2016 15th IEEE International Conference on Machine Learning and Applications

Time Series Classification Using Time Warping Invariant Echo State Networks

Pattreeya Tanisaro and Gunther Heidemann
Institute of Cognitive Science, University of Osnabrück, Germany
Email: {pattanisaro}, {gheideman} @uni-osnabrueck.de

***Abstract*—For many years, neural networks have gained gigantic interest and their popularity is likely to continue because of the success stories of deep learning. Nonetheless, their applications are mostly limited to static and not temporal patterns. In this paper, we apply time warping invariant Echo State Networks (ESNs) to time-series classification tasks using datasets from various studies in the UCR archive. We also investigate the influence of ESN architecture and spectral radius of the network in view of general characteristics of data, such as dataset type, number of classes, and amount of training data. We evaluate our results comparing it to other state-of-the-art methods, using One Nearest Neighbor (1-NN) with Euclidean Distance (ED), Dynamic Time Warping (DTW) and best warping window DTW.**

I. Introduction

Time series classification is a measure to assess the similarity of signals in the time domain. Even static object shapes can be transformed into time series [1]–[3], to which afterwards a time series classification method can be applied. Numerous measures have been introduced to solve time-series classification problems and were benchmarked using the real world problem data collected at the UCR archive [4]. The UCR archive is the largest collection of time-series datasets available from a variety of application domains such as medical, biological, physics, food science, sport and sensor information. At the time of writing, the archive consists of 85 datasets from various scientific disciplines of which the most recent 13 datasets have only been tested in [5].

Two important ideas for successful time series classification are (i) transforming the time series representations in order to reduce dimensionality and to speed up the classification [6] and (ii) extracting distinguishing features in a new data space [7]–[11]. Among the time series classification approaches, new data representations using distance functions in combination with similarity search using k-nearest neighbor classification (k-NN) have shown their distinguishing effectiveness [7]. Two common distance measures are Euclidean Distance (ED) and Dynamic Time Warping (DTW) [12]–[15]. They are the most-used techniques for benchmarking time series classification. Several studies (i.e. [5], [7], [16]) reveal that there is no single approach that wins on all datasets. Some similarity measures are superior to others on certain datasets, and inferior on different ones. The main reason for this is: in each application domain some significant features may not be interpretable in one approach, although changing their representation can reveal them. Hence, there is an effort to use an ensemble approach to account for the hypothesis that a combination of different classifiers is substantially better than a single classifier e.g. ensembles of elastic distance measures [17] and the collective of transformation-based ensembles (COTE) [5], [8].

In this paper, we will demonstrate the capability of a time invariant warping Echo State Network (ESN) to solve time series classification problems. ESN [18], [19] is a type of Recurrent Neural Networks (RNNs). Its weights are randomly initialized and left unchanged over the whole training procedure. The inputs are mapped to dynamic patterns of the reservoir neurons which exhibit such complexity and high dimensionality that different patterns become linearly separable. Time invariant warping Echo State Network [20] is an ESN with leaky integrator neurons that is inherently invariant under time warping of the series of data. Therefore, an ESN can be applied to a variety of tasks that deal with temporal patterns.

This paper is structured as follows. In section II, we discuss various interesting approaches for time series classification, especially popular similarity measures using distance functions. In section III, we explain ESNs and time warping invariant ESNs for classification tasks. In section IV, we discuss the datasets, ESN configurations and the comparisons of different ESN models versus 1-NN with ED, DTW, and best Warping Window DTW (referred as DTW-R in this paper, where R is the percentage of time series length described in [4]). Finally, we summarize results, highlight our findings and discuss the advantages of using an ESN in section V.

II. Background and related work

Most work on time series classification focuses on adaptation of distance measures for the 1-NN classifier, especially for ED and DTW. ED is perhaps the simplest and most straightforward. It is a one-to-one mapping of data points from two sequences. Therefore, ED is very fragile if two sequences are out of phase. DTW was first proposed in [21] to generate a warping path of two sequences such that the distance between the two is minimized. It allows for mapping one point of one sequence to many points of another sequence and is not restricted to phase shifting. Nevertheless, the warping path still has a few restrictions and more important, the boundary conditions that limit the search space, implying the shortcoming when searching in a large database. Keogh et al [15] introduce a lower bounding measure for indexing DTW. This concept leads to the speed up of search in k-NN. For this reason, DTW has become widely used in solving time-series

978-1-5090-6167-9/16 $31.00 © 2016 IEEE
DOI 10.1109/ICMLA.2016.166

classification problems.

As discussed in the previous section, generally work on time series classification involves two aspects (i) data representation methods to reduce dimensionality or to transform data into another feature space, and (ii) time series distance measures. Comparative studies of these two aspects appeared in [7] gauging eight representation methods and nine different well-known distance measures. Another comparative study of similarity measures for time series published a year later is [16]. It compares seven techniques such as ED, Fourier coefficients, DTW, Auto Regressive (AR) models, DTW, Edit distance on real sequences (EDR), time-warped edit distance (TWED) which is an extension of DTW and minimum jump costs dissimilarity. Another interesting work on time series classification is to use complexity-invariance distance (CID) measure [22]. They assume that in many domains, different classes have different complexities and even pairs of complex objects tend to be further apart under certain distance measures. This induces the incorrect assignment of a complex class to a simpler class in simple nearest neighbor classification. By estimating complexity using the CID approach, it produces significant improvements in classification accuracy in many cases. It also introduces a technique, called Texas Sharpshooter, to predict whether the method has accuracy superior to that of other methods ahead of time by computing the accuracy gain from training data. A new representation domain using recurrence plots called RPCD is proposed in [9]. The output outperforms ED in most cases. Another time series transformation is the improved SVM as presented by [23]. The transformed instances capture the necessary intra-class variations to redefine the decision margin using variance distributions from intra-class warping. However, the experiment displays only results obtained from small datasets while large training datasets are left out. Another technique to measure the similarity between time series is shapelet-based [24]. Shapelets are time series primitives that are used to find common shapes in a series. The benefits are: it can provide interpretable result and it is more robust on some datasets with dominant local features. The works of time series classification that are shapelet based can be found in [3], [25]. A very interesting approach favoring the interpretability of feature selection is demonstrated in [11]. It uses numerous algorithms to extract thousands of features from time series, then afterwards reduces those feature vectors using a process called greedy forward feature selection. These feature representations are acquired from a wide range of data properties, for example, statistics distributions, correlation structure, entropy, linear and non linear model fits (e.g., Gaussian Process, goodness of fit estimates), nonlinear time series analysis and others. Of all the studies that have been discussed, perhaps the best solution to solve time series classification problems by far is to use a collective of ensembles of classifiers on different data transformations. The Collective of Transformation-Based Ensembles (COTE) [8] is such an approach. It includes 35 classifiers in one ensemble as well as transforming data (including the shapelet) into an alternative data space where the distinctive features are easily detected. The very recent experimental evaluation [5] compares 18 approaches based on the UCR archive. It shows that COTE is most significantly better than DTW and has high accuracy. Nonetheless, it is very complicated, time demanding and computationally expensive. On the contrary, ESN is easy and computational cheap. Generalized ESNs show good performance and are comparable to those of three start-of-the-art approaches and the network can be further optimized. Hence, ESN can be considered as an alternative method to solve time series classification for many problems.

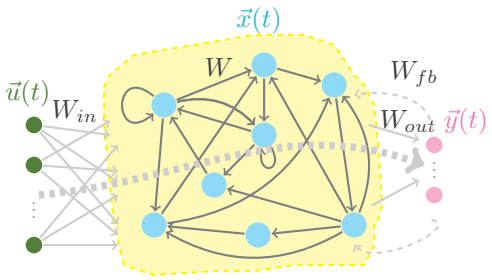

Fig. 1. Architecture of an ESN. The dashed lines denote the connections which are not compulsory.

III. Classification using an ESN

Original ESN was proposed by Jaeger in [18], [19]. It is a type of RNNs of which the weights are left untrained. Only the output weights are to be trained for the desired target at the read out connection where no cyclic dependencies are created. In general, the use of RNNs for practical applications is limited, since a simple training method like gradient descent is usually not feasible. Convergence is slow and not always guaranteed [19]. Moreover, cyclic dependencies in the network can lead to bifurcations when changing a parameter during training [26], and parameter updating is computationally expensive. Yet, a RNN has a certain biological plausibility and is applied to model complex dynamical systems, for instance, the work of [27] presents a framework for neurodynamical models of working memory using an ESN. The framework illustrates ESN mechanism properties for storing, maintaining, retrieving and removing data similar to the function of the brain. The core of an ESN is a sparsely connected random RNN called *reservoir*. When driven by input signals, each neuron of the reservoir creates its own nonlinear transformation of the incoming signal. There is no adaptation of the inter-connection weights in the reservoir, only the readout weights of the ESN are adapted to a classification task. A general ESN architecture is shown in Figure 1.

Here, we use a modification of the original ESN, called time warping invariant echo state network [20]. It applies the idea of time warping invariant neural network (TWINN) [28] showing that time warping using a fully connected recurrent neural network can have number of operation $O(N)$ where

N is the length of input. Time warping in neuron networks can be considered as *a variation of the speed of the process*. Furthermore, the idea of a *leaky integrator neuron* in ESN is chosen from a biologically inspired model in which the neuron will both accumulate inputs and in the mean time also leak the accumulated excitation. Therefore, it can handle time warped signals for pattern recognition.

Consider a continuous time neuron network with input dimensionality N_u, neurons in the reservoir N_x, and output dimensionality N_y. Let $\vec{u}(t) \in \mathbf{R}^{N_u}$, $\vec{x}(t) \in \mathbf{R}^{N_x}$ and $\vec{y}(t) \in \mathbf{R}^{N_y}$ denote the vectors of input activities, internal state and output unit activity for time t respectively. Further, let $W_{in} \in \mathbf{R}^{N_x \times N_u}$, $W \in \mathbf{R}^{N_x \times N_x}$ and $W_{out} \in \mathbf{R}^{N_y \times N_x}$ denote the weight matrices for input connections, internal connections, and output connections as seen in Figure 1. In addition, the output might be back-coupled to the reservoir via weights $W_{fb} \in \mathbf{R}^{N_x \times N_y}$. The internal unit activities $\vec{x}$ in Figure 1 are updated from time step $t-1$ to time t, $t = 1, ..., T$, by

$$\vec{x}(t) = f(W_{in}\vec{u}(t) + W\vec{x}(t-1) + W_{fb}\vec{y}(t)) \tag{1}$$

$f(\cdot)$ is an activation function of the neurons, a common choice is $tanh(\cdot)$ applied element-wise. A *leaky integration rate* $\alpha \in (0, 1]$ is the leakage rate determining the speed of the reservoir update dynamics [29], [20]. Then the update rule for the internal units is extended to

$$\vec{x}_{leaky}(t) = (1-\alpha)\vec{x}(t-1) + \alpha\vec{x}(t). \tag{2}$$

If there are also direct connections from the input $\vec{u}(t)$ to output layer, the output can be computed according to

$$\vec{y}(t) = f_{out}\left(W_{out}[\vec{u}(t); \vec{x}(t)]\right), \tag{3}$$

where $[\cdot;\cdot]$ is a matrix concatenation and f_{out} is a non-linear function. Accordingly, W_{out} now becomes $W_{out} \in \mathbf{R}^{N_y \times (N_x + N_u)}$. Typically, a simple linear regression is applied at the readout layer. Hence, equation 3 can be simplified to

$$\vec{y}(t) = W_{out}[\vec{u}(t); \vec{x}(t)]. \tag{4}$$

The class for testing input sequence $\vec{u}(t)$ is then computed by

$$\text{class}(\vec{u}(t)) = \underset{k}{\text{argmax}} \left\{ \frac{1}{|\tau|} \sum_{t \in \tau} \vec{y}_k(t) \right\} \tag{5}$$

where $y_k(t)$ is the corresponding output of class k, and τ is the length of input $u(t)$.

IV. Experimental Setup and Results

A. Datasets in the UCR archive

We benchmark 85 datasets from the UCR archive [4]. Each dataset contains a separate test and training set. All data is in one dimension and already normalized to have standard deviation to one and zero-mean. These data sometimes are grouped by data types or characteristics of the data. In Table II, each small colored square shown in front of the dataset name illustrates a different data type. Image outline data (■) from different disciplines are i.e., *Fish*, *OSULeaft*, *DistPhalanxAge*, *DistPhalanxOutline*, *DistPhalanxTW*, *WordSynonyms*, *Yoga*, *BirdChicken*, *Herring*, *FaceAll*, *50Words*, and *Symbols*. Motion datasets (■) are for instance, *Cricket_X*, *Cricket_Y*, *Cricket_Z*, *InlineSkate*, *Haptics*, *GunPoint*, and *MutantWorms*. The other data types are e.g., from sensor reading (■), device (■), ECG (■) and simulated data (■). Figure 2 shows the plot of training dataset *BeetleFly* obtained from shapelet transformation [25] to classify *Beetle* and *Fly* from their contours. Figure 3 shows the plot of 30 samples of extra virgin olive oils in the *OliveOil* from four different producing countries [30]. The spectra were collected using Fourier Transform infrared spectroscopy (FTIR) spectrometer, then the classification was conducted using FTIR in combination with several multivariate approaches. The classification of input signals in this case is hard to solve for most classifiers, despite the original paper reporting 96% recognition rate using simple partial least squares (PLS) and linear discriminant analysis (LDA). This is because the original data is modified to reduce multidimensional data to one dimension.

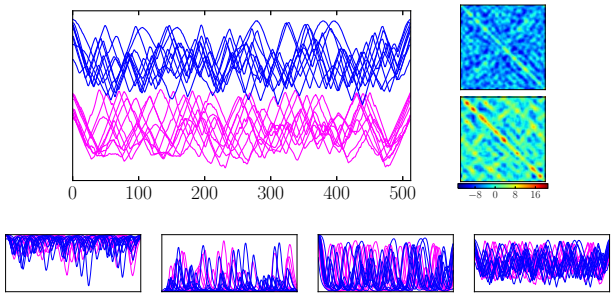

Fig. 2. The *BeetleFly* training set, which was obtained from shapelet transformations, consists of 10 samples of *Beetle* and *Fly* each. (Top left) The values on the x-axis show time, the values on the y-axis are shifted so that we are able to see the characteristic of each data class by different color. The corresponding covariance matrix of each class representation is shown on the top right. (Bottom) States (X) from four nodes in ESNs of $N_x = 250$.

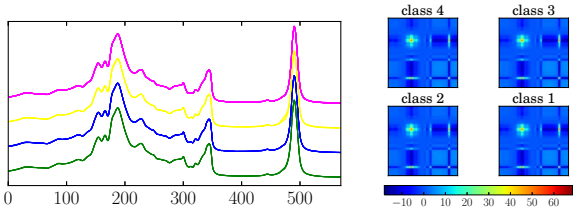

Fig. 3. (Left) Plot of the *OliveOil* training datasets of 30 samples from 4 different countries. (Right) The corresponding covariance matrices show no significant difference from 4 classes and this task is more difficult to solve than *BeetleFly*.

B. ESN Configurations

In order to get optimal recognition rates in ESN, there are several important parameters that should be taken into account. In our experiment, we set up three random ESN networks and adjust spectral radii and input weight scaling to create new ESN models to apply to all datasets. These

N_x/Connect/ScaleW_{in}/λ	**250/0.5/1.0/0.0**			**250/0.5/2.0/0.05**				**500/0.1/2.0/0.05**				**800/0.1/2.0/0.05**			
$\rho(W)$	0.55	0.9	2.0	0.55	0.9	2.0	5.0	0.55	0.9	2.0	5.0	0.55	0.9	2.0	5.0
BeetleFly	0.300	0.250	0.150	0.150	**0.100**	**0.100**	**0.100**	0.150	**0.100**	0.150	0.350	0.150	**0.100**	0.200	0.25
DiatomSizeR	0.078	0.078	0.082	**0.284**	0.275	0.219	**0.065**	**0.284**	0.268	0.206	**0.062**	**0.271**	0.255	0.206	**0.098**
GunPoint	0.033	0.020	0.033	0.100	0.033	**0.020**	0.033	0.053	0.027	**0.007**	0.033	0.047	**0.020**	0.027	**0.020**
OliveOil	0.433	0.233	0.167	**0.600**	**0.600**	**0.600**	**0.233**	**0.600**	**0.600**	0.567	**0.100**	**0.600**	**0.600**	0.567	**0.100**

TABLE I
ERROR RATES FROM FIFTEEN ESN CONFIGURATIONS IN THREE ESN ARCHITECTURES. HIGHLIGHTED COLUMNS IN GRAY ARE CHOSEN FOR COMPARISON IN TABLE II.

important ESN parameters are: **(i) The reservoir size** N_x. The maximum memory capacity of reservoir is bounded by the network size. In addition, with the idea that the bigger the network, the easier it is to find a linear combination of the signal. Nonetheless, there is a trade-off between the size of the network and computational effort. To have a compromise between performance and computational complexity as well as to avoid overfitting, we have utilized the knowledge of a number of training datasets to setup the ESN. In terms of applying one network to various datasets such as in the UCR archive, we choose a moderate network size $N_x = 500$ as a baseline, and adapting the network size to a smaller and larger value with $N_x = 250$ and 800, respectively. **(ii) Sparsity of the reservoir**. ESNs are set in a way that network should be large enough to maintain memory capacity, but have sparse connections. In our setups, we use $N_x = 500$ and 800 with 10% connectivity and $N_x = 250$ with 50%. **(iii) Spectral radius** $\rho(W)$: is computed from the maximum of absolute eigenvalue of weight matrix W. It is considered to be a scaling factor of W. In theory, we usually set $\rho(W) < 1$ to ensure the echo state property, however in practice we select $\rho(W)$ in a way that maximizes the performance where 1 serves as a reference point. Therefore, the spectral radius should be set bigger for tasks that require an extensive history of input and smaller for tasks where the output depends on the recent history of the input [26]. The errors from empirical studies using four spectral radii, 0.55, 0.9, 2.0 and 5.0 for selected datasets are shown in Table I. The digits in the table fluctuate due to the influence of changing spectral radii in different network architectures. In the table, *OliveOil* and *DiatomSizeR* have gained much better performances when increasing the spectral radius to a certain amount (the worst highlighted in red and the best in blue), whereas *BeetleFly* prefers a smaller spectral radius on a small network. **(iv) Leaky rate** (α): can be regarded as a time warping of the input signal or the speed of the dynamics of input and output. To simplify our experiment, we assume that there is little dynamic happening in the reservoir, therefore a small leaky rate should be sufficient to get good performances on most datasets. A rate of $\alpha = 0.1$ is used for all configurations. **(v) Input scaling**: should be small for the linear network (or the data that require short term memory) in order to drive the network around the resting state. We presume that most of the datasets in the UCR archive are nonlinear, therefore input scaling to W_{in} is set to 2.0 as a baseline for our models. Yet, to demonstrate the applicability of input scaling to capture the linear system, we insert input scaling 1.0 to $N_x = 250$. Apart from these important parameters, we also apply **(vi) ridge regression** by adding a small regularization coefficient $\lambda = 0.05$ at the readout to ensure a stable W_{out}. Furthermore, the networks all have uniformly distributed weights in the range of $[-0.5, 0.5]$ and we neglect feedback connection. A good general guideline about the setup of ESN configurations can be found in [26].

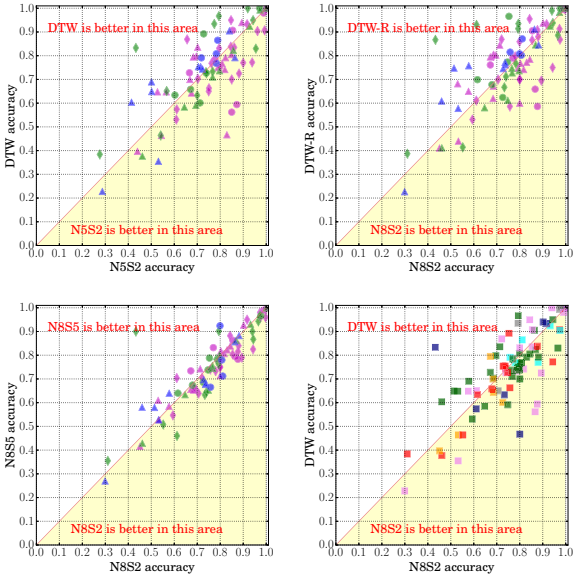

Fig. 4. Pairwise accuracy plots between two classifiers taken from the error rates in Table II. Each point represents a dataset. The colors in the top-left, top-right and bottom-left indicate the numbers of classes (nC): *magenta* for number of classes less than or equal 3, *green* for number of classes in [4..10] and *blue* for number of classes greater than 10. Different shapes indicate a high amount of training data: *diamond*(♦) for training data less than or equal 100, *triangle*(▲) for training data in (100..500], and *circle*(●) for training data greater than 500. (Bottom-right): The pairwise comparison between N8S2 and DTW is shown using type of dataset indicated in different colors and square markers (■) to signify the data type

C. Results and Discussion

We built fifteen models from three different ESN architectures, $N_x = 250, 500$ and 800 by varying spectral radii and input weight scaling for $N_x = 250$ as seen in Table I. The other parameters are fixed as constants. The best ESN model from these fifteen configurations from averaging over

Dataset	nC	nTrain	ED	DTWR	DTW	N5S2	N8S2	N8S5	optESN
50Words	50	450	0.369	0.242	0.310	0.499	0.422	0.360	0.360
Adiac	37	390	0.389	0.391	0.396	0.586	0.540	0.419	0.419
ArrowHead	3	36	0.200	0.200	0.297	0.297	0.314	0.337	0.274
Beef	5	30	0.333	0.333	0.367	0.300	0.267	0.367	0.267
BeetleFly	2	20	0.250	0.300	0.300	0.150	0.200	0.250	0.100
BirdChicken	2	20	0.450	0.300	0.250	0.200	0.200	0.250	0.150
CBF	3	30	0.148	0.004	0.003	0.003	0.003	0.040	0.002
Car	4	60	0.267	0.233	0.267	0.233	0.217	0.217	0.200
ChlorineCct	3	467	0.350	0.350	0.352	0.435	0.423	0.415	0.415
CinCECGts	4	40	0.103	0.070	0.349	0.433	0.388	0.540	0.388
Coffee	2	28	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Computers	2	250	0.424	0.380	0.300	0.332	0.316	0.344	0.304
Cricket_X	12	390	0.423	0.228	0.246	0.295	0.264	0.323	0.264
Cricket_Y	12	390	0.433	0.238	0.256	0.277	0.262	0.318	0.262
Cricket_Z	12	390	0.413	0.254	0.246	0.285	0.272	0.303	0.272
DiatomSizeR	4	16	0.065	0.065	0.033	0.206	0.206	0.098	0.062
DistalPhlxOAG	3	139	0.218	0.228	0.208	0.177	0.170	0.177	0.163
DistalPhlxOC	2	276	0.248	0.232	0.232	0.192	0.197	0.182	0.180
DistalPhlxTW	6	139	0.273	0.272	0.290	0.233	0.233	0.255	0.223
ECG	2	100	0.120	0.120	0.230	0.100	0.120	0.160	0.100
ECG5000	5	500	0.075	0.075	0.076	0.072	0.068	0.069	0.068
ECGFiveDays	2	23	0.203	0.203	0.232	0.238	0.220	0.267	0.144
Earthquakes	2	139	0.326	0.258	0.258	0.180	0.183	0.193	0.174
ElectricDevic	7	8926	0.450	0.376	0.399	0.288	0.274	0.314	0.274
FISH	7	175	0.217	0.154	0.177	0.143	0.131	0.137	0.086
Face_all	14	560	0.286	0.192	0.192	0.217	0.201	0.076	0.076
Face_four	4	24	0.216	0.114	0.170	0.045	0.034	0.057	0.034
FacesUCR	14	200	0.231	0.088	0.095	0.160	0.136	0.143	0.136
FordA	2	1320	0.341	0.341	0.438	0.150	0.133	0.206	0.133
FordB	2	810	0.442	0.414	0.406	0.128	0.123	0.217	0.123
GunPoint	2	50	0.087	0.087	0.093	0.007	0.027	0.020	0.007
Ham	2	109	0.400	0.400	0.533	0.171	0.200	0.286	0.171
HandOutl	2	370	0.199	0.197	0.202	0.304	0.304	0.340	0.301
Haptics	5	155	0.630	0.588	0.623	0.539	0.539	0.571	0.539
Herring	2	64	0.484	0.469	0.469	0.391	0.406	0.453	0.375
InlineSkate	7	100	0.658	0.613	0.616	0.724	0.689	0.645	0.645
InsectWing	11	220	0.438	0.422	0.645	0.469	0.468	0.472	0.442
ItalyPower	2	67	0.045	0.045	0.050	0.153	0.157	0.118	0.118
LargeKitchen	3	375	0.507	0.205	0.205	0.317	0.328	0.344	0.264
Lightning-2	2	60	0.246	0.131	0.131	0.344	0.279	0.295	0.246
Lightning-7	7	70	0.425	0.288	0.274	0.260	0.260	0.356	0.260
MALLAT	8	55	0.086	0.086	0.066	0.226	0.207	0.237	0.207
Meat	3	60	0.067	0.067	0.067	0.083	0.083	0.050	0.017
MedicalImages	10	381	0.316	0.253	0.263	0.262	0.234	0.243	0.234
MiddlePhlxOAG	3	154	0.260	0.253	0.250	0.210	0.207	0.215	0.207
MiddlePhlxOC	2	291	0.247	0.318	0.352	0.473	0.470	0.390	0.390
MiddlePhlxTW	6	154	0.439	0.419	0.416	0.356	0.353	0.366	0.353
MoteStrain	2	20	0.121	0.134	0.165	0.139	0.146	0.196	0.117
NonInvasFetalT1	42	1800	0.171	0.185	0.209	0.282	0.242	0.335	0.242
NonInvasFetalT2	42	1800	0.120	0.129	0.135	0.211	0.189	0.288	0.189
OSULeaf	6	200	0.479	0.388	0.409	0.306	0.252	0.347	0.252
OliveOil	4	30	0.133	0.133	0.167	0.567	0.567	0.100	0.100
PhalangesOC	2	1800	0.239	0.239	0.272	0.334	0.328	0.266	0.266
Phoneme	39	214	0.891	0.773	0.772	0.713	0.701	0.730	0.701
Plane	7	105	0.038	0.000	0.000	0.000	0.000	0.000	0.000
ProximalPhlxOAG	3	400	0.215	0.215	0.195	0.151	0.156	0.161	0.127
ProximalPhlxOC	2	600	0.192	0.210	0.216	0.223	0.230	0.168	0.168
ProximalPhlxTW	6	205	0.292	0.263	0.263	0.203	0.203	0.200	0.195
RefrigeratnDev	3	375	0.605	0.560	0.536	0.456	0.467	0.472	0.424
ScreenType	3	375	0.640	0.589	0.603	0.560	0.549	0.584	0.547
ShapeletSim	2	20	0.461	0.300	0.350	0.250	0.294	0.361	0.250
ShapesAll	60	600	0.248	0.198	0.232	0.217	0.193	0.220	0.193
SmallKitchenApp	3	375	0.659	0.328	0.357	0.325	0.315	0.347	0.315
SonyAIBOSurf	2	20	0.305	0.305	0.275	0.118	0.106	0.210	0.106
SonyAIBOSurfII	2	27	0.141	0.141	0.169	0.205	0.201	0.221	0.134
StarLightCurves	3	1000	0.151	0.095	0.093	0.146	0.145	0.137	0.137
Strawberry	2	370	0.062	0.062	0.060	0.106	0.098	0.073	0.073
SwedishLeaf	15	500	0.211	0.154	0.208	0.136	0.118	0.117	0.117
Symbols	6	25	0.100	0.062	0.050	0.057	0.056	0.133	0.052
SyntheticCtrl	6	300	0.120	0.017	0.007	0.027	0.027	0.027	0.013
ToeSegment1	2	40	0.320	0.250	0.228	0.061	0.057	0.175	0.057
ToeSegment2	2	36	0.192	0.092	0.162	0.154	0.123	0.185	0.123
Trace	4	100	0.240	0.010	0.000	0.080	0.060	0.100	0.040
TwoLeadECG	2	23	0.253	0.132	0.096	0.038	0.025	0.040	0.025
TwoPatterns	4	1000	0.090	0.002	0.000	0.028	0.023	0.034	0.022
UwaveGestureAll	8	896	0.052	0.034	0.108	0.272	0.247	0.212	0.212
uWaveGestureX	8	896	0.261	0.227	0.273	0.261	0.253	0.262	0.253
uWaveGestureY	8	896	0.338	0.301	0.366	0.397	0.384	0.360	0.360
uWaveGestureZ	8	896	0.350	0.322	0.342	0.329	0.321	0.326	0.321
Wafer	2	1000	0.005	0.005	0.020	0.011	0.009	0.010	0.009
Wine	2	57	0.389	0.389	0.426	0.389	0.389	0.296	0.167
WordSynonyms	25	267	0.382	0.252	0.351	0.498	0.486	0.420	0.420
Worms	5	77	0.635	0.586	0.536	0.459	0.448	0.492	0.448
WormsTwoClass	2	77	0.414	0.414	0.337	0.249	0.243	0.260	0.238
Yoga	2	300	0.170	0.155	0.164	0.298	0.287	0.257	0.257
Ranking			5.165	3.647	4.324	4.394	3.641	4.535	2.294

TABLE II
ERROR RATES FROM DIFFERENT CLASSIFIERS

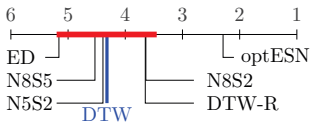

Fig. 5. Graphical representation of the Bonferroni-Dunn post-hoc test. The red line shows the two-tailedness of the critical difference when using *DTW* as a control.

85 datasets is a model with $N_x = 800$ with connectivity 0.1, $\rho(W) = 2.0$ (denoted as **N8S2**). We also want to show the influence of varying network size and spectral radius, therefore we pick two other models, $N_x = 500$, connectivity=0.1, $\rho(W) = 2.0$ (denoted as **N5S2**), and **N8S5** which is the same network architecture as **N8S2** but $\rho(W) = 5.0$. It is worth to note that a model with the parameters of $N_x = 250$, $\rho(W) = 0.55$ and an input scaling of 2.0 performs the worst, whereas the same model with an input scaling of 1.0 is ranked fifth. Table II shows error rates from all datasets of three ESN configurations. Detail information in the table are dataset name with data type indicated with a colored square in front, number of classes (nC), number of training sets (nTraining) and the error rates from 1-NN combined with three distance measure **ED**, **DTW** and **DTW-R**, three selected different ESN models, (**N5S2**, **N8S2**, **N8S5**) and lastly the best result from those fifteen models (**OptESN**). The pairwise comparisons of two classifiers are shown in Figure 4. Obviously, ESNs outperform DTW and DTW-R when the number of classes is less than 10, where most of magenta and green appear in the yellow areas in Figure 4 (top left and top right). Changing to larger spectral radii from 2.0 to 5.0 in Figure 4 (bottom left) does not always give better performance, although this improves accuracy in many cases. The output from **OptESN** testifies that high recognition rates from ESNs can only be obtained from fine tuning parameters in the network. Figure 4 on the bottom right shows the accuracy of **N8S2** and **DTW** from different types of data. There is no obvious preference of the data type in using ESN as a classifier.

In order to find a statistically significant difference among these classification methods, we follow [31] using a Friedman test for comparative studies of multiple classifiers with multiple datasets. The Friedman test is used to test the null-hypothesis that the average ranks of all algorithms are the same as the mean ranks. The computed F-distribution is 2.12 at confidence level 95%, where Friedman test gives $F_F = 20.41$. Therefore, we reject the null-hypothesis and proceed with a post-hoc test. Bonferroni-Dunn for the post-hoc test is used to control the family-wise error rate, besides it is suitable to compare all classifiers to a control. The graphical representation of the Bonferroni-Dunn test is illustrated in Figure 5 with a critical difference of 0.874. The illustration shows no significant difference between our control classifier DTW, which is considered a decent state-of-the-art approach for time-series classification, and all other classifiers except for optimal ESN (**OptESN**). Simply fine tuning the spectral radius

and adjusting input scaling for $N_x = 250$ can lead to much better performance. With some insights to the characteristic of the datasets, it is easier to optimize ESNs for particular datasets. It might be also interesting to explore the impact of other essential settings.

V. Conclusions

In this paper, we have proposed ESNs as an alternative method to solve time-series classification problems. The results show that ESNs are comparable to other state-of-the art approaches, ED and DTW; and can outperform these classifiers by choosing an appropriate network size depending on the amount of training data, and carefully selecting spectral radii and input weight scaling. Generally, benefits of using ESNs over distance measurement approaches are for example, the capability to capture linear and nonlinear effects in multi-dimensional data without data transformation and dimension reduction processes. Although DTW can be extended for multi-dimensional data, it leads to some issues e.g., the time complexity and the selection of dependent or independent warping distance function for multivariable [32]. In addition, ESNs can classify data with different length without effort, while comparing sequences of different lengths by DTW requires interpolation, which can worsen the accuracy [33]. Furthermore, ESNs offer a very simple learning mechanism and can be used for large and complex systems by optimizing the learning rules. Testing is fast, because the computation is performed only at the readout. For the case of linear regression, the complexity is similar to the least-squares.

References

[1] E. Keogh, L. Wei, X. Xi, M. Vlachos, S.-H. Lee, and P. Protopapas, "Supporting exact indexing of arbitrarily rotated shapes and periodic time series under euclidean and warping distance measures," *The VLDB Journal*, vol. 18, no. 3, pp. 611–630, Jun. 2009.

[2] G. E. Batista, B. Campana, and E. J. Keogh, "Classification of live moths combining texture, color and shape primitives," 2010, pp. 903 –906.

[3] J. Hills, J. Lines, E. Baranauskas, J. Mapp, and A. Bagnall, "Classification of time series by shapelet transformation," *Data Min. Knowl. Discov.*, vol. 28, no. 4, pp. 851–881, Jul. 2014.

[4] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista, "The ucr time series classification archive," July 2015, www.cs.ucr.edu/~eamonn/time_series_data/.

[5] A. Bagnall, A. Bostrom, J. Large, and J. Lines, "The great time series classification bake off: An experimental evaluation of recently proposed algorithms. extended version," *CoRR*, vol. abs/1602.01711, 2016.

[6] X. Xi, E. J. Keogh, C. R. Shelton, L. Wei, and C. A. Ratanamahatana, "Fast time series classification using numerosity reduction." in *ICML*, ACM International Conference Proceeding Series, vol. 148. ACM, 2006, pp. 1033–1040.

[7] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. J. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *Data Min. Knowl. Discov.*, vol. 26, no. 2, pp. 275–309, 2013.

[8] A. Bagnall, J. Lines, J. Hills, and A. Bostrom, "Time-series classification with COTE: the collective of transformation-based ensembles," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 9, pp. 2522–2535, 2015.

[9] D. F. Silva, V. de Souza, and G. Batista, "Time series classification using compression distance of recurrence plots," in *IEEE International Conference on Data Mining*, 2013, pp. 687–696.

[10] B. Fulcher, M. Little, and N. Jones, "Highly comparative time-series analysis: the empirical structure of time series and their methods," *JOURNAL OF THE ROYAL SOCIETY INTERFACE*, vol. 10, 2013.

[11] B. D. Fulcher and N. S. Jones, "Highly comparative feature-based time-series classification," *IEEE Trans. Knowl. Data Eng.*, vol. 26, no. 12, pp. 3026–3037, 2014.

[12] F. Itakura, "Minimum prediction residual principle applied to speech recognition," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 23, no. 1, pp. 67–72, Feb 1975.

[13] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD Workshop*, 1994, pp. 359–370.

[14] E. Keogh, "Exact indexing of dynamic time warping," in *Proceedings of the 28th International Conference on Very Large Data Bases*, VLDB 2002, pp. 406–417.

[15] E. Keogh and C. A. Ratanamahatana, "Exact indexing of dynamic time warping," *Knowl. Inf. Syst.*, vol. 7, no. 3, pp. 358–386, Mar. 2005.

[16] J. Serrà and J. L. Arcos, "An empirical evaluation of similarity measures for time series classification," *CoRR*, vol. abs/1401.3973, 2014.

[17] J. Lines and A. Bagnall, "Time series classification with ensembles of elastic distance measures," *Data Mining and Knowledge Discovery*, vol. 29, no. 3, pp. 565–592, 2015.

[18] H. Jaeger, "Adaptive nonlinear system identification with echo state networks," in *Advances in Neural Information Processing Systems 15 [Neural Information Processing Systems, NIPS 2002, December 9-14, 2002, Vancouver, British Columbia, Canada]*, 2002, pp. 593–600.

[19] H. Jaeger and H. Haas, "Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication," *Science*, vol. 304, no. 5667, p. 78–80, April 2 2004.

[20] M. Lukoševičius, D. Popovici, H. Jaeger, and U. Siewert, "Time warping invariant echo state networks," no. 2, May 2006.

[21] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series." in *KDD Workshop*, 1994, pp. 359–370.

[22] G. Batista, X. Wang, and E. J. Keogh, "A complexity-invariant distance measure for time series," in *Proceedings of the Eleventh SIAM International Conference on Data Mining, SDM 2011*,, pp. 699–710.

[23] J. Grabocka, A. Nanopoulos, and L. Schmidt-Thieme, "Invariant Time-Series Classification" in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2012*.

[24] L. Ye and E. J. Keogh, "Time series shapelets: a new primitive for data mining," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2009*, pp. 725–740

[25] J. Lines, L. M. Davis, J. Hills, and A. Bagnall, "A shapelet transform for time series classification," in *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2012*, pp. 289–297.

[26] M. Lukosevicius, "A practical guide to applying echo state networks," in *Neural Networks: Tricks of the Trade - Second Edition*, 2012, pp. 659–686.

[27] R. Pascanu and H. Jaeger, "A neurodynamical model for working memory," *Neural Netw.*, vol. 24, no. 2, pp. 199–207, Mar. 2011.

[28] G.-Z. Sun, H.-H. Chen, and Y.-C. Lee, "Time warping invariant neural networks," in *Advances in Neural Information Processing Systems 5*, 1993, pp. 180–187.

[29] H. Jaeger, M. Lukoševičius, D. Popovici, and U. Siewert, "Optimization and applications of echo state networks with leaky-integrator neurons," *Neural Networks*, vol. 20, no. 3, pp. 335–352, 2007.

[30] H. S. Tapp, M. Defernez, , and E. K. Kemsley*, "FTIR spectroscopy and multivariate analysis can distinguish the geographic origin of extra virgin olive oils," *Journal of Agricultural and Food Chemistry*, vol. 51, no. 21, pp. 6110–6115, 2003.

[31] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006.

[32] M. Shokoohi-Yekta, J. Wang, and E. J. Keogh, "On the non-trivial generalization of dynamic time warping to the multi-dimensional case," in *Proceedings of the 2015 SIAM International Conference on Data Mining*, pp. 289–297.

[33] O. Henniger and S. Müller, "Effects of time normalization on the accuracy of dynamic time warping," in *Proceedings of the 1st IEEE Conference on Biometrics: Theory, Applications and Systems (BTAS 2007)*, pp. 1–6.