Research Article

Time series classification: nearest neighbor versus deep learning models

Weiwei Jiang[1]

Received: 8 January 2020 / Accepted: 12 March 2020 / Published online: 20 March 2020

Abstract
Time series classification has been an important and challenging research task. In different domains, time series show different patterns, which makes it difficult to design a global optimal solution and requires a comprehensive evaluation of different classifiers across multiple datasets. With the rise of big data and cloud computing, deep learning models, especially deep neural networks, arise as a new paradigm for many problems, including image classification, object detection and natural language processing. In recent years, deep learning models are also applied for time series classification and show superiority over traditional models. However, the previous evaluation is usually limited to a small number of datasets and lack of significance analysis. In this study, we give a comprehensive comparison between nearest neighbor and deep learning models. Specifically, we compare 1-NN classifiers with eight different distance measures and three state-of-the-art deep learning models on 128 time series datasets. Our results indicate that deep learning models are not significantly better than 1-NN classifiers with edit distance with real penalty and dynamic time warping.

Keywords Time series classification · Nearest neighbor · Deep learning

1 Introduction

A time series is the data set arranged at sequential time intervals [22]. It is universally embedded in many fields and applications, from air pollution to electricity consumption, from earthquake to electrocardiogram, and from traffic readings to psychological signals [11, 27, 28, 40]. A time series can be univariate, where a sequence of measurements from the same variable are collected, or multivariate, where a sequence of measurements from multiple variables are collected [37].

Due to its generality, time series has been studied heavily in both academia and industry, mainly in three categories, namely, forecasting/prediction, clustering, and classification as follows:

- Time series forecasting/prediction: the task is to predict the value in the next time period, given historical data. For example, in a financial market, historical prices can be used to build prediction models for next day's price, which guides the trading decision and a better prediction contributes to a higher profit.
- Time series clustering: the task is group the unlabeled time series into different clusters, using an unsupervised learning approach. For example, given temperature data, different clime types are grouped and defined afterwards.
- Time series classification: the task is to build classifiers that could automatically decide the types of new series, given labeled time series as training data using an supervised learning approach. For example, ECG records the electrical voltage in the heart and is used to determine whether the heart is performing normally or suffering from abnormalities.

Electronic supplementary material The online version of this article (https://doi.org/10.1007/s42452-020-2506-9) contains supplementary material, which is available to authorized users.

✉ Weiwei Jiang, jiangweiwei@mail.tsinghua.edu.cn | [1]Department of Electronic Engineering, Tsinghua University, Beijing 100084, China.

In this study, we would focus on the univariate time series classification problem. Traditional time series classification methods can be classified into three categories:

- Distance-based methods: With some pre-defined similarity measures, e.g., Euclidean distance or Dynamic time warping, k-nearest-neighbors classifier is known to be the representation of the distance-based methods [30], which we would use in this study.
- Feature-based methods: these methods extract a set of features that are able to represent the global/local time series patterns, represented by Bag-of-Words (Bow) [32], Bag-of-features framework (TSBF) [5], Bag-of-SFA-Symbols (BOSS) [39].
- Ensemble-based methods: these methods combine different classifiers together to achieve a higher accuracy, for example, Elastic Ensemble (PROP) [33] combines 11 classifiers based on elastic distance measures with a weighted ensemble scheme. Shapelet ensemble (SE) [4] produces the classifiers through the shapelet transform in conjunction with a heterogeneous ensemble.

Recently, with the rise of big data, cloud computing and GPU-based acceleration, deep neural networks, including convolutional neural networks and residual networks, have achieved a great success for many problems, e.g., image classification [23], object detection [38], traffic forecasting [25], and handwritten numeral recognition [26]. Neural networks have also been applied to time series classification, e.g., a multi-channel CNN (MC-CNN) is proposed for multivariate time series classification [45], and a multi-scale CNN approach is proposed for univariate time series classification [12].

The difficulty for deep learning models in time series classification lies in two aspects: diverse problem types and limited training data. The success of deep learning models has been powered by large amount of datasets, e.g., ImageNet [15], in the past years. Whether their advantages remain in time series classification when the data size is relatively small is still an open question.

In this study, we focus on a specific question that if deep learning models are better than distance-based models or not. To study this problem, we compare 1-NN classifiers with eight different distance measures and three state-of-the-art deep learning models on 128 time series datasets. Our contributions are summarized as follows:

- We present a comprehensive comparison between two major types of time series classification models on a large number of 128 time series datasets.
- Our result reveal that deep learning models perform better than distance-based models, albeit the performance difference is not significant.
- Our detailed results as we provide in the supplemental file can be used as baselines for further studies.

The rest of this paper is organized as follows. Section 2 is the related work. Sections 3 and 4 define the distance-based and deep learning models, respectively. Section 5 gives the description of the time series datasets we use in this study. Section 6 presents the experiments and the associated result analysis. We conclude this paper in Sect. 7.

2 Related work

In this part, we would give a short review of the related work, covering the models both in the distance-based and deep-learning approaches. As deep-learning based time series classification only begins to spring up in the past decade, there is little work comparing these two approaches, and we aim to fill the gap.

2.1 Distance-based time series classification

Distance-based classifiers are both intuitive and easy to implement for time series classification, while the key point is the choice of the distance definition, which leads to diverse classification performances on different datasets. With this consideration, previous studies usually evaluate several distance measures simultaneously on as many datasets as possible.

Wang et al. [42] conducts an extensive experimental study re-implementing eight different time series representations and nine similarity measures and their variants, and testing their effectiveness on 38 time series data sets from a wide variety of application domains. The similarity measures include L_p-norms [44] (including L_1-norm, e.g., Manhattan Distance, L_2-norm, e.g., Euclidean Distance [16], and L_{inf}-norm), DISSIM [19], Dynamic Time Warping (DTW) [6], Longest Common SubSequence (LCSS) [41], Edit Sequence on Real Sequence (EDR) [9], Swale [36], Edit Distance with Real Penalty (ERP) [8], Threshold query based similarity search (TQuEST) [2], and Spatial Assembling Distance (SpADe) [10]. Bagnall and Lines [3] gives a series of experiments to compare 1-NN classifiers with Euclidean and DTW distance to standard classifiers (including SVM and Random Forest) on 77 time series classification problems and finds that 1-NN with DTW is competitive with these standard classifiers, while 1-NN with Euclidean distance is fairly easy to beat. Jeong et al. [24] defines a weighted dynamic time warping (WDTW) measure, which achieves accuracy improvement for time series classification than both the DTW and Euclidean distances. Lines and Bagnall [33] defines an ensemble classifier of elastic

distance measures, which significantly outperforms the individual classifiers with one kind of distance measure. Furthermore, Abanda et al. [1] gives a comprehensive review of distance-based time series classification.

2.2 Deep learning for time series classification

Inspired by their success in computer vision and natural language processing tasks, deep learning models are applied in time series classification problems in the past few years, as we mentioned in the Introduction section.

Wang et al. [43] proposes a fully convolutional network as a strong baseline for time series classification, which outperforms a multilayer perceptron and is competitive with a much deeper residual network. Karim et al. [29] proposes a long short term memory fully convolutional network that augments the fully convolutional networks with LSTM sub-modules. Attention mechanism is also explored for performance improvement and decision process visualization. Grabocka and Schmidt-Thieme [21] proposes a measure named NeuralWarp that models the alignment of time-series indices in a deep representation space, by modeling a warping function as an upper level neural network between deeply-encoded time series values. Inspired by the Inception-v4 architecture used for computer vision tasks, Fawaz et al. [18] proposes InceptionTime as an ensemble of deep convolutional neural network models and achieves a higher accuracy with a less time consumption, compared with the HIVE-COTE algorithm. Ma et al. [34] proposes a end-to-end framework called the Echo Memory Network, which uses echo state networks to learn the time series dynamics and multi-scale discriminative features.

Deep learning models are also applied in multivariate time series classification. Che et al. [7] proposes a metric learning model named Deep ExpeCted Alignment DistancE (DECADE) for multivariate time series, which yields a valid distance metric for time series with unequal lengths by sampling from an innovative alignment mechanism and captures complex temporal multivariate dependencies in local representation learned by deep networks. Furthermore, Fawaz et al. [17] gives a comprehensive review of deep learning models for time series classification.

3 Distance-based model

3.1 Nearest neighbor

The nearest neighbor algorithm leverages the similarity between different data samples and for a new data sample, the algorithm finds a predefined number (usually denoted as *k*) of training samples closest in distance to the new sample, and predict the label from these known samples. The algorithm is shown in Algorithm 1. Then we define the different distance measures in the next part and choose *k* as 1 for this study.

Algorithm 1 k-Nearest Neighbor

Classify (X, Y, x), where X is the training data with size m, Y is the class labels of X, x is an unknown new sample
for $i = 1$ to m **do**
  Compute distance $d(X_i, x)$
end for
Compute set I containing indices for the k smallest distances $d(X_i, x)$
return majority label for Y_i, where $i \in I$

3.2 Distance definitions

In this section, we give the definitions of the eight different distance measures used in this study. We classify these measures into three types, *lock-step measure*, *elastic measure*, and *threshold-based measure*, where lock-step measures compare the *i*th point of one time series to the *i*th point of another, thus requiring that two time series have exactly the same length, while elastic measures allow comparison of one-to-many points and one-to-none points. The summary of the measures and their abbreviations used in this study are as follows:

- Lock-step Measure
  - Manhattan Distance (Manhattan)
  - Euclidean Distance (Euclidean)
  - Pearson's Correlation (Cor)

- Elastic Measure
  - Dynamic time warping (DTW)
  - Longest common subsequence (LCSS)
  - Edit sequence on real sequence (EDR)
  - Edit distance with real penalty (ERP)
- Threshold-based Measure
  - Threshold Query Based Similarity Search (TQuEST)

3.2.1 Manhattan distance

Manhattan distance is a L_1-norm distance, which is measured along axes at right angles. Given two time series $X = \{x_0, x_1, \ldots, x_{N-1}\}$ and $Y = \{y_0, y_1, \ldots, y_{N-1}\}$, their Manhattan distance is defined as follows:

$$Manhattan\,(X, Y) = \sum_{i=0}^{N-1} |x_i - y_i| \tag{1}$$

3.2.2 Euclidean distance

Euclidean distance [16] is a L_2-norm distance, which is the straight-line distance in Euclidean space. Given two time series $X = \{x_0, x_1, \ldots, x_{N-1}\}$ and $Y = \{y_0, y_1, \ldots, y_{N-1}\}$, their Euclidean distance is defined as follows:

$$Euclidean(X, Y) = \sqrt{\sum_{i=0}^{N-1} (x_i - y_i)^2} \tag{2}$$

3.2.3 Pearson's correlation

Pearson's correlation is a number between − 1 and 1, which measures the extent to which two variables are linearly related. Given two time series $X = \{x_0, x_1, \ldots, x_{N-1}\}$ and $Y = \{y_0, y_1, \ldots, y_{N-1}\}$, their Pearson's correlation is defined as follows:

$$Cor(X, Y) = \frac{\sum_{i=0}^{N-1} (x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=0}^{N-1} (x_i - \overline{x})^2} \sqrt{\sum_{i=0}^{N-1} (y_i - \overline{y})^2}} \tag{3}$$

where $\overline{x}$ and $\overline{y}$ represent the mean values of two time series.

3.2.4 Dynamic time warping

Dynamic time warping (DTW) [6] computes the optimal alignment between points of two time series with a window size, e.g., 100 used in this study. Given two time series $X = \{x_0, x_1, \ldots, x_{N-1}\}$ and $Y = \{y_0, y_1, \ldots, y_{M-1}\}$, their dynamic time warping is defined recursively as follows:

$$DTW(X, Y) = \begin{cases} 0 & \text{if} \quad M - 1 = N - 1 = 0 \\ \inf & \text{if} \quad M - 1 = 0 \text{ or } N - 1 = 0 \\ d(x_0, y_0) + \min\{DTW(Rest(X), Rest(Y)), & \\ DTW(Rest(X), Y), DTW(X, Rest(Y))\} & \text{otherwise} \end{cases} \tag{4}$$

where $d(\cdot)$ is Euclidean distance, $Rest(X) = \{x_1, \ldots, x_{N-1}\}$, $Rest(Y) = \{y_1, \ldots, y_{M-1}\}$.

3.2.5 Longest common subSequence

Longest common subsequence (LCSS) [41] finds the longest subsequence common to all sequences in two time series. Given two time series $X = \{x_0, x_1, \ldots, x_{N-1}\}$ and $Y = \{y_0, y_1, \ldots, y_{M-1}\}$, their longest common subsequence is defined recursively as follows:

$$LCSS(X, Y) = \begin{cases} 0 & \text{if} \quad M - 1 = 0 \text{ or } N - 1 = 0 \\ LCSS(Rest(X), Rest(Y)) + 1 & \text{if} \quad |x_0 - y_0| \le \epsilon \\ \max\{LCSS(Rest(X), Y), LCSS(X, Rest(Y))\} & \text{otherwise} \end{cases} \tag{5}$$

where *epsilon* is a threshold parameter, which we choose as 0.1 empirically, $Rest(X) = \{x_1, \ldots, x_{N-1}\}$, $Rest(Y) = \{y_1, \ldots, y_{M-1}\}$.

3.2.6 Edit sequence on real sequence

Edit sequence on real sequence (EDR) [9] counts the number of edit operations (insert, delete, replace) that are necessary to transform one time series into the other time series. Given two time series $X = \{x_0, x_1, \ldots, x_{N-1}\}$ and $Y = \{y_0, y_1, \ldots, y_{M-1}\}$, their edit sequence on real sequence is defined recursively as follows:

$$EDR(X,Y) = \begin{cases} N & \text{if} \quad M-1=0 \\ M & \text{if} \quad N-1=0 \\ \min\{EDR(Rest(X),Rest(Y)) + d_{edr}(x_0,y_0), \\ EDR(Rest(X),Y)+1, EDR(X,Rest(Y))+1\} & \text{otherwise} \end{cases} \tag{6}$$

where $Rest(X) = \{x_1, \ldots, x_{N-1}\}$, $Rest(Y) = \{y_1, \ldots, y_{M-1}\}$, $d_{edr}(x_0, y_0)$ is 0, if the Euclidean distance between x_0 and y_0 is below a threshold, which we choose as 0.1 empirically in this study, otherwise $d_{edr}(x_0, y_0)$ is 1.

3.2.7 Edit distance with real penalty

Edit distance with real penalty (ERP) [8] is a variant of edit distance and searches for the minimal path in a Euclidean distance matrix that describes the mapping between the two time series. Given two time series $X = \{x_0, x_1, \ldots, x_{N-1}\}$ and $Y = \{y_0, y_1, \ldots, y_{M-1}\}$, their edit distance with real penalty is defined recursively as follows:

$$ERP(X,Y) = \begin{cases} \sum_{i=0}^{N-1} |y_i - g| & \text{if} \quad M-1=0 \\ \sum_{i=0}^{M-1} |x_i - g| & \text{if} \quad N-1=0 \\ \min\{ERP(Rest(X),Rest(Y)) + d(x_0,y_0), \\ ERP(Rest(X),Y) + d(x_0,g), ERP(X,Rest(Y)) + d(g,y_0)\} & \text{otherwise} \end{cases} \tag{7}$$

where $Rest(X) = \{x_1, \ldots, x_{N-1}\}$, $Rest(Y) = \{y_1, \ldots, y_{M-1}\}$, $d(\cdot)$ is Euclidean distance, g is a parameter, which we choose as 0 empirically in this study.

3.2.8 Threshold query based similarity search

Threshold query based similarity search (TQuEST) [2] represents the series based on a set of time intervals that fulfill some conditions including all values are above a predefined threshold, and the TQuEST distance between two time series is defined in terms of the similarity between their threshold passing interval sets. Specifically, given two time series X and Y, their threshold query based similarity search is defined as follows:

$$\begin{aligned} TQuEST(X,Y) = & \frac{1}{|S(X,\tau)|} \sum_{s \in S(X,\tau)} \min_{s' \in S(Y,\tau)} d(s,s') \\ & + \frac{1}{|S(Y,\tau)|} \sum_{s' \in S(Y,\tau)} \min_{s \in S(X,\tau)} d(s',s) \end{aligned} \tag{8}$$

where $S(X,\tau)$ and $S(Y,\tau)$ are sets of time intervals in which all the values are above a predefined threshold, e.g., 0.1 used in this study, and the distance between two time intervals $s = (s_l, s_u)$ and $s' = (s'_l, s'_u)$ is defined as $d(s,s') = \sqrt{(s_l - s'_l)^2 + (s_u - s'_u)^2}$.

4 Deep learning model

In this study, we use three deep learning models for time series classification. Previous study shows that a fully convolutional network is a strong baseline for time series classification, which outperforms a multi-layer perceptron and is competitive with a much deeper residual network [43].

4.1 Multi-layer perceptron

Multi-layer perceptron (MLP) or mutlti-layer nueral network is the most traditional form of deep neural networks, with multiple hidden layers between the input and output layers. With the standard back-propagation algorithm, the gradient of the loss function can be efficiently calculated and the weights can be set iteratively. To avoid overfitting in a deeper neural network, regularization methods and dropout operations are often used. In this study, we use a multi-layer perceptron with four layers, where each one is fully connected to the output of its previous layers, with a dropout operation with rate as 0.2. All the three hidden fully connected layers contain 500 neurons and ReLU is used as the activation function. The classification result is generated by the final layer as a softmax classifier.

4.2 Fully convolutional neural network

Convolutional neural network (CNN) is a special form of deep neural networks, which achieves a great success for two-dimensional image processing tasks. In the hidden layer of CNNs, each group of neurons (as a filter) shares the same weights and performs a convolution operation in different regions of the image. Max pooling is used in CNNs to reduce the original input size, which uses a max operation to choose the max value for each feature map.

CNNs were developed for image classification, in which the model accepts a two-dimensional input representing an image's pixels and color channels. Yet, 2D CNNs may not be a viable option in numerous applications over 1D

signal, e.g., univariate time series as we consider here. To address this issue, 1D CNNs have been proposed recently, which perform only 1D convolutions (scalar multiplications and additions). Also the max pooling operations are reduced to one dimension.

Instead of using max pooling for size reduction, fully convolutional neural networks (FCNs) do not contain any local pooling layers and the length of a time series is kept unchanged throughout the convolutions. In this study, we use a 1D fully convolutional neural network with three blocks. Each block is composed of a convolutional layer (we use 128, 256 and 128 filters with length as 3 for the three blocks) and a batch normalization layer with ReLU as the activation function. The result of the third block is connected to a global average pooling layer, which is further connected to a traditional softmax classifier.

4.3 Residual network

Residual network (ResNet) was originally proposed to solve image classification, which is featured by the shortcut residual connections between consecutive convolutional layers [23]. In this study, we use a residual network with three residual blocks. Each residual block consists of three convolutional layers (we use 64, 128 and 128 filters with length as 3 for the three layers), a batch normalization layer and the ReLU activation function. We choose the filter's length as 8, 5 and 3 respectively for the three residual blocks. The residual block's input is added to the output of the third convolutional layer and then fed to the next layer. Similar to the fully convolutional neural network, the result of the third residual block is connected to a global average pooling layer and a traditional softmax classifier, which generates the final classification result, and the convolutional layers and filters are also one-dimensional.

5 Data description

In this study, we use the latest UCR Time Series Classification Archive [13], which contains 128 datasets from 15 types, as shown in Table 1. The top three types with the largest number of datasets are Image (32), Sensor (30) and Motion (17).

The train versus test sample numbers are shown in Fig. 1. Most of the datasets have a relatively small size, e.g., less than 1000 train samples and less than 2000 test samples. This is different from the problems in computer vision, where millions of images are collected for training deep learning models with thousands to billions of parameters. With such a small number of training samples, it would raise a concern that the deep learning models would overfit the training samples and show a poor performance on test sets.

We also plot the distributions of class numbers and time series lengths of datasets with different types in Fig. 2a, b, respectively. Later we would evaluate how these factors would affect the performance of different classification models.

6 Experiment and analysis

6.1 Implementation

The 1-NN classifiers with different distance measures are implemented with an R package named *TSdist* [35], which provides a *OneNN* function and is convenient to switch among different distances. The deep learning models are implemented with Python packages including *fastai* [1] and *PyTorch* [2]. They are trained with Adam [31] with the learning rate 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$ and $\epsilon = 1e{-}08$. The loss function for all tested deep learning model is categorical cross entropy. We choose the best model that achieves the lowest training loss and apply it on the test set.

The experiments are conducted on a desktop computer, which has a Windows 10 operating system, 16GB random-access memory (RAM) and Intel core i5-9600K central processing unit (CPU). A graphical accelerated processing (GPU) of GeForce RTX 2070 with 8GB RAM is also equipped to accelerate the training process of the deep learning models.

6.2 Data preprocessing

For each dataset, we perform two standard preprocessing technologies:

Missing data fill-in with mean values;
Z-score normalization, which is calculated by subtracting the mean from an individual value and then dividing the difference by the standard deviation.

[1] https://www.fast.ai/.

[2] https://pytorch.org/.

Table 1 Dataset grouped by problem types, where ECG represents electrocardiography, EOG represents electrooculography, EPG represents electrical penetration graph, HRM represents high-resolution melt

Type (Count)	Dataset names	
Device (9)	Computers	ElectricDevices
	LargeKitchenAppliances	RefrigerationDevices
	ScreenType	SmallKitchenAppliances
	ACSF1	HouseTwenty
	PLAID	
ECG (6)	ECG200	ECG5000
	ECGFiveDays	NonInvasiveFetalECGThorax1
	NonInvasiveFetalECGThorax2	TwoLeadECG
EOG (2)	EOGHorizontalSignal	EOGVerticalSignal
EPG (2)	InsectEPGRegularTrain	InsectEPGSmallTrain
Hemodynamics (3)	PigAirwayPressure	PigArtPressure
	PigCVP	
HRM (1)	Fungi	
Image (32)	Adiac	ArrowHead
	BeetleFly	BirdChicken
	DiatomSizeReduction	DistalPhalanxOutlineAgeGroup
	DistalPhalanxOutlineCorrect	DistalPhalanxTW
	FaceAll	FaceFour
	FacesUCR	FiftyWords
	Fish	HandOutlines
	Herring	MedicalImages
	MiddlePhalanxOutlineAgeGroup	MiddlePhalanxOutlineCorrect
	MiddlePhalanxTW	OSULeaf
	PhalangesOutlinesCorrect	ProximalPhalanxOutlineAgeGroup
	ProximalPhalanxOutlineCorrect	ProximalPhalanxTW
	ShapesAll	SwedishLeaf
	Symbols	WordSynonyms
	Yoga	Crop
	MixedShapesRegularTrain	MixedShapesSmallTrain
Motion (17)	CricketX	CricketY
	CricketZ	GunPoint
	Haptics	InlineSkate
	ToeSegmentation1	ToeSegmentation2
	UWaveGestureLibraryAll	UWaveGestureLibraryX
	UWaveGestureLibraryY	UWaveGestureLibraryZ
	Worms	WormsTwoClass
	GunPointAgeSpan	GunPointMaleVersusFemale
	GunPointOldVersusYoung	
Power (1)	PowerCons	

Table 1 (continued)

Type (Count)	Dataset names	
Sensor (30)	Car	ChlorineConcentration
	CinCECGTorso	Earthquakes
	FordA	FordB
	InsectWingbeatSound	ItalyPowerDemand
	Lightning2	Lightning7
	MoteStrain	Phoneme
	Plane	SonyAIBORobotSurface1
	SonyAIBORobotSurface2	StarLightCurves
	Trace	Wafer
	AllGestureWiimoteX	AllGestureWiimoteY
	AllGestureWiimoteZ	DodgerLoopDay
	DodgerLoopGame	DodgerLoopWeekend
	FreezerRegularTrain	FreezerSmallTrain
	GesturePebbleZ1	GesturePebbleZ2
	PickupGestureWiimoteZ	ShakeGestureWiimoteZ
Simulated (8)	CBF	Mallat
	ShapeletSim	SyntheticControl
	TwoPatterns	BME
	SmoothSubspace	UMD
Spectro (8)	Beef	Coffee
	Ham	Meat
	OliveOil	Strawberry
	Wine	EthanolLevel
Spectrum (4)	Rock	SemgHandGenderCh2
	SemgHandMovementCh2	SemgHandSubjectCh2
Traffic (2)	Chinatown	MelbournePedestrian
Trajectory (3)	GestureMidAirD1	GestureMidAirD2
	GestureMidAirD3	

Since all the deep learning models take one-dimensional time series data as input, there is no need for 1D-to-2D reshape in the preprocessing step.

6.3 Evaluation metrics

The evaluation metric for a single dataset would be the accuracy/error ratio on the test set, which corresponds to the ratio of correctly/wrongly classified samples. For deep learning models, we use the mean accuracy averaged over 10 runs and each run takes 40 epochs. The accuracy/error ratio could be used to evaluate the performance of the classifiers on a single dataset. However, to comprehensively evaluate the classifier's performance across different datasets, we propose to use three further evaluation metrics based on the accuracy for the individual datasets:

- win rate: win rate evaluates the ratio of datasets that a classifier is most accurate for. For example, if FCN beats every other classifier by earning a 100% accuracy on every dataset, then its win rate is 100%, while the others are 0%;
- average ranks: similar to win rate, but we use the rank value instead of only choosing the best one, e.g., the most accurate classifier would have a rank value as 1,

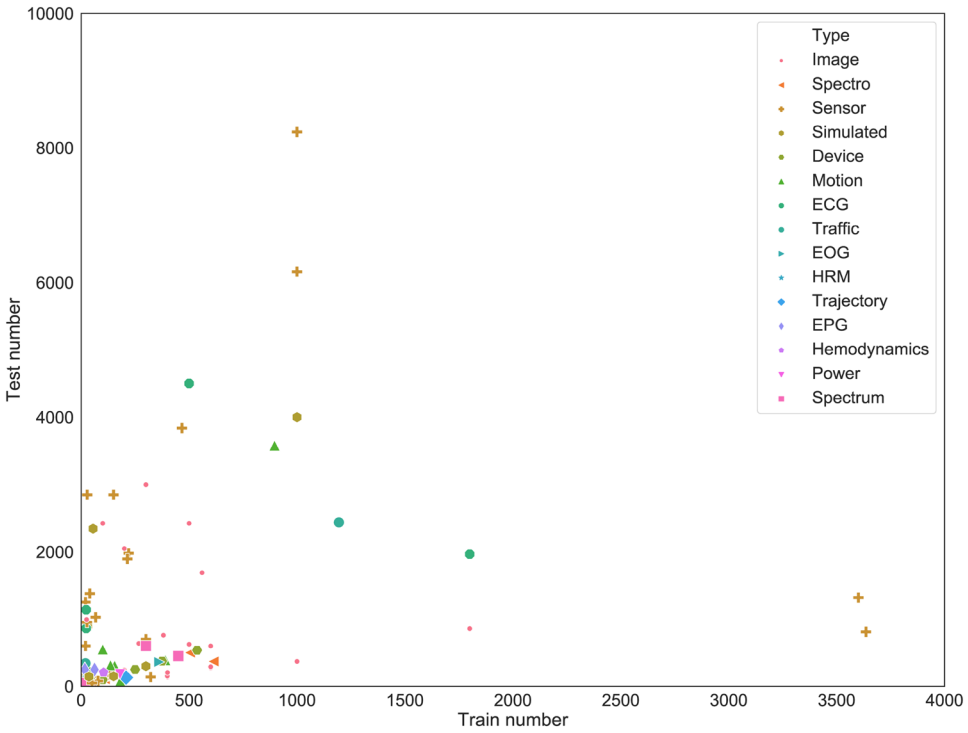

Fig. 1 The train versus test sample numbers of datasets with different types

and the second most accurate one would get a rank value as 2, etc. Then we take the average value for the ranks across all datasets;

– critical difference diagram: following Fawaz et al. [17], we perform the Wilcoxon signed-rank test with Holm's alpha (5%) correction [20] and use the critical difference diagram for visualization [14], in which a thick horizontal line shows a group of classifiers (a clique) that are not-significantly different in terms of accuracy.

6.4 Results

In this section, we discuss our experimental results based on the above evaluation metrics. We refer to the distance-based classifiers with their abbreviations and use FCN, MLP and ResNet to denote the three deep learning models.

6.4.1 Accuracy

The distribution of accuracy for different classifiers is shown in Fig. 3. It is not so obvious which classifier is much better than the others in Fig. 3.

6.4.2 Win rate

Win rates of different methods across the 128 datasets split by problem types in Table 1 is shown in Table 2. Overall ResNet has the highest win rate as 18.6% and keeps this advantage for datasets with type as Motion and Sensor, while EDR achieves the best win rate for datasets with type as Image.

6.4.3 Average ranks

Average ranks of different methods across the 128 datasets split by problem types in Table 1 is shown in Table 3. The result agrees with Table 1. We further evaluate the impact of different factors, namely, the training sizes (in Table 4), time length (in Table 5), class numbers (in Table 6). ResNet achieves the best average ranks with some exceptions:

- ERP is the best for datasets with a training size less than 100 and a length less than 81;
- DTW is the best for datasets with a time length between 251 and 450 and a class number between 10 and 19.

The better performance of ERP on datasets with a smaller training size and a shorter time length is reasonable, since deep learning models tend to perform better with a large amount of training data. The better performance of DTW over deep learning models needs further investigation.

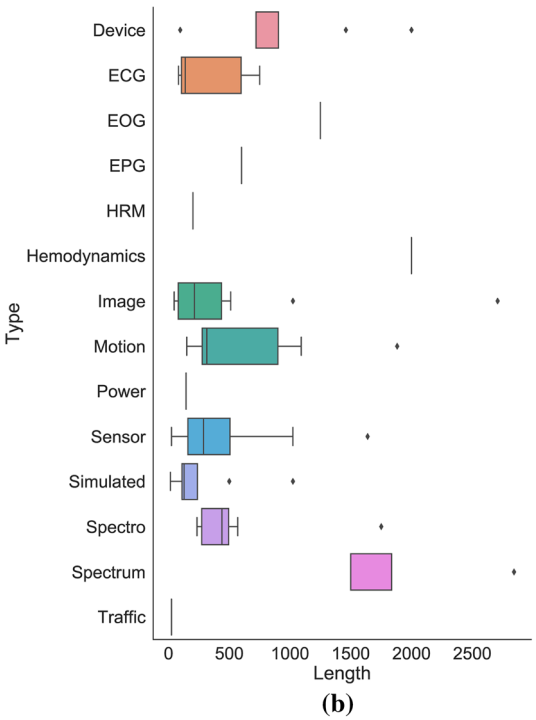

Fig. 2 The distributions of class numbers and time series lengths of datasets with different types. **a** Class numbers; b Time series lengths

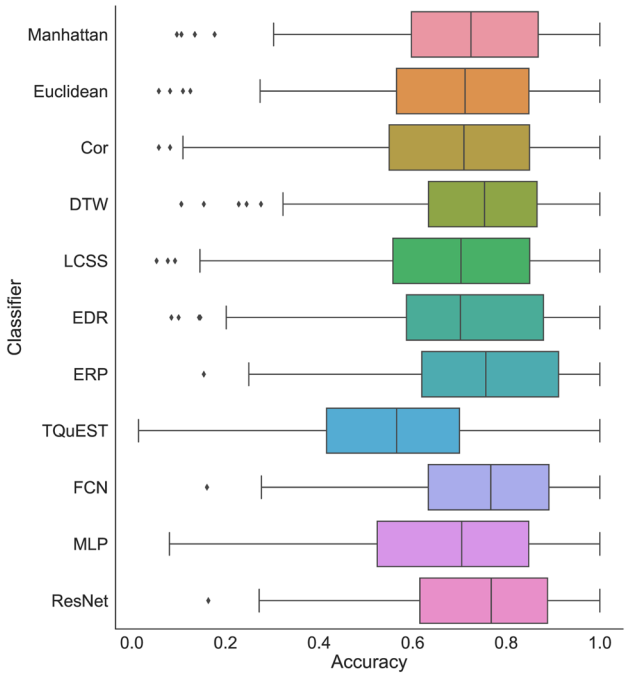

Fig. 3 The distribution of accuracy of different classifiers

6.4.4 Critical difference diagram

The critical difference diagram is shown in Fig. 4. As we can tell from Fig. 4, deep learning models fail to outperform 1-NN classifiers with significant differences, even though sophisticated models including FCN and ResNet perform better. MLP with a simple structure fails to beat the 1-NN classifiers. This indicates that 1-NN classifiers with ERP and DTW distances remain strong baselines for time series classification problems.

6.4.5 Pair-wise comparison

For better illustrations, we give the pairwise comparisons of accuracy between deep learning models versus 1-NN classifiers with different distance measures (Fig. 5 for FCN, Fig. 6 for MLP, and Fig. 7 for ResNet). The results further confirm that FCN and ResNet present a better performance, while MLP shows no obvious advantages than distance-based models. For further analysis, the readers may refer to the detailed results as we provide in the supplemental file.

Table 2 Performances of different methods across the 128 datasets split by problem types in Table 1 (bold indicates the best model)

Type	Manhattan	Euclidean	Cor	DTW	LCSS	EDR	ERP	TQuEST	FCN	MLP	ResNet
Device	0.0	0.0	0.0	11.1	11.1	0.0	0.0	0.0	**55.6**	0.0	22.2
ECG	16.7	0.0	0.0	0.0	16.7	0.0	0.0	0.0	0.0	**50.0**	16.7
EOG	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0	0.0	0.0
EPG	**16.7**	0.0	0.0	0.0	**16.7**	**16.7**	**16.7**	0.0	**16.7**	8.3	8.3
HRM	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0
Hemodynamics	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0	0.0	0.0	0.0	0.0
Image	0.0	2.9	2.9	8.6	2.9	**28.6**	5.7	0.0	11.4	17.1	20.0
Motion	9.1	0.0	0.0	9.1	4.5	9.1	18.2	0.0	9.1	9.1	**31.8**
Power	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	**100.0**	0.0
Sensor	13.9	2.8	5.6	13.9	2.8	5.6	11.1	0.0	11.1	5.6	**27.8**
Simulated	0.0	0.0	0.0	**33.3**	0.0	0.0	22.2	11.1	**33.3**	0.0	0.0
Spectro	0.0	10.0	10.0	10.0	0.0	0.0	0.0	0.0	10.0	**60.0**	0.0
Spectrum	0.0	20.0	20.0	0.0	0.0	0.0	**60.0**	0.0	0.0	0.0	0.0
Traffic	0.0	0.0	**50.0**	0.0	0.0	0.0	0.0	0.0	0.0	**50.0**	0.0
Trajectory	0.0	0.0	0.0	**33.3**	0.0	0.0	0.0	0.0	**33.3**	0.0	**33.3**
Overall	6.4	2.6	3.8	11.5	4.5	12.2	11.5	0.6	14.1	14.1	**18.6**

Each entry is the percentage of datasets a classifier is most accurate for

Table 3 Average ranks of different methods across the 128 datasets split by problem types in Table 1 (bold indicates the best model)

Type	Manhattan	Euclidean	Cor	DTW	LCSS	EDR	ERP	TQuEST	FCN	MLP	ResNet
Device	6.11	8.44	8.89	4.06	5.28	5.78	3.89	8.78	**2.33**	9.56	2.89
ECG	6.67	6.00	6.00	8.83	7.33	7.00	4.83	9.50	3.33	**3.17**	3.33
EOG	6.00	4.50	4.50	**1.00**	9.50	11.00	3.00	9.50	4.50	8.00	4.50
EPG	**3.50**	10.50	10.50	8.50	**3.50**	**3.50**	**3.50**	8.50	**3.50**	5.50	5.00
Hemodynamics	6.00	8.83	8.83	5.67	7.67	**1.00**	2.67	11.00	3.67	7.33	3.33
HRM	2.00	5.50	5.50	3.00	10.00	8.00	**1.00**	11.00	7.00	4.00	9.00
Image	6.17	6.48	6.48	5.56	5.88	**4.72**	4.75	9.61	5.28	6.31	4.75
Motion	5.68	7.65	7.71	5.59	6.47	4.68	3.53	9.47	3.35	9.24	**2.65**
Power	2.00	4.50	4.50	9.00	10.00	8.00	3.00	11.00	6.00	**1.00**	7.00
Sensor	5.20	5.93	6.28	5.03	6.47	5.67	5.42	9.72	4.73	7.17	**4.38**
Simulated	6.31	7.19	7.19	**3.44**	6.31	7.38	4.50	8.06	3.88	7.50	4.25
Spectro	5.44	4.44	4.44	6.50	8.69	7.44	6.44	7.38	6.38	**2.00**	6.88
Spectrum	4.00	6.25	6.25	4.00	7.75	8.25	**2.75**	11.00	6.50	**2.75**	6.50
Traffic	6.00	6.50	3.50	6.00	10.00	9.00	5.50	11.00	**2.50**	3.50	**2.50**
Trajectory	7.00	3.67	5.67	3.00	9.17	10.17	8.00	10.67	2.67	4.33	**1.67**
Overall	5.70	6.53	6.65	5.30	6.65	5.80	4.66	9.44	4.45	6.56	**4.25**

Table 4 Average ranks of different methods across the 128 datasets grouped by the training size (bold indicates the best model)

Train size	Manhattan	Euclidean	Cor	DTW	LCSS	EDR	ERP	TQuEST	FCN	MLP	ResNet
< 100	5.45	6.33	6.21	5.31	6.11	5.50	**4.75**	8.67	6.10	5.36	6.21
100–399	5.80	6.77	7.20	4.75	6.47	5.67	4.45	9.78	3.92	7.48	**3.71**
400–799	6.05	6.67	6.83	5.67	7.00	6.62	5.08	10.32	3.25	5.60	**2.90**
> 799	5.56	6.12	5.88	6.50	8.19	5.97	4.59	9.31	3.31	8.06	**2.50**

Table 5 Average ranks of different methods across the 128 datasets grouped by the datasets' length (bold indicates the best model)

Length	Manhattan	Euclidean	Cor	DTW	LCSS	EDR	ERP	TQuEST	FCN	MLP	ResNet
< 81	5.45	6.33	6.21	5.31	6.11	5.50	**4.75**	8.67	6.10	5.36	6.21
81–250	5.67	6.58	7.10	5.68	6.65	5.10	5.18	9.98	3.83	6.83	**3.38**
251–450	6.04	7.11	7.32	**3.66**	6.34	6.70	3.68	9.91	4.14	7.04	4.07
451–700	6.00	6.29	6.54	6.50	7.21	6.29	5.46	9.88	2.50	7.00	**2.33**
701–1000	5.75	6.75	6.88	6.00	7.62	3.81	4.31	8.50	4.50	8.50	**3.38**
> 1000	5.38	5.50	4.88	7.00	8.75	8.12	4.88	10.12	2.12	7.62	**1.62**

Table 6 Average ranks of different methods across the 128 datasets grouped by the class number (bold indicates the best model)

Class number	Manhattan	Euclidean	Cor	DTW	LCSS	EDR	ERP	TQuEST	FCN	MLP	ResNet
< 10	5.73	6.52	6.59	5.67	6.37	5.83	4.91	9.13	4.59	6.20	**4.47**
10–19	5.00	6.76	6.87	**3.42**	7.68	5.84	3.53	10.00	4.26	8.37	4.26
> 20	6.43	6.32	6.75	5.29	7.14	5.57	4.57	10.79	3.79	6.57	**2.79**

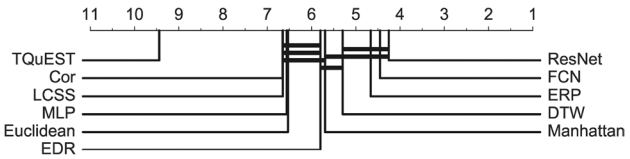

Fig. 4 The critical difference diagram

7 Conclusion

In this study, we give an exploration for the question that whether deep learning models are better than distance-based nearest neighbor models or not for time series classification. We compare 1-NN classifiers with eight different distance measures and three state-of-the-art deep learning models on 128 time series datasets. The results indicate that even though ResNet and FCN show a better performance for most of the datasets, they are not significantly different from 1-NN classifiers with some traditional distance measures (e.g., edit distance with real penalty and dynamic time warping) in terms of accuracy.

The findings can be interpreted from two aspects. From the positive aspect, sophistic deep learning models do perform better for most of the datasets, which give them a higher probability of achieving a good result on a new time series classification problem. From the negative aspect, the success of deep learning techniques in image processing, video processing and natural language processing where they are dominant is not so easy to be replicated in time series classification. Old-fashion approach of using 1-NN classifiers with distances including edit distance with real penalty and dynamic time warping is still very competitive nowadays.

For further evaluation, more time series data are needed, especially those from real problems. Multivariate time series may present different results from univariate time series. Compared with the diverse deep learning models used for image classification, we only evaluate three models in this project, which leaves a huge space of exploring neural networks with different structures.

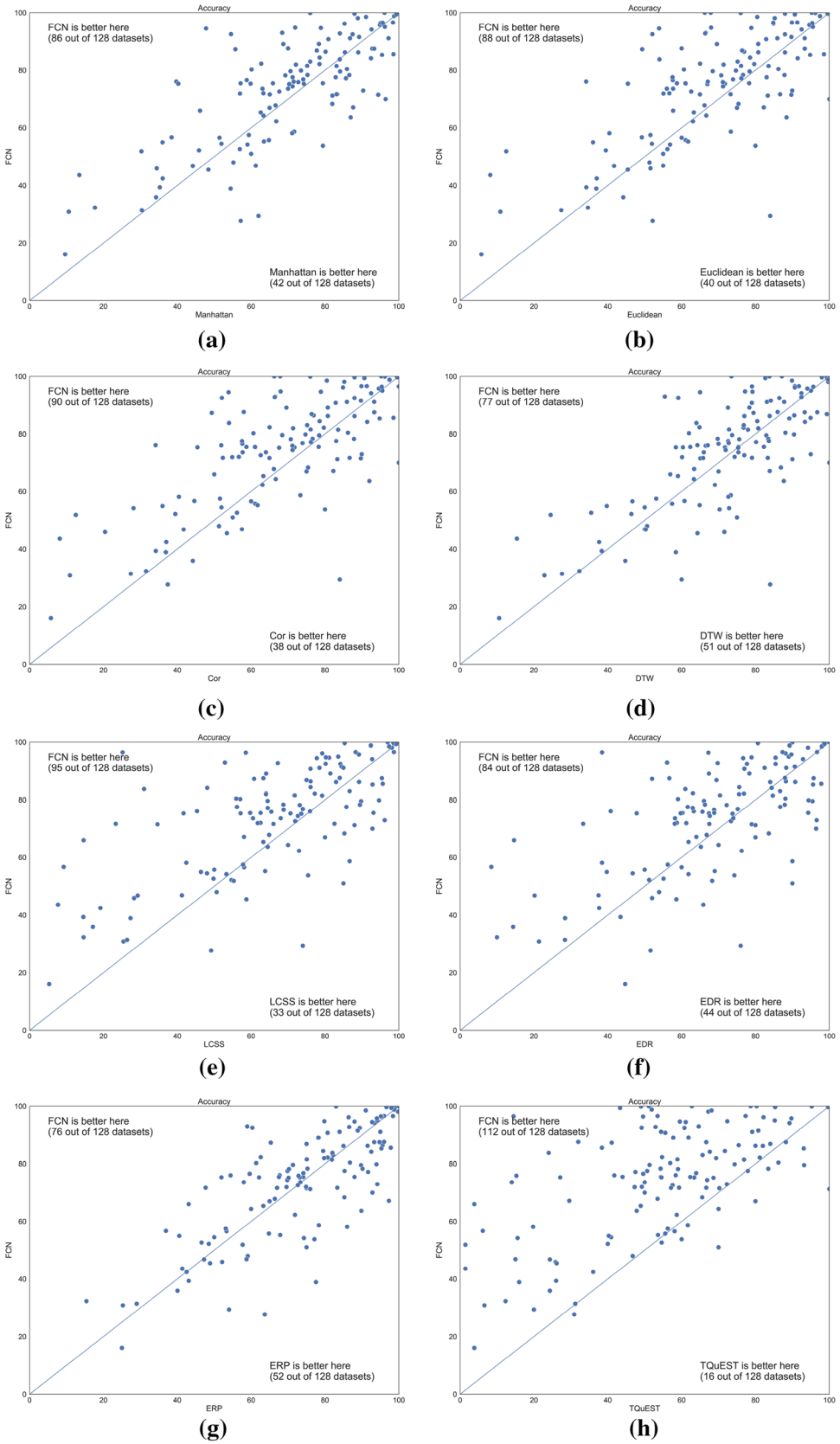

Fig. 5 Accuracy of FCN versus 1-NN classifiers with different distance measures. **a** FCN versus Manhattan; **b** FCN versus Euclidean; **c** FCN versus Cor; **d** FCN versus DTW; **e** FCN versus LCSS; **f** FCN versus EDR; **g** FCN versus ERP; **h** FCN versus TQuEST

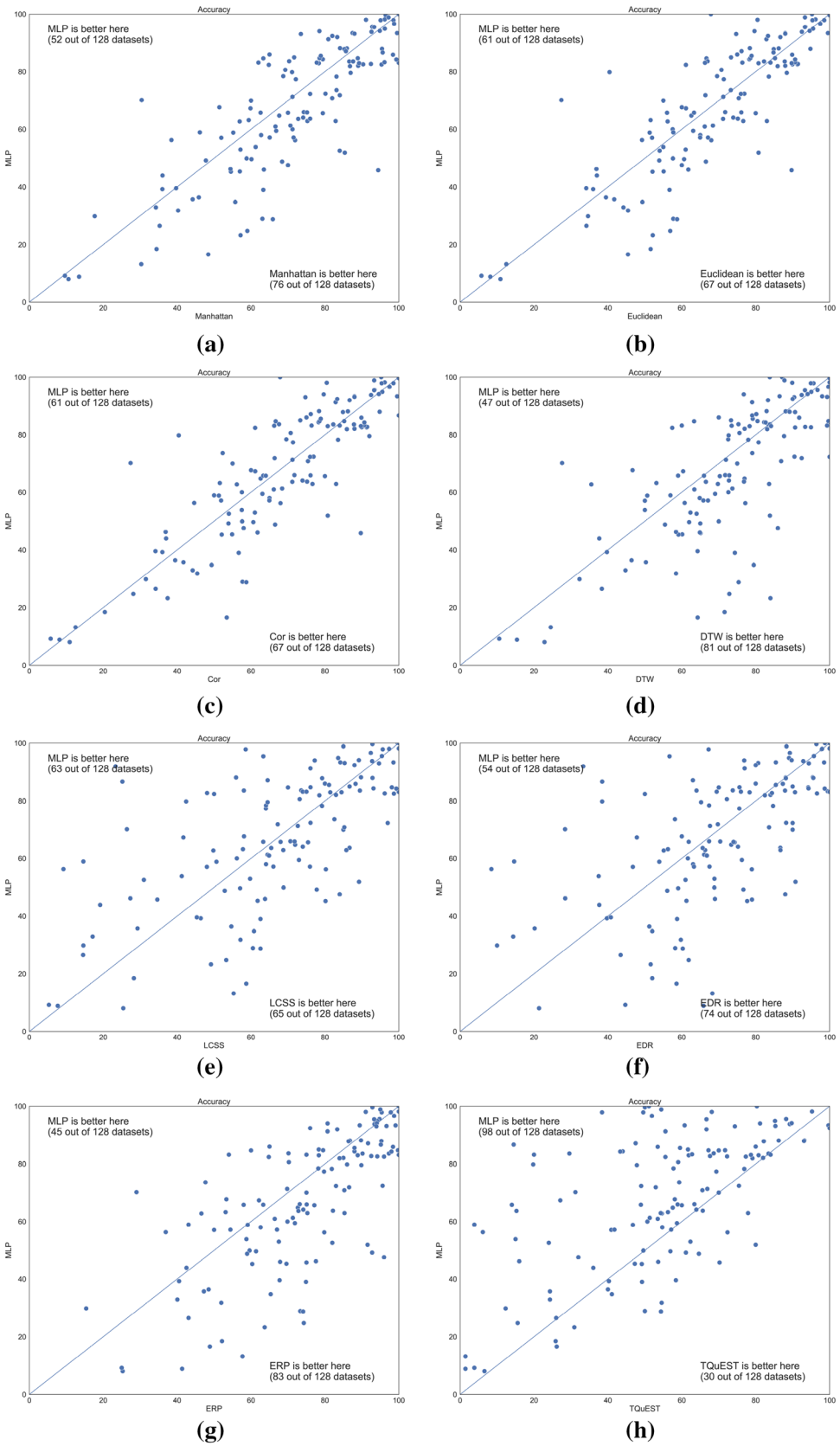

Fig. 6 Accuracy of MLP versus 1-NN classifiers with different distance measures. **a** MLP versus Manhattan; **b** MLP versus Euclidean; **c** MLP versus Cor; **d** MLP versus DTW; **e** MLP versus LCSS; **f** MLP versus EDR; **g** MLP versus ERP; **h** MLP versus TQuEST

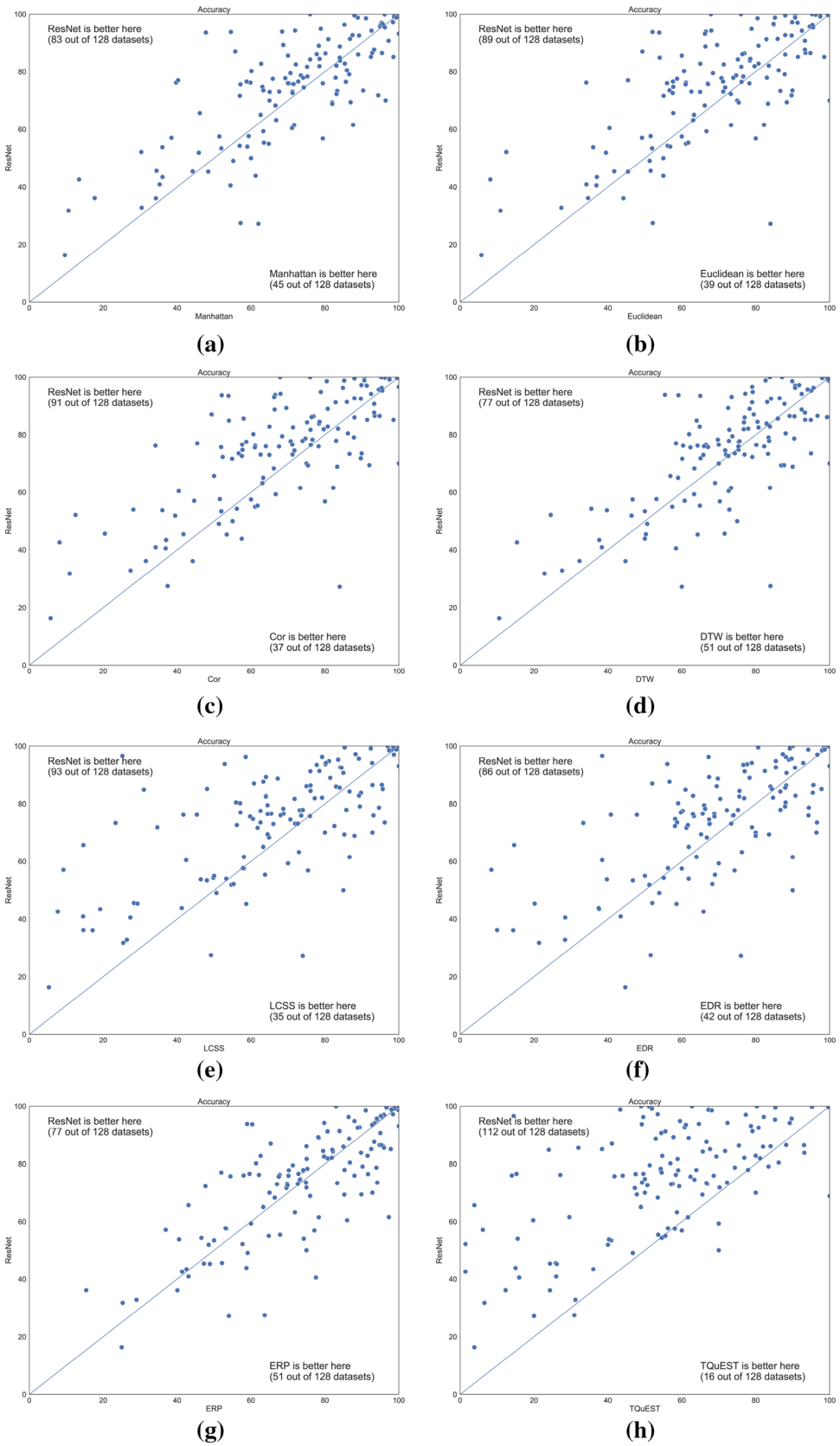

Fig. 7 Accuracy of ResNet versus 1-NN classifiers with different distance measures. **a** ResNet versus Manhattan; **b** ResNet versus Euclidean; **c** ResNet versus Cor; **d** ResNet versus DTW; **e** ResNet versus LCSS; **f** ResNet versus EDR; **g** ResNet versus ERP; **h** ResNet versus TQuEST

Compliance with ethical standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

References

1. Abanda A, Mori U, Lozano JA (2019) A review on distance based time series classification. Data Min Knowl Discov 33(2):378–412
2. Aßfalg J, Kriegel HP, Kröger P, Kunath P, Pryakhin A, Renz M (2006) Similarity search on time series based on threshold queries. In: International conference on extending database technology. Springer, pp 276–294
3. Bagnall A, Lines J (2014) An experimental evaluation of nearest neighbour time series classification. arXiv preprint arXiv:14064757
4. 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
5. Baydogan MG, Runger G, Tuv E (2013) A bag-of-features framework to classify time series. IEEE Trans Pattern Analy Mach Intell 35(11):2796–2802
6. Berndt DJ, Clifford J (1994) Using dynamic time warping to find patterns in time series. KDD Workshop Seattle 10:359–370
7. Che Z, He X, Xu K, Liu Y (2017) Decade: a deep metric learning model for multivariate time series. In: KDD workshop on mining and learning from time series
8. Chen L, Ng R (2004) On the marriage of lp-norms and edit distance. In: Proceedings of the thirtieth international conference on very large data bases, Vol 30. VLDB Endowment, pp 792–803
9. Chen L, Özsu MT, Oria V (2005) Robust and fast similarity search for moving object trajectories. In: Proceedings of the 2005 ACM SIGMOD international conference on management of data. ACM, pp 491–502
10. Chen Y, Nascimento MA, Ooi BC, Tung AK (2007) Spade: on shape-based pattern detection in streaming time series. In: 2007 IEEE 23rd international conference on data engineering. IEEE, pp 786–795
11. Cui Y, Shi J, Wang Z (2015) Complex rotation quantum dynamic neural networks (crqdnn) using complex quantum neuron (cqn): applications to time series prediction. Neural Netw 71:11–26
12. Cui Z, Chen W, Chen Y (2016) Multi-scale convolutional neural networks for time series classification. arXiv preprint arXiv:16030 6995
13. Dau HA, Keogh E, Kamgar K, Yeh CCM, Zhu Y, Gharghabi S, Ratanamahatana CA, Yanping, Hu B, Begum N, Bagnall A, Mueen A, Batista G, Hexagon-ML (2018) The UCR time series classification archive. https://www.cs.ucr.edu/~eamonn/time_series_data_2018/
14. Demšar J (2006) Statistical comparisons of classifiers over multiple data sets. J Mach Learn Res 7(Jan):1–30
15. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L (2009) Imagenet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern recognition. IEEE, pp 248–255
16. Faloutsos C, Ranganathan M, Manolopoulos Y (1994) Fast subsequence matching in time-series databases, vol 23. ACM, New York
17. Fawaz HI, Forestier G, Weber J, Idoumghar L, Muller PA (2019a) Deep learning for time series classification: a review. Data Min Knowl Discov 33(4):917–963
18. Fawaz HI, Lucas B, Forestier G, Pelletier C, Schmidt DF, Weber J, Webb GI, Idoumghar L, Muller PA, Petitjean F (2019b) Inceptiontime: finding alexnet for time series classification. arXiv preprint arXiv:190904939
19. Frentzos E, Gratsias K, Theodoridis Y (2007) Index-based most similar trajectory search. In: 2007 IEEE 23rd international conference on data engineering. IEEE, pp 816–825
20. 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(Dec):2677–2694
21. Grabocka J, Schmidt-Thieme L (2018) Neuralwarp: time-series similarity with warping networks. arXiv preprint arXiv:18120 8306
22. Hamilton JD (1994) Time series analysis, vol 2. Princeton University Press, Princeton
23. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 770–778
24. Jeong YS, Jeong MK, Omitaomu OA (2011) Weighted dynamic time warping for time series classification. Pattern Recognit 44(9):2231–2240
25. Jiang W, Zhang L (2018) Geospatial data to images: a deep-learning framework for traffic forecasting. Tsinghua Sci Technol 24(1):52–64
26. Jiang W, Zhang L (2020) Edge-siamnet and edge-triplenet: new deep learning mdels for handwritten numeral recognition. IEICE Trans Inf Syst 103:720
27. Jiang W, Lian J, Shen M, Zhang L (2017) A multi-period analysis of taxi drivers' behaviors based on GPS trajectories. In: 2017 IEEE 20th international conference on intelligent transportation systems (ITSC). IEEE, pp 1–6
28. Kadous MW et al (2002) Temporal classification: extending the classification paradigm to multivariate time series. University of New South Wales, Kensington
29. Karim F, Majumdar S, Darabi H, Chen S (2017) Lstm fully convolutional networks for time series classification. IEEE Access 6:1662–1669
30. Keogh E, Ratanamahatana CA (2005) Exact indexing of dynamic time warping. Knowl Inf Syst 7(3):358–386
31. Kingma DP, Ba J (2014) Adam: a method for stochastic optimization. arXiv preprint arXiv:14126980
32. Lin J, Keogh E, Wei L, Lonardi S (2007) Experiencing sax: a novel symbolic representation of time series. Data Min Knowl Discov 15(2):107–144
33. Lines J, Bagnall A (2015) Time series classification with ensembles of elastic distance measures. Data Min Knowl Discov 29(3):565–592
34. Ma Q, Zhuang W, Shen L, Cottrell GW (2019) Time series classification with echo memory networks. Neural Netw 117:225
35. Mori U, Mendiburu A, Lozano JA (2016) Distance measures for time series in r: the tsdist package. R J 8(2):451–459
36. Morse MD, Patel JM (2007) An efficient and accurate method for evaluating time series similarity. In: Proceedings of the 2007 ACM SIGMOD international conference on Management of data. ACM, pp 569–580
37. Prieto OJ, Alonso-González CJ, Rodríguez JJ (2015) Stacking for multivariate time series classification. Pattern Anal Appl 18(2):297–312
38. Redmon J, Farhadi A (2018) Yolov3: an incremental improvement. arXiv preprint arXiv:180402767
39. Schäfer P (2015) The boss is concerned with time series classification in the presence of noise. Data Min Knowl Discov 29(6):1505–1530
40. Sharabiani A, Darabi H, Rezaei A, Harford S, Johnson H, Karim F (2017) Efficient classification of long time series by 3-d dynamic time warping. IEEE Trans Syst Man Cybern Syst 47(10):2688–2703

41. Vlachos M, Kollios G, Gunopulos D (2002) Discovering similar multidimensional trajectories. In: Proceedings 18th international conference on data engineering. IEEE, pp 673–684
42. Wang X, Mueen A, Ding H, Trajcevski G, Scheuermann P, Keogh E (2013) Experimental comparison of representation methods and distance measures for time series data. Data Min Knowl Discov 26(2):275–309
43. Wang Z, Yan W, Oates T (2017) Time series classification from scratch with deep neural networks: a strong baseline. In: 2017 international joint conference on neural networks (IJCNN). IEEE, pp 1578–1585
44. Yi BK, Faloutsos C (2000) Fast time sequence indexing for arbitrary lp norms. VLDB 385:99
45. Zheng Y, Liu Q, Chen E, Ge Y, Zhao JL (2016) Exploiting multi-channels deep convolutional neural networks for multivariate time series classification. Front Comput Sci 10(1):96–112