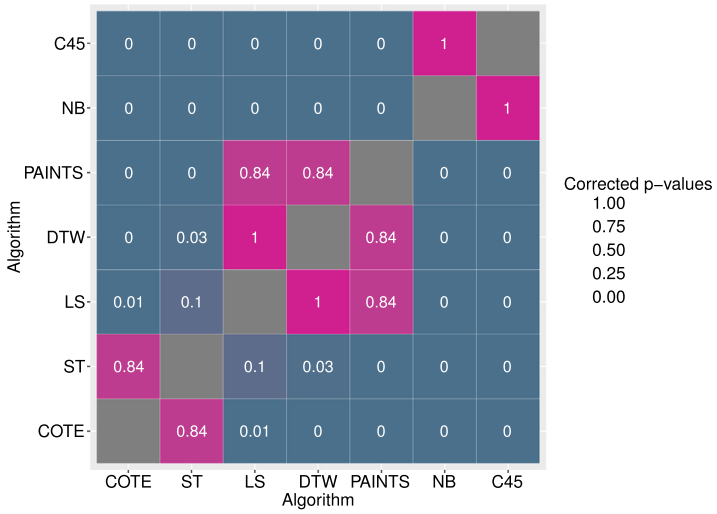


Fig. 7. The p-value matrix generated when doing all the pairwise comparisons in the Bergmann and Hommel's method.

A comparison with a couple of standard classifiers and six top-level time series classifiers has shown that the new method brings in a new quality. It is worth to emphasize, that the method does not require any heuristic optimizations or greedy searches and it can be implemented in a robust, stable, and fully deterministic manner.

In the future, we plan to apply neural networks to implement the visual pattern recognition step. Also, we will test various configurations of graphical parameters and their influence on classification accuracy.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Appendix A. Statistics of the processed time series

See Table A.2.

Table A.2
Elementary information about processed time series.

Data set name	Number of classes	Length	Train size	Test size	Stats on train set min	Stats on train set max	Stats on test set min	Stats on test set max	Type
Beef	5	470	30	30	−3.29	3.72	−3.39	3.15	spectro
BeetleFly	2	512	20	20	−2.52	2.51	−2.51	2.41	image
BirdChicken	2	512	20	20	−2.82	2.12	−3.10	2.44	image
CBF	3	128	30	900	−2.32	3.24	−3.55	3.79	simulated
CinCECGtorso	4	1639	40	1380	−8.59	10.54	−11.21	11.73	ECG
Coffee	2	286	28	28	−2.06	2.18	−2.12	2.10	spectro
Computers	2	720	250	250	−3.75	21.60	−1.61	26.39	device
DiatomSizeReduction	4	345	16	306	−1.77	1.98	−1.98	2.45	image
Dist.PhalanxOutlineAgeGr.	3	80	139	400	−1.91	2.03	−1.99	2.06	image
Dist.PhalanxOutlineCor.	2	80	276	600	−2.18	2.46	−2.16	2.45	image
DistalPhalanxTW	6	80	139	400	−1.90	2.00	−1.99	2.06	image
Herring	2	512	64	64	−2.19	2.13	−2.21	2.07	image
Mallat	8	1024	55	2345	−1.61	2.76	−1.70	2.94	simulated
Mid.PhalanxOutlineCor.	2	80	600	291	−1.72	1.88	−1.66	2.07	image
Mid.PhalanxTW	6	80	154	399	−1.58	1.71	−1.72	1.92	image
ShapeletSim	2	500	20	180	−1.81	1.89	−1.87	1.87	simulated
SmallKitchenAppliances	3	720	375	375	−5.07	26.80	−3.87	26.80	device
SonyAIBORobotSurf.1	2	70	20	601	−2.73	3.63	−3.63	4.00	sensor
SonyAIBORobotSurf.2	2	65	27	953	−4.14	4.23	−4.02	4.50	sensor
StarLightCurves	3	1024	1000	8236	−2.68	5.29	−2.62	5.46	sensor
Strawberry	2	235	370	613	−2.13	3.72	−2.33	3.68	spectro
SwedishLeaf	15	128	500	625	−3.41	3.22	−2.94	3.29	image
ToeSegmentation1	2	277	40	277	−3.58	3.93	−6.68	4.82	motion
ToeSegmentation2	2	343	36	130	−2.64	3.93	−3.68	5.55	motion
Trace	4	275	100	100	−2.22	3.97	−2.39	3.94	sensor
TwoLeadECG	2	82	23	1139	−3.15	1.87	−3.80	1.93	ECG
TwoPatterns	4	128	1000	4000	−1.94	1.94	−1.93	1.92	simulated
Wafer	2	152	1000	6164	−3.05	11.79	−2.98	12.13	sensor
Wine	2	234	57	54	−1.94	3.20	−1.92	3.19	spectro
WormsTwoClass	2	900	181	77	−4.89	4.20	−4.31	4.86	motion
hline									

References

Abdiansah, A., Wardoyo, R., 2015. Time complexity analysis of support vector machines (svm) in libsvm. Int. J. Comput. Appl. 128 (3), 28–34. Published by Foundation of Computer Science (FCS), NY, USA..

Amri, A.A., Ismail, A.R., Zarir, A.A., 2018. Comparative performance of deep learning and machine learning algorithms on imbalanced handwritten data. Int. J. Adv. Comput. Sci. Appl. 9 (2).

Atluri, G., Karpatne, A., Kumar, V., 2018. Spatio-temporal data mining: a survey of problems and methods. ACM Comput. Surv. 51 (4). 83:1–83:41.

Bagnall, A., Janacek, G., 2014. A run length transformation for discriminating between auto regressive time series. J. Classif. 31 (2), 154–178.

Bagnall, A., Lines, J., Hills, J., Bostrom, A., 2015. Time-series classification with COTE: the collective of transformation-based ensembles. IEEE Trans. Knowl. Data Eng. 27 (9), 2522–2535.

Bagnall, A., Lines, J., Vickers, W., and Keogh, E., 2019. The UEA & UCR time series classification repository..

Baydogan, M.G., Runger, G., 2016. Time series representation and similarity based on local autopatterns. Data Min. Knowl. Disc. 30 (2), 476–509.

Baydogan, M.G., Runger, G., Tuv, E., 2013. A bag-of-features framework to classify time series. IEEE Trans. Pattern Anal. Mach. Intell. 35 (11), 2796–2802.

Breiman, L., 2001. Random forests. Mach. Learn. 45 (1), 5–32.

Cortes, C., Vapnik, V., 1995. Support-vector networks. Mach. Learn. 20 (3), 273–297.

Dau, H.A., Silva, D.F., Petitjean, F., Forestier, G., Bagnall, A., Mueen, A., Keogh, E., 2018. Optimizing dynamic time warping's window width for time series data mining applications. Data Min. Knowl. Disc. 32 (4), 1074–1120.

Garcia, S., Herrera, F., 2008. An extension on "Statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons. J. Mach. Learn. Res. 9, 2677–2694.

Gorecki, T., Luczak, M., 2014. Non-isometric transforms in time series classification using DTW. Knowl.-Based Syst. 61, 98–108.

Grabocka, J., Schilling, N., Wistuba, M., Schmidt-Thieme, L., 2014. Learning time-series shapelets. Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'14. ACM, New York, NY, USA, pp. 392–401.

Grabocka, J., Wistuba, M., Schmidt-Thieme, L., 2015. Scalable classification of repetitive time series through frequencies of local polynomials. IEEE Trans. Knowl. Data Eng. 27 (6), 1683–1695.

Hatami, N., Gavet, Y., Debayle, J., 2018. Classification of time-series images using deep convolutional neural networks. In: Proceedings of the 10th International Conference on Machine Vision (ICMV 2017), vol. 10696. pp. 106960Y-1-106960Y-8..

Homenda, W., Pedrycz, W., 2018. Pattern Recognition: A Quality of Data Perspective. John Wiley & Sons Inc..

Hu, W., Huang, Y., Wei, L., Zhang, F., Li, H., 2015. Deep convolutional neural networks for hyperspectral image classification..

Ignatov, A., 2018. Real-time human activity recognition from accelerometer data using convolutional neural networks. Appl. Soft Comput. 62, 915–922.

Iman, R.L., Davenport, J.M., 1980. Approximations of the critical region of the Friedman statistic. Commun. Stat. Theory Methods 9 (6), 571–595.

Kate, R.J., 2016. Using dynamic time warping distances as features for improved time series classification. Data Min. Knowl. Disc. 30 (2), 283–312.

Li, W., Wu, G., Zhang, F., Du, Q., 2017. Hyperspectral image classification using deep pixel-pair features. IEEE Trans. Geosci. Remote Sens. 55 (2), 844–853.

Lines, J., Taylor, S., Bagnall, A., 2018. Time series classification with HIVE-COTE: the hierarchical vote collective of transformation-based ensembles. ACM Trans. Knowl. Disc. Data 12 (5). 52:1–52:35.

Livni, R., Shalev-Shwartz, S., Shamir, O., 2014. On the computational efficiency of training neural networks. Proceedings of the 27th International Conference on Neural Information Processing Systems – Volume 1, NIPS'14. MIT Press, Cambridge, MA, USA, pp. 855–863.

Luo, Y., Wen, Y., Tao, D., Gui, J., Xu, C., 2016. Large margin multi-modal multi-task feature extraction for image classification. IEEE Trans. Image Process. 25 (1), 414–427.

Molina, M.E., Perez, A., Valente, J.P., 2016. Classification of auditory brainstem responses through symbolic pattern discovery. Artif. Intell. Med. 70, 12–30.

Montgomery, J.S., Hopkinson, C., Brisco, B., Patterson, S., Rood, S.B., 2018. Wetland hydroperiod classification in the western prairies using multitemporal synthetic aperture radar. Hydrol. Process. 32 (10), 1476–1490.

Mori, U., Mendiburu, A., Lozano, J.A., 2016. Similarity measure selection for clustering time series databases. IEEE Trans. Knowl. Data Eng. 28 (1), 181–195.

Muro, J., Strauch, A., Heinemann, S., Steinbach, S., Thonfeld, F., Waske, B., Diekkruger, B., 2018. Land surface temperature trends as indicator of land use changes in wetlands. Int. J. Appl. Earth Obs. Geoinf. 70, 62–71.

O'Reilly, C., Moessner, K., Nati, M., 2017. Univariate and multivariate time series manifold learning. Knowl.-Based Syst. 133, 1–16.

Silva, D.F., Souza, V.M.A.D., Batista, G.E.A.P.A., 2013. Time series classification using compression distance of recurrence plots. In: 2013 IEEE 13th International Conference on Data Mining, pp. 687–696.

Stefan, A., Athitsos, V., Das, G., 2013. The move-split-merge metric for time series. IEEE Trans. Knowl. Data Eng. 25 (6), 1425–1438.

van Noord, N., Postma, E., 2017. Learning scale-variant and scale-invariant features for deep image classification. Pattern Recogn. 61, 583–592.

Wang, Z., Oates, T., 2015. Imaging time-series to improve classification and imputation. Proceedings of the 24th International Conference on Artificial Intelligence, IJCAI'15. AAAI Press, pp. 3939–3945.

Wang, X., Yu, F., Pedrycz, W., 2016. An area-based shape distance measure of time series. Appl. Soft Comput. 48, 650–659.

Xu, Y., Li, Z., Zhang, B., Yang, J., You, J., 2017. Sample diversity, representation effectiveness and robust dictionary learning for face recognition. Inf. Sci. 375, 171–182.

Zhou, P., Lin, Z., Zhang, C., 2016. Integrated low-rank-based discriminative feature learning for recognition. IEEE Trans. Neural Netw. Learn. Syst. 27 (5), 1080–1093.