

# **Time-Dependent Data: Classification and Visualization**

Von der Humanwissenschaften der Universität Osnabrück  
zur Erlangung der Würde eines  
Doktors der Naturwissenschaften (Dr. rer. nat.)  
genehmigte Abhandlung

Vorgelegt von

**Pattreeya Tanisaro**

aus Chaiphum, Thailand

Osnabrück, 2019

This is a cumulative doctoral thesis that includes articles published in journals and conference proceedings.

Supervisor and first examiner: Prof.Dr. Gunther Heidemann  
Co-examiner: Prof.Dr. Visvanathan Ramesh  
Prof.Dr. Gordon Pipa

# ABSTRACT

The analysis of the immensity of data in space and time is a challenging task. For this thesis, the time-dependent data has been explored from various directions. The studies focused on data visualization, feature extraction, and data classification. The data that has been used in the studies comes from various well-recognized archives and has been the basis of numerous researches. The data characteristics ranged from the univariate time series to multivariate time series, from hand gestures to unconstrained views of general human movements. The experiments covered more than one hundred datasets. In addition, we also discussed the applications of visual analytics to video data. Two approaches were proposed to create a feature vector for time-dependent data classification. One is designed especially for a bio-inspired model for human motion recognition and the other is a subspace-based approach for arbitrary data characteristics. The extracted feature vectors of the proposed approaches can be easily visualized in two-dimensional space. For the classification, we experimented with various known models and offered a simple model using data in subspaces for light-weight computation. Furthermore, this method allows a data analyst to inspect feature vectors and detect an anomaly from a large collection of data simultaneously. Various classification techniques were compared and the findings were summarized. Hence, the studies can assist a researcher in picking an appropriate technique when setting up a corresponding model for a given characteristic of temporal data, and offer a new perspective for analyzing the time series data.

This thesis is comprised of two parts. The first part gives an overview of time-dependent data and of this thesis with its focus on classification; the second part covers the collection of seven publications.



# ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor Prof.Dr.Gunther Heidemann for giving me the opportunity to do my doctorate under his supervision. During the years of his supervision, I had expansive freedom to pursue the projects that I like, for which I am truly grateful. Moreover, I really appreciate his trust and encouragement, allowing me to conduct my research confidently over the years.

Another important person who diligently proofread most of the publications is my best friend, Manfred Weis. Despite his tight working schedule, he is always there for me whenever I am in need of help. I am very much obliged and thank him from the bottom of my heart for his patience and constant support. Many thanks go also to my colleagues at IKW: Nico Müller, Dr.Mingya Liu, Axel Schaffland, Dr.Ulf Krumnack, and Dr.Julius Schönig, for useful information sharing and proofreading of my papers. They make the workplace a good place to work.

I must say I am very fortunate to be able to grow up with the unconditional love of the family. I would like to acknowledge and send my obligation to my mother who raised three kids aged six, five and two years old single-handedly. With her labor and sweat for many years, she sent two kids to engineering schools and one to a medical school. I do not know how to express enough how grateful I am for her love. My gratitude also goes to both of my brothers for always being there for me and taking care of mother in my place. I am so blessed to have them as brothers. The love of my family has given me the confidence to be who I am. Moreover, I would like to thank my friends and colleagues around me, even though I have been in a foreign country alone, I have never felt lonely. I really value our relationships.

Last but not least, I am truly lucky to be born in Thailand, a country where the opportunities are open for women at par to men. Also, I am indebted to Germany for granting me to study here. I feel just at home in Germany as I do in Thailand.



# CONTENTS

<b>Abstract</b>	<b>iii</b>
<b>Acknowledgements</b>	<b>v</b>
<b>List of Abbreviations and Acronyms</b>	<b>xi</b>
<b>List of Publications</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Definitions of Time Series Data . . . . .	4
1.2 Related Work . . . . .	4
1.2.1 Feature representation of time series data . . . . .	5
1.2.2 Data visualization . . . . .	5
1.2.3 Data classification . . . . .	7
1.3 Motivation and Major Contributions . . . . .	9
1.3.1 Datasets . . . . .	10
<b>Bibliography</b>	<b>13</b>
<b>2 Publications</b>	<b>19</b>
2.1 Visual Analytics for Video Applications . . . . .	21
2.2 Time Series Classification Using Time Warping Invariant Echo State Networks . . . . .	29
2.3 Quasi View-Independent Human Motion Recognition in Subspaces . . . . .	37
2.4 Classifying Bio-Inspired Model in Point-Light Human Motion Using Echo State Network . . . . .	45
2.5 Bidirectional Recurrent Neural Networks for Human Motion Recognition . . . . .	55
2.6 Dimensionality Reduction for Visualization of Time Series and Trajectories . . . . .	75
2.7 A Very Concise Feature Representation For Time Series Classification Understanding . . . . .	89





# LIST OF FIGURES

1.1	Pipeline for time-dependent data classification. . . . .	2
1.2	A univariate time series dataset called “Plane” from the UCR archive [10]. The data sequences which are out of sync are illustrated with brown lines. . . . .	3
1.3	The distance plot of nine instances from seven classes in the “Plane” dataset. . . . .	3
1.4	The results of 210 sequences of “Plane” in Figure 1.2a after adopting our proposed technique in <i>Tanisaro2019a</i> by employing different settings. . . . .	4
1.5	The classification approaches. . . . .	10



# LIST OF ABBREVIATIONS AND ACRONYMS

<b>AR</b>	Autoregressive
<b>BoF</b>	Bag of Feature
<b>BPTT</b>	Back Propagation Through Time
<b>BRNNs</b>	Bi-Directional Recurrent Neural Networks
<b>CNN</b>	Convolutional Neural Network
<b>COTE</b>	The Collective of Transformation-Based Ensembles
<b>DTW</b>	Dynamic Time Warping
<b>ED</b>	Euclidean Distance
<b>EDR</b>	Distance on Real Sequence
<b>ESN</b>	Echo State Network
<b>FCN</b>	Fully Convolutional Network
<b>GAF</b>	Gramian Angular Fields
<b>GPU</b>	Graphics Processing Unit
<b>HMM</b>	Hidden Markov Model
<b>HOG</b>	Histogram of Oriented Gradients
<b>kNN</b>	k-Nearest Neighbors
<b>LCSS</b>	Longest Common Subsequence
<b>LSM</b>	Liquid-State Machine
<b>LSTM</b>	Long Short-term Memory
<b>MLP</b>	Multilayer Perceptron
<b>MM</b>	Markov Model
<b>RNN</b>	Recurrent Neural Network
<b>RP</b>	Recurrence Plot
<b>SVM</b>	Support Vector Machine



# LIST OF PUBLICATIONS

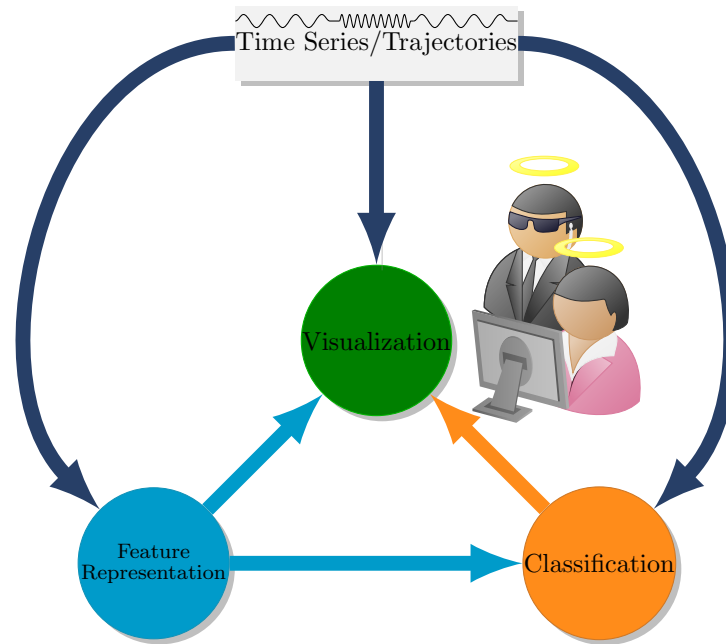
- Tanisaro2019b* P. Tanisaro & G. Heidemann. *A Very Concise Feature Representation Understanding For Time Series Classification*. In The 16th International Conference on Machine Vision Applications (MVA), 2019.
- Tanisaro2019a* P. Tanisaro & G. Heidemann. *Dimensionality Reduction for Visualization of Time Series and Trajectories*. In The 21st Scandinavian Conference on Image Analysis (SCIA), 2019.
- Tanisaro2018* P. Tanisaro & G. Heidemann. *An Empirical Study on Bidirectional Recurrent Neural Networks for Human Motion Recognition*. In The 25th International Symposium on Temporal Representation and Reasoning (TIME), pages: 21:1-21:19, 2018.
- Tanisaro2017b* P. Tanisaro, C. Lehman, L. Sütfeld, G. Pipa & G. Heidemann. *Classifying Bio-Inspired Model in Point-Light Human Motion Using Echo State Network*. In The 26th International Conference on Artificial Neural Networks (ICANN), pages: 84-91, 2017.
- Tanisaro2017a* P. Tanisaro, F. Mahner & G. Heidemann. *Quasi View-Independent Human Motion Recognition in Subspaces*. In The 9th ACM International Conference on Machine Learning and Computing (ICMLC), pages: 278-283, 2017.
- Tanisaro2016* P. Tanisaro & G. Heidemann. *Time Series Classification Using Time Warping Invariant Echo State Networks*. In The 15th IEEE International Conference on Machine Learning and Application (ICMLA), pages: 831-836, 2016.
- Tanisaro2015* P. Tanisaro, J. Schöning, K. Kurzhals, G. Heidemann & D. Weiskopf. *Visual Analytics For Video Applications*. *IT-Information Technology*, 57 : 30-36, 2015. De Gruyter.



# PART 1

## INTRODUCTION

Time-dependent data such as time series and trajectories are of fundamental importance to various application domains. The most commonly known examples are, for instance, the monitoring of stock prices and of vital signs for detecting the abnormal exceeding of a specified threshold and classifying of signals. For the past many years, applications based on neural networks have gained gigantic popularity, especially, to solve time series forecasting and classification. Since these neural network based approaches, particularly for deep network layers, yield striking results compared to the traditional methods. Furthermore, there are extensive models available in various publicly accessible frameworks. However, a remaining major problem of time-dependent data analysis is the reasoning process, how to interpret the artifacts or feature representation to understand the situation. While most of the current research focuses on modifying the network models and fine-tuning millions of hyper-parameters in order to get the best classification results; nonetheless, we still lack an understanding of the underlying intrinsic properties of that data. For the purpose of classification, the three common processes that are parts of a classification pipeline depicted in Figure 1.1 are taken into account. These three processes are i) visualization (●), to allow users to visually analyze the data. ii) feature representation (●), to extract information from the data and represent it as a feature vector. This process usually involves statistics such as correlation structure and distribution. iii) classification of the data (●). The steps which involve these three processes in the pipeline consist of i) extraction of the data to features, ii) direct visualization of the data, iii) classification of the raw data,



**Figure 1.1:** Pipeline for time-dependent data classification.

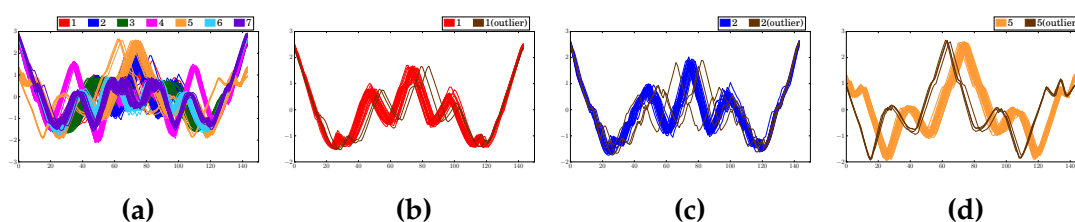
iv) visualization of the extracted features, v) classification of the data from the extracted features, and vi) visualization of the classification results.

Because of the rapid advancement in deep learning, most recent extensive range of research in time-dependent data domain put their focuses on the classification step (●) especially of the raw data; hence, the challenging steps left behind are the visualization (●) and feature extraction (●) for data understanding. A classical visualization approach is normally to use a line graph where one axis represents time and the other shows the value. An example of time series with 210 instances of a univariate time series with a fixed length of 144 is shown in Figure 1.2a. This dataset can be considered as a simple case of time series classification because each instance has only one feature and all instances in the dataset have equal lengths. Nevertheless, we can see from this illustration that it is still extremely difficult to differentiate each data group because of the data clutter, and it is almost not possible to spot the outliers of each data cluster unless a certain amount of data is loaded. Figure 1.2b-1.2d show the line plots of 30 instances of classes labeled “1”(■), “2”(■) and “5”(■) separately. Now, finding out the anomalies from the data becomes much easier. However, a situation will become more complex if each data sequence has a different length, particularly with a large number of data features in a large dataset. Time series, particularly for multivariate time series, cannot be visualized or analyzed directly. The only

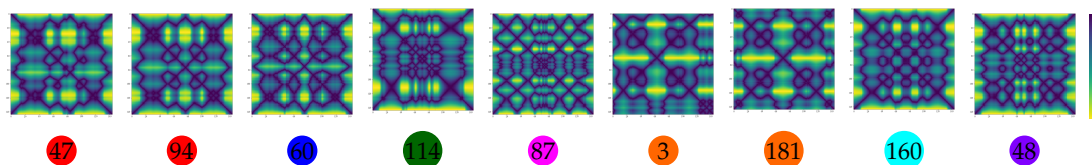


traditionally known technique employed for this purpose is the unthresholded recurrence plot (RP) [13, 19]. This technique is also known as distance plot or similarity matrix which allows a high-dimensional phase space trajectories to be visualized in subspaces through a two-dimensional representation. Figure 1.3 shows the distance plot of nine instances of seven classes from Figure 1.2. Below each instance is the label which is colored according to the data class associated with its id. We can see that the distance plot is not suitable for comparing hundreds of data simultaneously, particularly for data of unequal sequence length yielding matrices of different sizes. However, four images shown in Figure 1.4 which employed the proposed technique in *Tanisaro2019a* can give a better overview of the whole data cluster. These images can reveal the relationships of the data clusters, e.g., data class labeled “1” (■) and “2” (■) have very close relationships. Furthermore, a few outliers from Figure 1.2b and Figure 1.2c, and four outliers from 1.2d can be easily detected.

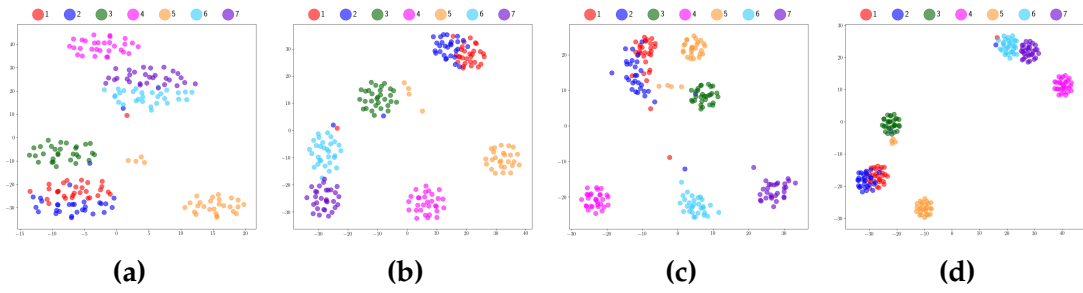
In our studies, we performed six steps in the pipeline as illustrated in Figure 1.1. The datasets used in the studies are categorized into four groups: i) univariate time series. ii) multivariate time series. iii) trajectories of human motion from the Motion Capture (MoCap), and iv) videos. The term time series may refer to trajectories of human motions in this thesis.



**Figure 1.2:** A univariate time series dataset called “Plane” from the UCR archive [10]. The data sequences which are out of sync are illustrated with brown lines.



**Figure 1.3:** The distance plot of nine instances from seven classes in the “Plane” dataset.



**Figure 1.4:** The results of 210 sequences of “Plane” in Figure 1.2a after adopting our proposed technique in *Tanisaro2019a* by employing different settings.

## 1.1 Definitions of Time Series Data

Most statistical packages offer time series plots such as line plot which is usually a univariate time series using a two-dimensional plot, where one axis shows time e.g. seconds, minutes, weeks and so on (usually displayed horizontally) while the other axis shows the data. However, in our work, we employed only time series with a fixed number of evenly spaced time points for an arbitrary number of feature dimensions.

**Definition 1.** A time series or trajectory  $X$  of any high-dimensional data of  $m$  variables which represent the measurement of a quantity over a period of time is defined as pairs of time  $t = 1, 2, 3 \dots T$  and  $x(t) \in \mathbb{R}^m$ :

$$X = [(t_1, x_1), (t_2, x_2), \dots, (t_T, x_T)] \quad (1.1)$$

where  $X \in \mathbb{R}^{m \times T}$ .

For a classification of multiple data instances, we extend the Definition 1 accordingly.

**Definition 2.** For a set of time series data  $D$ , with a given number  $p$  of time series data, for any data indexed  $i$  in  $p$ , the input matrix  $X$  from Equation 1.1 can also be written as  $X^i$ ,  $i = 1, \dots, p$ . The set of individual data sequences is specified by  $X^i \in \mathbb{R}^{m \times T_i}$  and  $T_i$  is an arbitrary length in the set of  $X$ . For any  $X^i$ , there is a corresponding label  $y^i$  of a class  $k$ . ( $k \in K$ ) where  $K$  is the number of classes.

## 1.2 Related Work

### 1.2.1 Feature representation of time series data

Most work in time-dependent data analysis tends to plot the signals over time, hence the applicability of these methods is restricted by the number of signals that can be visualized concurrently. As a consequence, these methods do not support the comparison of feature vectors nor of the signals which are required in the analysis of a classification task. Nonetheless, an interesting technique that adds to the analysis of feature vectors by means of visualization can be found in [22]. This method employed the technique of the unthresholded recurrence plot in combination with the Bag-of-Features (BoF) obtained from the Histogram of Oriented Gradients (HOG) to create a feature representation for view-independent classification of human motions. However, the drawback of this approach is that the length of all human motion data must be truncated to equal length and cannot be displayed simultaneously. Another intriguing method which was proposed by [44] introduced an approach to encode univariate time series into images by using a polar coordinate called Gramian Angular Fields (GAF) system, generating a Gramian matrix. The classification of this method is utilized by convolutional neural networks.

### 1.2.2 Data visualization

Some visualization techniques to analyze various time series and trajectories will be briefly discussed considering from application domains such as medicine, economy and finance, science, industries and a few others.

An example of employing visualization in economy is an interactive tool developed by [40] which introduced a three dimensional hierarchical circular and column plot. It took time dimension into account to interpret the database of monthly European inflation data. Furthermore, two techniques presented in [46] enable users to interactively analyze large time series data in stock markets. The first technique lets users visually analyze combinations of assets, market sectors as well as countries. The second technique allows a user to cluster a selection of data and analyze the distribution of the assets. Traditional clinical data visualization primarily focuses on analyzing vital signs and laboratory results in one dimension and the abnormality detection is based on whether the value of an individual parameter falls below or above the threshold. However, an unusual process introduced in [35] created a multivariate time series amalgam (MTSA) of physiological data and laboratory results from univariate time series. With this information, physicians can visually interpret various aspects of the laboratory observations. Combining such univariate results can provide greater insight, which will lead to a better diagnosis. In addition, [34] recently presented a framework called *m-TSNE* to visualize two healthcare

datasets: one from its study to quantify the activity levels of cancer patients undergoing chemotherapy treatment, and the other an EEG dataset from the UCI archive [27]. Another interesting technique which applied dimensionality reduction for time series is found in [42]. It introduced a factor decomposition specific for time series that builds upon Bayesian multivariate autoregressive model (MAR) from a low-rank estimation of the autoregressive matrices. The algorithm was tested using neuroimaging data from EEG and ECoG.

Another application of dimensionality reduction and clustering in neuroscience can be found in [7]. It proposed a method to cluster the white matter (WM) fibers in the brain by considering the position of fibers in three-dimensional space, which helps brain researchers to understand and predict the effect of certain neurodegenerative pathologies. It can be of great use in neurosurgical planning to interactively guide the surgeon during the operation.

An interactive technique in the field of geology called screen-space metric was introduced in [12]. The tool provides a metrics to quantify important patterns which allow the data scientists to navigate through different time steps. Not only can the geologist confirm the hypotheses about the chemical reactions with this approach, one can also formulate new hypotheses. Moreover, some applications in the industries include, for example, a frequent-pattern analysis tool designed by [16] for data center cooling and oil well production. It employed motif layout, motif distortion and motif merging to provide insights into the recurring patterns. An application for analyzing weather data from cities in the US was demonstrated in [32]. It provided an interactive tool as a plugin for the web to visualize multidimensional time series data using MDS. The tool allows data analysts to adjust the weights of distance measure for each time series as well as to choose available algorithms. An example of analyzing large network traffic data is an interactive visualization technique called ClockMap proposed in [14]. It combined a circular nested treemap layout with a circular glyph with zooming and panning functionalities. Additionally, an interesting technique called Temporal Multidimensional Scaling (TMDS) presented in [20] could reveal two threats to network security, namely, a distributed brute-force attack, and port scans. It computes a temporal one-dimensional MDS plot from multivariate time dependent data. This work is pretty close to our work in the sense that it also applies the dimensionality reduction to the high-dimensional data but is different in that this technique keeps tracking data over time. Another technique which based on Multidimensional Scaling (MDS) by adding a temporal component for visualization called “time curves” can be found in [1]. It shows the use cases in several applications such as document histories, precipitation patterns and fMRI data but it does not support the precise reading of an output, the data analyst needs to be trained in order to make an interpretation of the line curve symbols for a specific application. Furthermore, a technique

called “visualizing time-dependent data using dynamic t-SNE” [37], albeit the paper’s name mentioned “time-dependent data”, but in fact, it proposed a technique for a non-time-series by just using several time steps to generate a better view of data. The results of applying this technique were demonstrated using the MNIST handwritten digit database and SVHN (Street View House Number) dataset, not the time series data.

Hence, we can conclude that a general solution to visualize multivariate time-dependent data is to combine user interaction and visualization techniques into a visual analytics pipeline. The interactive exploration permits users to switch between overview and detailed analysis through zooming and panning the data. Thus, a reasoning process such as detecting the anomalies in the data usually has to be executed by the data analyst. Such methodologies as those aforementioned put focus on inspecting signals over time. Therefore, they are not suitable to inspect hundreds of signals simultaneously.

### 1.2.3 Data classification

We looked at the classification approaches by grouping them into seven categories. In our studies, we compare outputs based on some of these methodologies. These seven categories as illustrated in Figure 1.5 are:

- **Model-based.** Known model-based approaches for time series classification are, for instance, Autoregressive model (AR), Markov Model (MM) and Hidden Markov Model (HMM). Yet, these model-based approaches have several drawbacks which limit their applicability; therefore, they are not favored for general time series classification tasks. An interesting recent work, however, can be found in [36] that introduced a technique called the Hidden-Unit Logistic Model (HULM) which is similar to Hidden-state Conditional Random Field (HCRF). The difference lies in the fact of how the hidden units are defined. The HRCF uses a single multinomial unit much like an HMM, whereas the HULM uses a large number of binary stochastic hidden units to represent the latent state.
- **Distance-based with similarity measure.** The distance-based technique is perhaps the most commonly known solution for time series classification, especially for univariate time series. Generally after employing a distance function and measuring the similarity between two time series, a classifier such as k-Nearest Neighbors (kNN) or Support Vector Machine (SVM) can be directly applied. These well-recognized approaches are, for example, Euclidean Distance (ED), Dynamic Time Warping (DTW) [6, 25, 26], Edit Distance on Real Sequence (EDR) [9], Longest Common Subsequence (LCSS) [43] and other variants. A recent method which combines raw

and normalized data with DTW being tested for multivariate instead of univariate time series can be found in [30].

- Reservoir RNN. This is a type of neural network where an input signal is fed into the random dynamic system called reservoir and the training is performed only at the readout stage. Because the training is performed with a small amount of neurons at the readout without relying on back-propagation of the gradient, the training is quite fast. There are two types of reservoir computing, the Liquid-State Machine (LSM) [31] and Echo State Network (ESN) [21].
- Gradient-based RNN. The gradients are obtained by a technique called Back Propagation Through Time (BPTT) which is used to update weights in RNNs. The networks of RNNs are unrolled, errors are calculated and accumulated across each timestep, then rolled-up the network and the weights updated. A well-known special architecture for the gradient-based method called Long Short-term Memory (LSTM) [18] is a type of gated recurrent unit which is well-suited for time series classification and predictions. It was designed to solve the exploding and vanishing gradient problems which were encountered with training traditional RNNs. It started to get attention after a groundbreaking achievement of sequence labeling in speech recognition in [15], which employed LSTM in combination with a Bi-Directional Recurrent Neural Networks (BRNNs) to form a deep Bidirectional LSTM (DB-LSTM). The accomplishments of LSTM have been adopted in many commercial products by major tech companies, for instance, Google translate and speech recognition in smartphone by Google, Siri by Apple and Amazon Alexa by Amazon. The extension architectures of LSTM, which are currently well-recognized for time series classification benchmarking, is the combination of LSTM and Fully Convolutional Network (FCN) called LSTM-FCN introduced by [23, 24]. FCN [29, 39] originally was designed for image semantic segmentation. It has several advantages over traditional convolutional networks i.e., it tries to learn local image representations and can handle different input sizes. Examples of models which are derived from LSTM-FCN are MLSTM-FCN and MALSTM-FCN [24].
- CNN-based. Here, we separated approaches from Convolutional Neural Network (CNN)-based from the RNN because even though these two approaches adopted the gradient descent to compute the changing weights during the training; their network architectures are different. The CNN-based methods for time series classification can be found in some works, for example, [45] adopted the pure FCN for classifying data in the UCR archive. It outperforms the COTE [3], MLP and ResNet [17] testing on 44

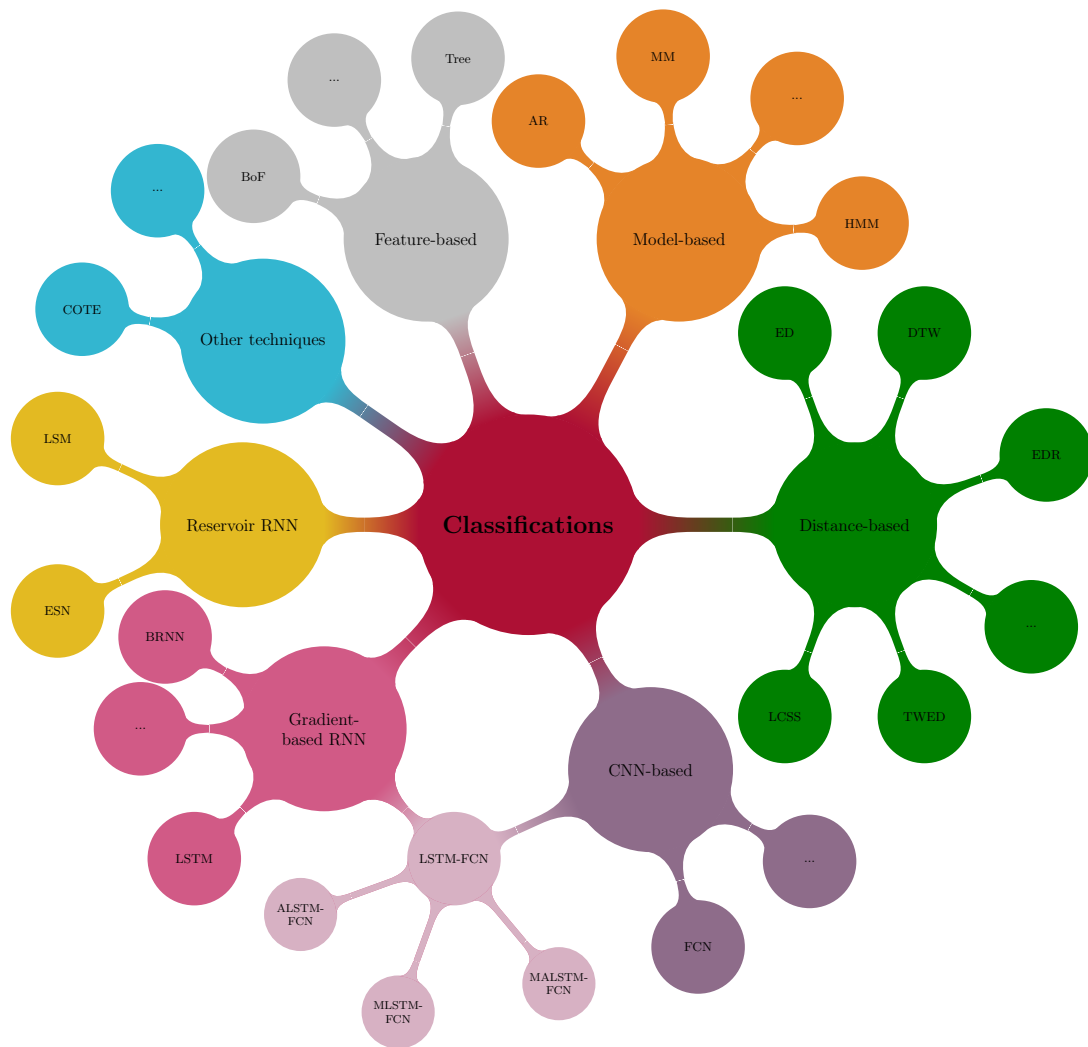
selected datasets. Furthermore, [41] proposed to combine both CNN and DB-LSTM for action recognition in video data. An architecture-variation of convolutional network called temporal convolutional network (TCN) which adopted one-dimensional FCN is extensively discussed in [4]. The results of using TCN outperformed other RNNs on most datasets.

- Feature-based. Some appealing works based on the extracted features of the time series data are, for example, the work of [38] which proposed to use Bag-of-Pattern (BoP) model to form a feature vector, while [5] suggested to use a tree-base learning strategy to discover the patterns from time series segments.
- Other techniques. By far, the best classifier other than a Neural Network based and is completely tested for 85 datasets of the UCR archive, is an ensemble-based technique called COTE [28, 3, 2].

### 1.3 Motivation and Major Contributions

The prime objective in our studies is to understand the patterns in temporal data, specifically, time series and trajectories, used for classification. We had proposed some techniques for feature representations and visualization. In addition, we also performed empirical comparative studies of several state-of-the-art approaches for time series classification, i.e., we had performed the tasks shown in Figure 1.1, where the studies were done in seven publications for those six processes in the pipeline. Each process was demonstrated in the related publications listed as follows.

- Visualization of time series (www → ●). The discussion of illustration of trajectories from videos is shown in *Tanisaro2015*. The visualization technique using distance plots as a reference can be found in *Tanisaro2018*, *Tanisaro2019a* and *Tanisaro2019b*.
- Our proposed methods to extract features from the raw time series data (www → ●) are demonstrated in *Tanisaro2017a*, *Tanisaro2017b*, *Tanisaro2019a* and *Tanisaro2019b*. *Tanisaro2017a* and *Tanisaro2017b* focus on getting feature representation for human motion recognition, while *Tanisaro2019a* and *Tanisaro2019b* show a general model applied on diverse types of time series and trajectories data.
- The classification results of raw data (www → ●). We employed various classification techniques for various datasets shown in *Tanisaro2016*, *Tanisaro2017a*, *Tanisaro2018*, and *Tanisaro2019b*. The big picture of classification approaches is illustrated in Figure 1.5 and discussed in section 1.2.3.



**Figure 1.5:** The classification approaches.

- Our proposed feature representations (● → ●) have been clarified by visualization in *Tanisaro2017b*, *Tanisaro2019a*, and *Tanisaro2019b*.
- The classification results of the extracted features (● → ●) have been provided in *Tanisaro2017a*, *Tanisaro2017b*, and *Tanisaro2019b*.
- The classification results (● → ●) are visually explained in *Tanisaro2019b*.

### 1.3.1 Datasets

There are many well-known sets of time-dependent data publicly available. However, in the studies we limited the data from four aspects:



1. The univariate time series from the UCR archive [10] which consists of 85 datasets. It is the largest archive for time series classification benchmarking.
2. The multivariate time series of 15 datasets obtained from [5]. Recently, this archive has started to become a de facto standard for benchmarking multivariate time series classification.
3. Four Motion Capture (MoCap) datasets i.e., three human motion recognition from CMU [11], HDM05 [33], and UTD-MHAD [8], and hand gesture recognition from MHAD [8]. In our studies, we put focus on view-independent motion classification and separation of train versus test subjects while most of other research does not condition on these two issues.
4. Videos. The discussion of video data is found in *Tanisaro2015*.



## BIBLIOGRAPHY

- [1] B. Bach, C. Shi, N. Heulot, T. Madhyastha, T. Grabowski, and P. Dragicevic. Time curves: Folding time to visualize patterns of temporal evolution in data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):559–568, Jan. 2016. 6
- [2] 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*, 2016. 9
- [3] 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.*, 27(9):2522–2535, 2015. 8, 9
- [4] S. Bai, J. Z. Kolter, and V. Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *CoRR*, abs/1803.01271, 2018. 9
- [5] M. G. Baydogan and G. Runger. Time series representation and similarity based on local autopatterns. *Data Mining and Knowledge Discovery*, 30(2):476–509, Mar 2016. 9, 11
- [6] D. J. Berndt and J. Clifford. Using dynamic time warping to find patterns in time series. In U. M. Fayyad and R. Uthurusamy, editors, *KDD Workshop*, pages 359–370. AAAI Press, 1994. 7
- [7] F. Cauteruccio, C. Stamile, G. Terracina, D. Ursino, and D. Sappey-Mariniery. An automated string-based approach to white matter fiber-bundles clustering. In *2015 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, July 2015. 6
- [8] C. Chen, R. Jafari, and N. Kehtarnavaz. *UTD-MHAD: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor*, volume 2015-December, pages 168–172. IEEE Computer Society, 12 2015. <http://www.utdallas.edu/~kehtar/UTD-MHAD.html>. 11

## 14 Bibliography

- [9] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*. 7
- [10] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista. The UCR time series classification archive, 2015. ix, 3, 11
- [11] CMU Graphics Lab. CMU Motion Capture Database. <http://mocap.cs.cmu.edu/>. Online; 11
- [12] A. Dasgupta, R. Kosara, and L. Gosink. Vimtex: A visualization interface for multivariate, time-varying, geological data exploration. In *Proceedings of the 2015 Eurographics Conference on Visualization, EuroVis '15*, pages 341–350. Eurographics Association, 2015. 6
- [13] J. P. Eckmann, O. S. Kamphorst, and D. Ruelle. Recurrence plots of dynamical systems. *Europhysics Letters*, 4, Nov. 1987. 3
- [14] F. Fischer, J. Fuchs, and F. Mansmann. ClockMap: Enhancing Circular Treemaps with Temporal Glyphs for Time-Series Data. In M. Meyer and T. Weinkauff, editors, *EuroVis - Short Papers*. The Eurographics Association, 2012. 6
- [15] A. Graves, N. Jaitly, and A.-r. Mohamed. Hybrid speech recognition with Deep Bidirectional LSTM. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 273–278. IEEE, Dec. 2013. 8
- [16] M. Hao, M. Marwah, H. Janetzko, R. Sharma, D. Keim, U. Dayal, D. Patnaik, and N. Ramakrishnan. Visualizing frequent patterns in large multivariate time series, 2011. 6
- [17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. 8
- [18] S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In Kremer and Kolen, editors, *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press, 2001. 8
- [19] J. S. Iwanski and E. Bradley. Recurrence plots of experimental data: To embed or not to embed? *Chaos*, 8(4), 1998. 3
- [20] D. Jäckle, F. Fischer, T. Schreck, and D. A. Keim. Temporal mds plots for analysis of multivariate data. *IEEE Transactions on Visualization and Computer Graphics*, 22(1):141–150, Jan 2016. 6

- [21] H. Jaeger and H. Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. *Science*, 304(5667):78–80, 2004. 8
- [22] I. N. Junejo, E. Dexter, I. Laptev, and P. Perez. View-independent action recognition from temporal self-similarities. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):172–185, Jan. 2011. 5
- [23] F. Karim, S. Majumdar, H. Darabi, and S. Chen. LSTM fully convolutional networks for time series classification. *CoRR*, abs/1709.05206, 2017. 8
- [24] F. Karim, S. Majumdar, H. Darabi, and S. Harford. Multivariate lstm-fcns for time series classification, 2018. 8
- [25] E. Keogh. Exact indexing of dynamic time warping. In *Proceedings of the 28th International Conference on Very Large Data Bases, VLDB '02*, pages 406–417. VLDB Endowment, 2002. 7
- [26] E. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowl. Inf. Syst.*, 7(3):358–386, Mar. 2005. 7
- [27] M. Lichman. UCI machine learning repository, 2013. 6
- [28] J. Lines and A. Bagnall. Time series classification with ensembles of elastic distance measures. *Data Mining and Knowledge Discovery*, 29(3):565–592, 2015. 9
- [29] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3431–3440, 2015. 8
- [30] M. Luczak. Combining raw and normalized data in multivariate time series classification with dynamic time warping. *Journal of Intelligent and Fuzzy Systems*, 34(1):373–380, 2018. 8
- [31] W. Maass, T. Natschläger, and H. Markram. Real-time computing without stable states: A new framework for neural computation based on perturbations. *Neural Comput.*, 14(11):2531–2560, Nov. 2002. 8
- [32] S. Martin and T.-T. Quach. Interactive visualization of multivariate time series data. In *Proceedings, Part II, of the 10th International Conference on Foundations of Augmented Cognition: Neuroergonomics and Operational Neuroscience - Volume 9744*, pages 322–332, New York, NY, USA, 2016. Springer-Verlag New York, Inc. 6

## 16 Bibliography

- [33] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. Technical Report CG-2007-2, Universität Bonn, June 2007. 11
- [34] M. Nguyen, S. Purushotham, H. To, and C. Shahabi. m-tsne: A framework for visualizing high-dimensional multivariate time series. *CoRR*, abs/1708.07942, 2017. 5
- [35] P. Ordóñez, M. DesJardins, C. Feltes, C. U. Lehmann, and J. Fackler. Visualizing multivariate time series data to detect specific medical conditions. *AMIA Annual Symposium Proceedings*, pages 530–534, November 2008. 5
- [36] W. Pei, H. Dibeklioglu, D. M. J. Tax, and L. van der Maaten. Multivariate time-series classification using the hidden-unit logistic model. *IEEE Transactions on Neural Networks and Learning Systems*, 29(4):920–931, April 2018. 7
- [37] P. E. Rauber, A. X. Falcão, and A. C. Telea. Visualizing time-dependent data using dynamic t-sne. In *Proceedings of the Eurographics / IEEE VGTC Conference on Visualization: Short Papers, EuroVis '16*, pages 73–77, Goslar Germany, Germany, 2016. Eurographics Association. 7
- [38] P. Schäfer and U. Leser. Multivariate time series classification with WEASEL+MUSE. *CoRR*, abs/1711.11343, 2017. 9
- [39] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.*, 39(4):640–651, Apr. 2017. 8
- [40] T. Tekusová and T. Schreck. Visualizing time-dependent data in multivariate hierarchic plots - design and evaluation of an economic application. In *Proc. Int. Conference on Information Visualization*, pages 143–150. IEEE Computer Society, 2008. 5
- [41] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad, and S. W. Baik. Action recognition in video sequences using deep bi-directional lstm with cnn features. *IEEE Access*, 6:1155–1166, 2018. 9
- [42] D. Vidaurre, I. Rezek, S. L. Harrison, S. S. Smith, and M. Woolrich. Dimensionality reduction for time series data. *ArXiv e-prints*, June 2014. 6
- [43] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. Keogh. Indexing multi-dimensional time-series with support for multiple distance measures.

In *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '03, pages 216–225, New York, NY, USA, 2003. ACM. 7

- [44] Z. Wang and T. Oates. Encoding time series as images for visual inspection and classification using tiled convolutional neural networks. In *Workshops at the Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015. 5
- [45] Z. Wang, W. Yan, and T. Oates. Time series classification from scratch with deep neural networks: A strong baseline. *CoRR*, abs/1611.06455, 2016. 8
- [46] H. Ziegler, M. Jenny, T. Gruse, and D. A. Keim. Visual market sector analysis for financial time series data. In *IEEE VAST*, pages 83–90. IEEE Computer Society, 2010. 5





**PART** 

**PUBLICATIONS**



## 2.1 Visual Analytics for Video Applications

**Abstract:** In this article, we describe the concept of video visual analytics with a special focus on the reasoning process in the sensemaking loop. To illustrate this concept with real application scenarios, two visual analytics tools one for video surveillance and one for eye-tracking data analysis that cover the sensemaking process, are discussed in detail. Various aspects of video surveillance such as browsing and playback, situational awareness, and deduction of reasoning from visual analytics are examined. On account of the visual analysis of recorded eye tracking data from watching videos, application features such as a space-time cube, spatio-temporal clustering, and automatic comparison of multiple participants are reviewed of how they can support the analytical process. Based on this knowledge, open challenges in video visual analytics are discussed in the conclusion.

**Originally published in:** *it-Information Technology*, 57: 30-36, 2015. De Gruyter.

**DOI:** <http://dx.doi.org/10.1515/itit-2014-1072>



# Visual Analytics for Video Applications

Pattreeya Tanisaro, Julius Schöning, Kuno Kurzhals, Gunther Heidemann, Daniel Weiskopf

---

**Abstract:** In this article, we describe the concept of video visual analytics with a special focus on the reasoning process in the sensemaking loop. To illustrate this concept with real application scenarios, two visual analytics tools one for video surveillance and one for eye-tracking data analysis that cover the sensemaking process, are discussed in detail. Various aspects of video surveillance such as browsing and playback, situational awareness, and deduction of reasoning from visual analytics are examined. On account of the visual analysis of recorded eye tracking data from watching video, application features such as a space-time cube, spatio-temporal clustering, and automatic comparison of multiple participants are reviewed of how they can support the analytical process. Based on this knowledge, open challenges in video visual analytics are discussed in the conclusion.

**ACM CCS:** Human-centered computing → Visual analytics;

**Keywords:** Visual analytics; video data; video surveillance; eye tracking; sensemaking

---

## 1 Introduction

Due to the high complexity of video data in the spatial and temporal domain, the analysis of large amounts of recorded video material is a challenging task. There has been a lot of effort to provide computer vision techniques to extract meaningful information, nevertheless, automated computer vision is not yet powerful enough for reliably detecting anomalies, tracking objects or higher reasoning. To bridge this gap, automatic approaches can be combined with video visualization in visual analytics systems, to support the analyst in observing events, and in the formulation and evaluation of hypotheses. Hence, in order to make the vast amount of complex video data accessible, applications, tools, and methods for analysis must be used to take some load off the analyst. In the following, we will describe the concept of video visual analytics based on Höferlin et al. [15, 6, 14, 4, 8, 16, 9, 12, 7, 10, 17] with the main focus on reasoning sensemaking. With two different scenarios, (i) video surveillance and (ii) eye tracking video analysis, we discuss how the concept of video visual analytics can be applied in such cases.

## 2 Concept of Video Visual Analytics

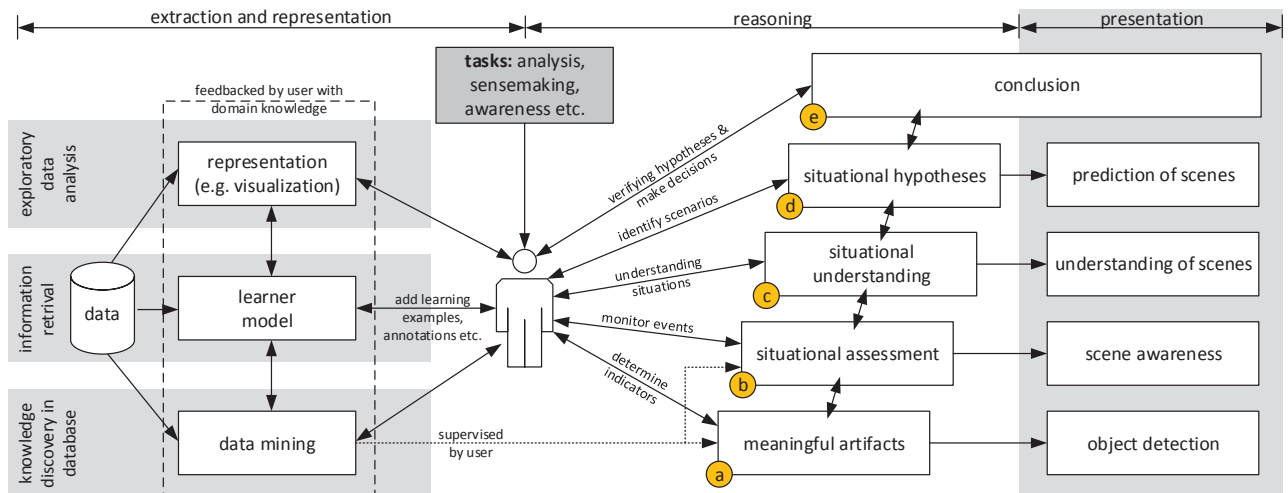
The overall purpose of visual analysis tools and techniques is “to synthesize information and derive insight from massive, dynamic, ambiguous, and often conflicting data; detect the expected and discover the unex-

pected; provide timely, defensible, and understandable assessments; and communicate assessment effectively for action” [26]. To cope with large amounts of video data and often vaguely defined analysis tasks, a sophisticated concept with a robust architecture is necessary for video visual analytics.

An overview of the architecture of video visual analytics is shown in Figure 1. This architecture relies on the *integrative view on visual analytics and sense-making processes* [17], the *interactive learning of ad-hoc classifiers for video visual analytics* [8], the *visual analytics process* [18] and the *sensemaking process* [25]. The basis of video visual analytics is formed by the principle of **sensemaking**, which can be split into two main parts, the **extraction and representation** and the **reasoning**. In addition to the sensemaking process by an analyst, i.e. a domain expert, the **presentation** provides the result including all interim results for the stakeholders. As a remark, instead of the term “representation”, the more common term “visualization” is often be used. Due to the fact that we believe, that visualization by itself may not be the best solution to encode the data for the analyst, we use the term representation.

### 2.1 Sensemaking

For sensemaking, an analyst uses her own cognition and mind for answering a specific question based on data supported by representation tools like visualization, and by organizing tools. This cognitive process, - the sense-



**Figure 1:** Architecture of video visual analytics. The sensemaking process by an analyst consists of *extraction and representation* and *reasoning*. The outcomes of each reasoning step are given in the *presentation*.

making process [25, 24] - is the general idea behind the architecture of video visual analysis, as shown in Figure 1. The core of sensemaking is an iterative learning loop. This loop is initiated by the human analyst, with her domain knowledge and the task description. The analyst sets up the analysis, by extracting and encoding the raw data to get first insights, meaningful artifacts like vehicles, ideas how to assess situations or scenes, and comes up with first assumptions. The meaningful artifacts consist of individual objects or a bundle of objects and additional information like physical properties. With this knowledge, the analyst alters the previous extraction and representation of the data. As a result of each extraction and representation, nonrelevant data such as scenes without an artifact will be minimized or completely filtered out. Based on this second extraction and representation, the analyst starts the second reasoning process and gets further insights. This loop is repeated until the result of the reasoning has reached a certain threshold to fulfill the task or after exceeding a defined time. Russell et al. [25] made two important remarks on the sensemaking: First, sensemaking returns a result at any time, but the quality of the result increases in each iteration. Second, the most time consuming part of sensemaking is the extraction and representation of the data.

### 2.1.1 Extraction and Representation

The goal of the extraction and representation part is filtering out nonrelevant data, minimizing the complexity of the data and creating meaningful representations for the human analyst. In the architecture of Figure 1, the extraction of the data is done semiautomatically by a learner model [8]. The analyst with her knowledge about the task and the domain adds learning examples and defines analysis filters, like a trajectory filter [15] or a spatial filter [12]. The learner model sets up the data mining algorithm and the representation. To be able to man-

age the amount of data, it is necessary to provide and support the analyst with additional techniques. One of the most important requirements for visual analytics is information scalability [26], to enable the analyst to concentrate only on important parts for her reasoning. Further important techniques are the automatic extraction of meaningful artifacts and the automatic assessment of situations. These mentioned techniques should be supervised by the analyst to ensure the quality claims on the whole analysis. A more detailed explanation of the extraction and representation of video data can be found in Höferlin et al. [17].

### 2.1.2 Reasoning

The idea of reasoning is to achieve task-specific conclusions by applying human judgment [27]. The conclusions, as combination of evidence and assumptions, contain the insight of the analyst when going through the sensemaking process. In previous papers by Höferlin et al. [17, 8, 15], this reasoning process is represented as a black box. To identify potential improvements and deficiencies within the reasoning process of visual analysis tools, we integrate a detailed reasoning process into the architecture of video visual analysis. As seen in Figure 1 (a)-(e) the reasoning process can be split up into five elements and may be considered as a stepwise process in which the complexity of reasoning increases with every step:

(To gain a better sense of each element of reasoning, we will describe the entire process based on a simple exemplary task: The behavior analysis of one specific man, based on the video data of one surveillance camera.)

- (a) The first step in reasoning is to determine indicators which identify **meaningful artifacts**. Due to the cooperation between the learner model set up by the analyst and the data mining, the automatic extraction of meaningful artifacts will be represented to the analyst. Under analyst's supervision, artifacts can be

detected and annotated automatically. For uncomplicated scenes this step can be performed automatically. If we map this step on our exemplary task, meaningful artifacts observed by the surveillance camera would be e.g., a traffic light, a street, a pavement, a man or a bag.

- (b) **Situational assessment** can only be achieved by monitoring expected and unexpected events, such as the interaction or the movement of meaningful artifacts. This enables the analyst to nurture situational awareness. The monitoring of events can also be supported by analytics tools, where the analyst defines restrictions in terms of space and time or key trajectories. As a result, the analyst will be notified by a visual or auditive signal created by the tool [15]. In our example task, the analyst will be alerted, if a man with a bag crossing the street and the analyst become aware of it.
- (c) For the steps (c)-(e), the requirements to human judgment are much higher than in the previous steps. As a consequence, opportunities for computational support are decreasing strongly. This is mainly due to the time span, which is relatively long compared to earlier steps, but necessary for **situational understanding**. The analyst understands the situation based on historic and current events she draws attention to. In our example, we become aware of two events. In the first event, one man with an empty bag crosses the street. In the second event, some minutes later, the same man crosses the street from the opposite direction with a filled bag. The likeliest interpretation of this scene is that the man went shopping.
- (d) In a penultimate step the analyst identifies possible hypotheses and their probability of occurrence. Based on these **situational hypotheses** the analyst can identify and predict patterns which might be relevant for the task. To project onto our example, we recognize the man who went shopping several times. So possible hypotheses are e.g., “he went shopping every day” or “he went shopping every Monday and Friday”.
- (e) The **conclusion** completes the reasoning process. The analyst verifies the hypotheses until a certain threshold is reached or a certain time is spent and concludes the reasoning with a result. The result of our example is that the man went shopping every two days at 5pm.

## 2.2 Presentation

Summarizing the result including all intermediate steps into a package of materials in the way that decision makers can relive the parts of the analysis is the goal of the presentation. How this could be done, is not part of our architecture and has to be considered separately.

## 3 Related Work

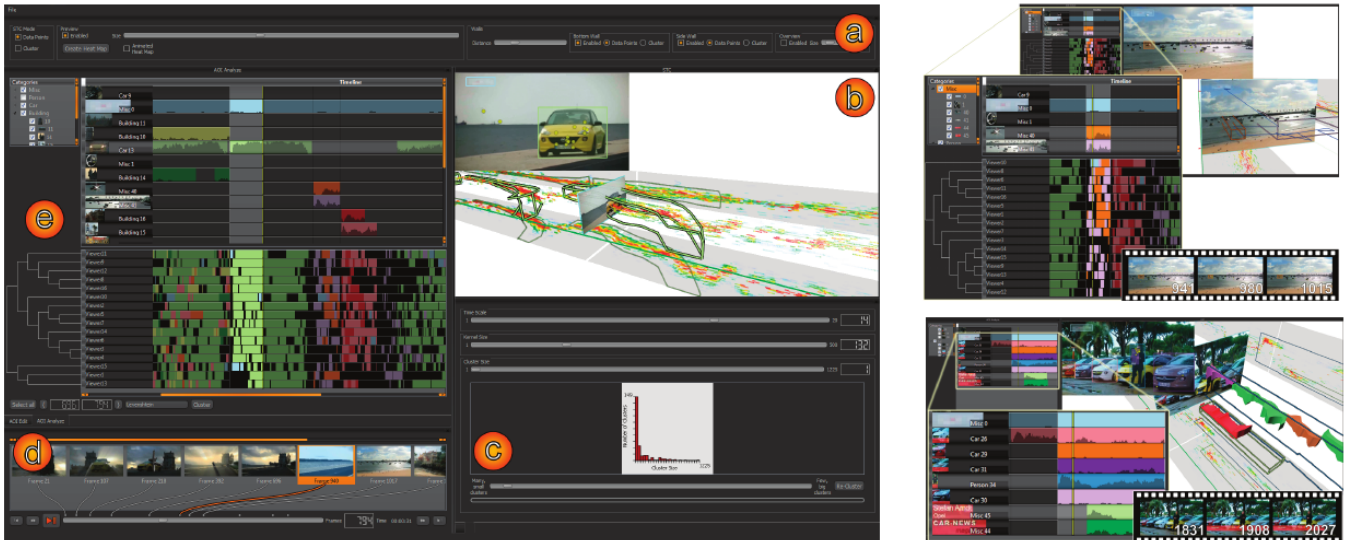
For many years, video visual analysis has played an important role only in security and entertainment industries. Recently, a wide range of applications and challenges in the areas of medicine, sports and science have arisen.

An example for visual analytics in medicine with a glyph-based visualization [2] helps clinical scientists interpreting and comparing sperm motility measurements captured from video. For the application of visual analytics in sports, a snooker coaching system [11] employs video visualization by involving coaches and players in the loop of intelligent reasoning, where the output points out the difficulties of the automated semantic reasoning. A real-time sport performance analysis using glyph layout at different scales in interactive visualization for rugby is presented by [21]. This approach helps coaching staff and team analysts to examine actions and event, and assists in the decision making during the matches. For the surveillance system, [22] introduces an interactive exploration of video surveillance data to analyze the moving objects and their properties.

In this article, we discuss two promising video visual analytics projects in detail. A video surveillance system presented by Höferlin et al. [4, 8, 16, 9, 10] which comprises many important key features in surveillance. The effectiveness of their approach for the analysis of video surveillance data was demonstrated in the IEEE VAST Challenge 2009. The second visual analytics approach which will be examined is a novel tool to analyze eye tracking data recorded from video. This approach includes methods for the spatiotemporal analysis of gaze data in context of the video and allows users to annotate and investigate Areas of Interest (AOIs).

## 4 Visual Analytics for Video Surveillance

Höferlin et al. propose a system that engages a human analyst in several layers of processing, for instance, training classifiers by using an interactive learning approach. The system authorizes users to directly adjust the complex classifier models [8]. As a consequence, users build up the trust from the training process generated by the models. It also provides a detail-on-demand technique for selected parts of data, as well as the filter feedback to verify the hypotheses [15]. Considering the common models in video surveillance according to search target intervals, they can be classified as online and offline processing. In case of offline processing analysis, the vast amount of historic data in a time period is diagnosed (Section 4.1). Online processing which is capable of real-time task analysis will be discussed in section 4.2 and analysis and reasoning in section 4.3.



**Figure 2:** ISeeCube: Visual Analysis of Gaze Data for Video [19]. Left: The main component areas are (a) viewer controls (b) space-time view (c) parameter controls. (d) video controls and key frames (e) timeline view. Top-Right: Outlier detection and individual viewer behavior. Bottom-Right: Distribution of attention.

#### 4.1 Video Browsing and Playback

Video navigation techniques allow users to browse the video, increase the playback speed to reduce the time to watch irrelevant parts of the video, and gain insight from its content. Browsing the video, particularly the recorded video footage of video surveillance is a crucial task. By contrast, typical fast-forward algorithms either present every frame for a very short duration or display only every  $n^{\text{th}}$  frame and skip the rest. This process requires several automated computer vision procedures to extract and represent the meaningful artifacts. Höferlin et al. in [6] propose an adaptive fast forward algorithm using temporal information of a video to gain between an estimated noise distribution and the absolute frame difference distribution by means of Rényi entropy. Another attempt to solve an adaptive video playback speed [9] is by using a visual attention model. The model is trained with different sets of features from different scenes. Besides, the evaluation of various fast-forward techniques is compared in [16]. It assesses the video visualization in term of object identification and motion perception, as well as the performance in the context of adaptive fast-forward.

#### 4.2 Situational Awareness

Due to the limited capacity of human perception, situational awareness becomes indispensable when the reaction of users is required within short time spans, particularly for real-time analysis such as monitoring and surveillance tasks. The most common perceptual deficits in video analysis are: (i) **Inattentive blindness**, which occurs when an observer’s attention is engaged in a task which causes the observer to fail noticing other obvious events. This occurrence is one of the major causes of

accidents and human error; (ii) **Change blindness** is a phenomenon in which dramatic changes occur in a visual scene right there but the changes are not noticed by the observer; and (iii) **Boredom and mental workload**. Boredom can be significant when people become under-stimulated or underload as suggested in [23]. Some endeavors to solve these issues in situational awareness are, e.g. [5, 4] use an auditory display to reduce workload off control room operators based on the multiple resource theory [28]. Höferlin et al. suggest that sonification is one of the key factors for efficient video surveillance [4], in correspondence to a study reported in [3], which observes the behavior of operators in a control room. Hence, situational awareness can benefit from sonification such that it allows users to listen to the events while their gazes are off the screen.

#### 4.3 Presentation and Reasoning

A visualization technique called “VideoPerpetuoGram” or VPG [1] is employed in the system to snap the sequences of video stream for detected objects with continuous illustrations. Höferlin et al. [13, 15] demonstrate the use of this method in IEEE VAST Challenge 2009. The challenge was to identify a suspicious person who was an embassy employee and transferred secret data to an outsider within 10 hours of surveillance footage. The VPG yields lots of trajectories which required user to filter the most irrelevant according to user’s hypotheses. The trajectories belonging to cars driving on the streets are filtered out, because the hypothesis was that a meeting of people requires the split and merge of trajectories. Another constraint was that the meeting would not take place on a street. Such analysis demands a vast amount of knowledge and is difficult for a machine to learn. Nevertheless, computer vision and machine learning



ning can answer the questions such as “Where are the streets?” and “Which trajectories are likely to be the meeting?”. To define and refine the hypotheses that lead to the solution depends on the domain specialist [15].

## 5 Visual Analysis of Gaze Data

A visual analytics approach to analyze eye-tracking data from multiple participants watching videos is presented by Kurzahls et al. in [20, 19]. This application (see Figure 2) employs multiple coordinated views, to obtain an understanding of general viewing behavior as well as individual differences between viewers. It can be used, e.g. to analyze whether attention of viewers watching commercials is on the intended objects, or distracted by confounding factors.

Applying a visualization technique called space-time cube (STC), the gaze data of the the whole video can be displayed in a static, spatiotemporal 3D visualization. Sparse gaze points can be filtered out, to highlight time spans of attentional synchrony. In these time spans, the attention of all viewers was attracted by a special object or event that should be investigated in detail. According to the Figure 1, this approach can be classified as extraction of meaningful artifacts. Additionally, a spatiotemporal clustering of the gaze points provides important information where important AOIs appear in the video. Furthermore, heat maps and scan paths which are the standard eye-tracking visualization techniques are also integrated into the application. To compare scan paths of multiple viewers, a timeline visualization can be exploited. In combination with different similarity measures for a hierarchical clustering of the viewers, groups of similar viewing behavior (Figure 2 Top-Right) and outliers (Figure 2 Bottom-Right) can be identified as situational understanding.

With the presented visual analytics approach, the combined analysis of video with additional data sources such as eye tracking data has been demonstrated. Although an extended analysis with other data sources, such as physiological data would increase the effort for an analysis, the concept of video visual analytics can still be applied.

## 6 Challenges

Based on our proposed architecture of video visual analytics, we identify open challenges in the following domains: (i) For eye tracking, the automatic identification and semantic annotation of AOIs are still challenging tasks that require further research. Especially the enrichment of AOIs with semantic information requires visual analytics, since automatic algorithms are not capable to process this task. (ii) Higher reasoning in video surveillance system. The application should extend methods for a situational understanding from graph-based

representation and offer possible conclusions. Besides, a model generation to support annotation and information retrieval of the video should be provided. (iii) Generating an auditory display for complex scenes is very appealing to help support situational awareness in surveillance. and (iv) there are still a lot of opportunities for the new data presentation aspect for visual analytics.

## 7 Conclusion

We presented an overview of video visual analytics concept and how it can be applied to two common scenarios with different requirements to the data analysis. As a foundation for future work, we pointed out the open challenges based on the described architecture and two application scenarios of video visual analytics.

## 8 Acknowledgments

This work was funded by German Research Foundation (DFG) as part of the Priority Program “Scalable Visual Analytics” (SPP 1335).

### Literature

- [1] BOTCHEN, R. P., BACHTHALER, S., SCHICK, F., CHEN, M., MORI, G., WEISKOPF, D., AND ERTL, T. Action-based multifield video visualization. 885–899.
- [2] DUFFY, B., THIYAGALINGAM, J., WALTON, S., SMITH, D. J., TREFETHEN, A., KIRKMAN-BROWN, J. C., GAFFNEY, E. A., AND CHEN, M. Glyph-based video visualization for semen analysis. *IEEE TVCG 99*, Pre-Prints (2013), 1.
- [3] GILL, M., SPRIGGS, A., ALLEN, J. AND JESSIMAN, P., SWAIN, D. AND HEMMING, M. K. D., AND LITTLE, R. Room operation: findings from control room observations. *Home Office Scientific Development Branch Publication 17/05* (2005).
- [4] HÖFERLIN, B., HÖFERLIN, M., GOLOUBETS, B., HEIDEMANN, G., AND WEISKOPF, D. Auditory support for situation awareness in video surveillance. In *In Proceedings of the ICAD* (2012).
- [5] HÖFERLIN, B., HÖFERLIN, M., RASCHKE, M., HEIDEMANN, G., AND WEISKOPF, D. Interactive auditory display to support situational awareness in video surveillance. In *In Proceedings of the ICAD* (2011).
- [6] HÖFERLIN, B., HÖFERLIN, M., WEISKOPF, D., AND HEIDEMANN, G. Information-based adaptive fast-forward for visual surveillance. *Multimedia Tools and Applications 55*, 1 (2011), 127–150.
- [7] HÖFERLIN, B., HÖFERLIN, M., WEISKOPF, D., AND HEIDEMANN, G. Scalable video visual analytics. *Information Visualization Journal* (2013).
- [8] HÖFERLIN, B., NETZEL, R., HÖFERLIN, M., WEISKOPF, D., AND HEIDEMANN, G. Inter-active learning of ad-hoc classifiers for video visual analytics. In *IEEE Conference on VAST 2012* (2012), pp. 23–32.
- [9] HÖFERLIN, B., PFLÜGER, H., HÖFERLIN, M., HEIDEMANN, G., AND WEISKOPF, D. Learning a visual attention model for adaptive fast-forward in video surveillance. In *In Proceedings of ICPRAM* (2012), vol. 2012, pp. 25–32.
- [10] HÖFERLIN, B. C. *Scalable Visual Analytics in Video Surveillance*. PhD thesis, Universität Stuttgart, 2013.

- [11] HÖFERLIN, M., GRUNDY, E., BORGIO, R., WEISKOPF, D., CHEN, M., GRIFFITHS, I. W., AND GRIFFITHS, W. Video visualization for snooker skill training. *Computer Graphics Forum* 29, 3 (2010), 1053–1062.
- [12] HÖFERLIN, M., HÖFERLIN, B., HEIDEMANN, G., AND WEISKOPF, D. Interactive schematic summaries for faceted exploration of surveillance video. *IEEE Transactions on Multimedia* 4 (2013), 908–920. [Project Phase 2] Scalable Visual Analytics of Video Data.
- [13] HÖFERLIN, M., HÖFERLIN, B., AND WEISKOPF, D. Video visual analytics of tracked moving objects. In *Proceedings of 3rd Workshop on Behaviour Monitoring and Interpretation* (2009), vol. 541, CEUR Workshop Proceedings, pp. 59–64.
- [14] HÖFERLIN, M., HÖFERLIN, B., WEISKOPF, D., AND HEIDEMANN, G. Interactive schematic summaries for exploration of surveillance video. In *Proceedings of the ACM ICMR '11* (2011), ACM, pp. 9:1–9:8.
- [15] HÖFERLIN, M., HÖFERLIN, B., WEISKOPF, D., AND HEIDEMANN, G. Uncertainty-aware video visual analytics of tracked moving objects. *J. Spatial Information Science* 2, 1 (2011), 87–117.
- [16] HÖFERLIN, M., KURZHALS, K., HÖFERLIN, B., HEIDEMANN, G., AND WEISKOPF, D. Evaluation of fast-forward video visualization. *IEEE TVCG* 18, 12 (2012), 20952103.
- [17] HÖFERLIN, M. J. *Video Visual Analytics*. PhD thesis, Universität Stuttgart, 2013.
- [18] KEIM, D. A., MANSMANN, F., SCHNEIDEWIND, J., THOMAS, J., AND ZIEGLER, H. Visual analytics: Scope and challenges. In *Visual Data Mining*. Springer, 2008, pp. 76–90.
- [19] KURZHALS, K., HEIMERL, F., AND WEISKOPF, D. Isee-cube - visual analysis of gaze data for video. In *Proceedings of the 2014 Symposium on ETRA* (2014), ACM, Ed., pp. 43–50.
- [20] KURZHALS, K., AND WEISKOPF, D. Space-time visual analytics of eye-tracking data for dynamic stimuli. *IEEE TVCG* 12, 19 (2013), 2129–2138.
- [21] LEGG, P. A., CHUNG, D. H. S., PARRY, M. L., JONES, M. W., LONG, R., GRIFFITHS, I. W., AND CHEN, M. Matchpad: Interactive glyph-based visualization for real-time sports performance analysis. *Computer Graphics Forum* 31 (2012), 1255–1264.
- [22] MEGHDADI, A. H., AND IRANI, P. Interactive exploration of surveillance video through action shot summarization and trajectory visualization. *IEEE Trans. Vis. Comput. Graph.* (2013), 2119–2128.
- [23] P. STAGER, D. H., AND JUBIS, R. Underlying factors in air traffic control incidents. In *in Proceedings of the Human Factors Society 33rd Annual Meeting, Santa Monica, CA* (1989), pp. 43–46.
- [24] PIROLI, P., AND CARD, S. The sensemaking process and leverage points for analyst technology as identified through cognitive task analysis. In *Proceedings of International Conference on Intelligence Analysis* (2005).
- [25] RUSSELL, D. M., STEFIK, M. J., PIROLI, P., AND CARD, S. K. The cost structure of sensemaking. *Proceedings of the SIGCHI conference on Human factors in computing systems - CHI '93* (1993).
- [26] THOMAS, J. J., AND COOK, K. A., Eds. *Illuminating the Path: The Research and Development Agenda for Visual Analytics*. IEEE Computer Soc., Los Alamitos, Calif., 2005.
- [27] THOMAS, J. J., AND COOK, K. A. A visual analytics agenda. *IEEE Computer Graphics and Applications* 26, 1 (2006), 10–13.
- [28] WICKENS, C. Multiple resources and performance prediction. *Theoretical Issues in Ergonomics Science* (2002).



**Pattreeya Tanisaro** received her bachelor of Electrical Engineering from Khon Kaen University, Thailand and earned her master degree in Computational Science from Goethe-University Frankfurt. Since 2013, she has been doing her Ph.D at the Institute of Cognitive Science, Osnabrück University.

E-Mail: pattanisar@uni-osnabrueck.de



**Julius Schöning** Julius Schöning received his master of science in intelligent embedded Microsystems from the University of Freiburg in 2013. In parallel he work since 2009 as software requirement engineer for HMI-Systems at the company CLAAS in Harsewinkel. Since 2014 he joined the Institute of Cognitive Science at the University of Osnabrück.

E-Mail: juschoening@uni-osnabrueck.de



**Kuno Kurzhals** Kurzhals Kuno received the Diplom degree in Computer Science at the University of Stuttgart, Germany in 2012. As a research assistant he is working at the Visualization Research Center (VISUS) at the University of Stuttgart. His main research interests are eye tracking, visualization, visual analytics, and computer vision. A specific focus of his research is on developing new visualization methods to analyze eye movement data from dynamic stimuli.

E-Mail: kuno.kurzhals@vis.uni-stuttgart.de



**Prof. Dr. Gunther Heidemann** Gunther Heidemann received the Diploma degree in physics from the University of Karlsruhe, and the doctorate degree in computer science from the University of Bielefeld, Germany. He held positions as assistant professor in computer engineering at the Florida State University and the Florida A&M University in Tallahassee, and as professor of computer science at the University of Stuttgart, Germany. Currently he is a professor of cognitive science at the University of Osnabrück, Germany.

Address: University of Osnabrück, Institute of Cognitive Science (IKW), D-49076 Osnabrück, E-Mail: gheidema@uni-osnabrueck.de



**Prof. Dr. Daniel Weiskopf** Daniel Weiskopf received the Diplom (MSc) degree in physics and the Dr. rer. nat. (PhD) degree in physics, both from Eberhard-Karls-Universität Tübingen, Germany, and he received the Habilitation degree in computer science at the University of Stuttgart, Germany. Since 2007, he has been a professor of computer science at the Visualization Research Center (VISUS) and the Visualization and Interactive Systems Institute (VIS) at the University of Stuttgart.

Address: University of Stuttgart, Visualization Research Center (VISUS), D-70569 Stuttgart, E-Mail: weiskopf@visus.uni-stuttgart.de

## 2.2 Time Series Classification Using Time Warping Invariant Echo State Networks

**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.

**Originally published in:** The 15th IEEE International Conference on Machine Learning and Application (ICMLA), 2016. pages: 831-836, IEEE

**DOI:** <http://dx.doi.org/10.1109/ICMLA.2016.0149>

30 2.2 • Time Series Classification Using Time Warping Invariant Echo State Networks

# 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

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

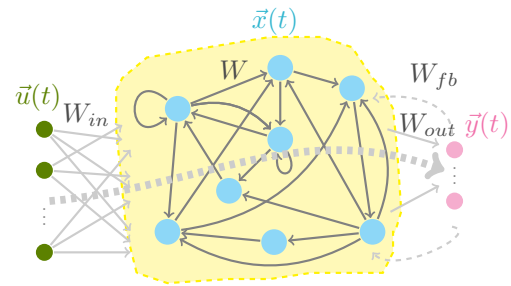


Fig. 1. Architecture of an ESN. The dashed lines denote the connections which are not compulsory.

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 computationally 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.

### 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  $t$ ,  $t = 1, \dots, T$ , by

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

$f(\cdot)$  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

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

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

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

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

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

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

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

where  $y_k(t)$  is the corresponding output of class  $k$ , and  $\tau$  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*, *OSULeaf*, *DistPhalanxAge*,

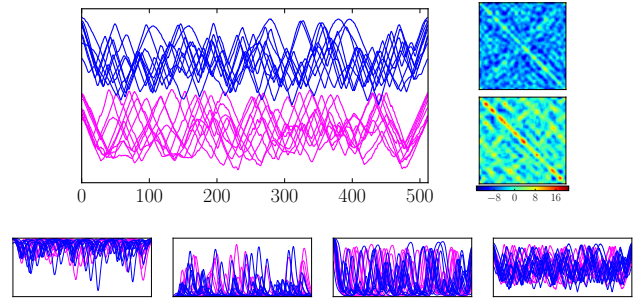


Fig. 2. The *BeetleFly* training set, which was obtained from shapelet transformations, consists of 10 samples of *Beetle* and *Fly* each. (Top left) The values on the x-axis show time, the values on the y-axis are shifted so that we are able to see the characteristic of each data class by different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

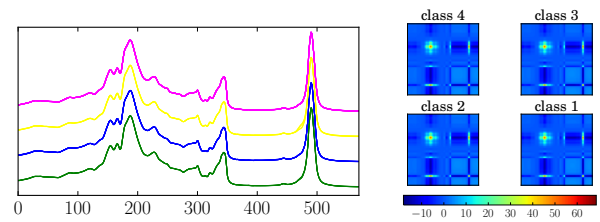


Fig. 3. (Left) Plot of the *OliveOil* training datasets of 30 samples from 4 different countries. (Right) The corresponding covariance matrices show no significant difference from 4 classes and this task is more difficult to solve than *BeetleFly*.

*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.

### 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/Scale $W_{in}/\lambda$	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	<b>0.100</b>	<b>0.100</b>	<b>0.100</b>	0.150	<b>0.100</b>	0.150	0.350	0.150	<b>0.100</b>	0.200	0.25
DiatomSizeR	0.078	0.078	0.082	<b>0.284</b>	0.275	0.219	<b>0.065</b>	<b>0.284</b>	0.268	0.206	<b>0.062</b>	<b>0.271</b>	0.255	0.206	<b>0.098</b>
GunPoint	0.033	0.020	0.033	0.100	0.033	<b>0.020</b>	0.033	0.053	0.027	<b>0.007</b>	0.033	0.047	<b>0.020</b>	0.027	<b>0.020</b>
OliveOil	0.433	0.233	0.167	<b>0.600</b>	<b>0.600</b>	<b>0.600</b>	<b>0.233</b>	<b>0.600</b>	<b>0.600</b>	0.567	<b>0.100</b>	<b>0.600</b>	<b>0.600</b>	0.567	<b>0.100</b>

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 ( $\alpha$ ):** 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

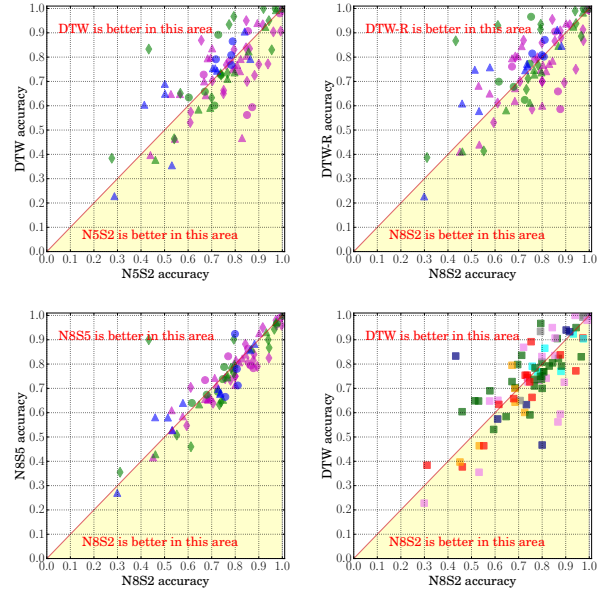


Fig. 4. Pairwise accuracy plots between two classifiers taken from the error rates in Table II. Each point represents a dataset. The colors in the top-left, top-right and bottom-left indicate the numbers of classes ( $n_C$ ): *magenta* for number of classes less than or equal 3, *green* for number of classes in  $[4..10]$  and *blue* for number of classes greater than 10. Different shapes indicate a high amount of training data: *diamond*( $\blacklozenge$ ) for training data less than or equal 100, *triangle*( $\blacktriangle$ ) for training data in  $(100..500]$ , and *circle*( $\bullet$ ) 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 ( $\blacksquare$ ) to signify the data type

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].

### 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
Lighting-2	2	60	0.246	0.131	0.131	0.344	0.279	0.295	0.246
Lighting-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
RefrigeratrDev	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
SonyAIBOSurfIII	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

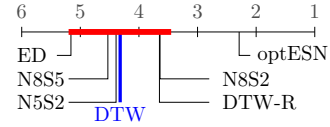


Fig. 5. Graphical representation of the Bonferroni-Dunn post-hoc test. The red line shows the two-tailedness of the critical difference when using *DTW* as a control.

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 raises some issues e.g., the 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 worsens 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 motifs 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.

## 2.3 Quasi View-Independent Human Motion Recognition in Subspaces

**Abstract:** We demonstrate the efficiency of a proposed feature extraction technique for dimensionality reduction in classifying human motions using MoCap data. We follow a technique introduced by Körner and Denzler for designing feature vectors before and after a transformation of these features into a subspace. This approach uses an average of a skeleton in a video in combination with a specific normalization to form an action descriptor. The major advantage of this approach is that it yields a very small fixed data size regardless of video length. This leads to a very fast computation with any classifiers.

In our experiment, we show the robustness of applying this feature extraction technique for various dimensionality reduction approaches under two conditions: i) untrained camera angles in combination with untrained subjects, ii) data loss; by further extension of i) by subsampling the original test data. Furthermore, we suggest an anatomically in-frame normalization which can improve the classification performance in several scenarios.

**Originally published in:** In Proceedings of 9th International Conference on Machine Learning and Computing (ICMLC) 2017. pages: 278–283, ACM

**DOI:** <http://dx.doi.org/10.1145/3055635.3056577>



# Quasi View-Independent Human Motion Recognition in Subspaces

Pattreeya Tanisaro  
Institute of Cognitive Science  
University of Osnabrück

Florian Mahner  
Institute of Cognitive Science  
University of Osnabrück

Gunther Heidemann  
Institute of Cognitive Science  
University of Osnabrück

## ABSTRACT

We demonstrate the efficiency of a proposed feature extraction technique for dimensionality reduction in classifying human motions using MoCap data. We follow a technique introduced by Körner and Denzler [18] for designing feature vectors before and after a transformation of these features into a subspace. This approach uses an average of a skeleton in a video in combination with a specific normalization to form an action descriptor. The major advantage of this approach is that it yields a very small fixed data size regardless of video length. This leads to a very fast computation with any classifiers.

In our experiment, we show the robustness of applying this feature extraction technique for various dimensionality reduction approaches under two conditions: i) untrained camera angles in combination with untrained subjects, ii) data loss; by further extension of i) by subsampling the original test data. Furthermore, we suggest an anatomically in-frame normalization which can improve the classification performance in several scenarios.

## CCS Concepts

•Computing methodologies → Activity recognition and understanding; Motion capture; Dimensionality reduction and manifold learning;

## Keywords

Human Motion Recognition, Feature Extraction, Subspaces, Dimensionality Reduction

## 1. INTRODUCTION

Human motion recognition by computer vision has a broad field of applications, ranging, e.g., from safety applications [13, 14, 29] to studies on human behavior in psychology and cognitive science [21, 19, 31, 22]. Over the years, storage systems have become much cheaper, which resulted in an ever

increasing a quality of video footage. Nevertheless, monitoring or anomaly detection in surveillance can still not be automatized due to the high complexity of video data in a spatial and temporal domain. Though it is possible to detect and track persons in video and interpret behavior to a limited degree, such techniques have not yet sufficient reliability for the market — there are too many false alarms. This is because a semantic understanding of human action is a difficult task, especially when viewpoint variation is involved. This leads to numerous human action recognition studies based on MoCap data. Unfortunately, many of these studies perform classification using one default view (Camera is still and no panning and tilting considered, while subjects move in only one direction) [23, 20, 18, 11], whereas in real life situations the subject can move from any direction toward or away from the camera (view independence).

Another important factor for real-time application is speed. As a consequence, dimensionality reduction, i.e., the encoding of high dimensional data in a space of lower dimension, has well-known benefits: reduced computational cost and memory requirements as well as improved generalization. Furthermore, dimensionality reduction can provide an interpretation of given data by means of visualization. Numerous manifold learning methods such as PCA, Kernel PCA (K-PCA) [27], Local Linear Embedding (LLE) [26] and Isomap [30] have been widely used for dimensionality reduction, nonetheless, they are not designed to perform directly on time series data. Generally, time series classification task involve two aspects [25]: i) data representation methods to reduce dimensionality or to transform data into another feature space and ii) time series distance measures. A large number of approaches [16, 4, 32, 2, 25] have been introduced to solve such time series classification problems and were benchmarked using the UCR archive. The UCR archive [7] is a collection of time series datasets from a variety of application domains, and the entire datasets in the archive are in one dimension. Encapsulating multidimensional data to such a low-dimensional subspace has also a disadvantage, that is the classification is much more difficult to solve [25].

Our objective is to study the efficiency and robustness of a proposed feature extraction technique under rotational variance of subjects which is equivalent to camera panning. The study is conducted using 3D and 2D MoCap data from the CMU MoCap dataset [8]. The 2D data is produced by projecting the 3D onto a 2D plane and transformed to screen coordinates simulating a 2D point-light video. Such 2D point-light motion has been extensively studied in the biological

motion perception in psychology and cognitive science [31, 21, 31, 19]. Additionally, we also present the result of classifying multidimensional data using a naive approach versus employing a subspace projection using various classification approaches. Several manifold learning techniques are exploited in combination with diverse classification methods to achieve the best performance.

## 2. RELATED WORK

Most of the work on human motion recognition follows a popular approach in computer vision called a single-layer approach [1], in which the recognition is performed directly on video data for simple and short sequential movements of humans. We also carry out our experiment and study of other related work in this manner.

One of the most promising methods for view-independent recognition is suggested by [15]. It employs self-similarity matrices [28] which are similar to the unthresholded recurrence plot (RP) [10]. RP is a method used to visualize high dimensional phase space trajectories. The matrix elements correspond to the points in times at which a state of a dynamical system recurs. This matrix is a measure of similarity as a sum of squared differences. Therefore, it is used to represent each action as a key feature for action descriptors. The benefit of using this technique is that the descriptors are stable across view changes. The recognition relies on the Bag-of-Features (BoF) obtaining from the Histogram of Oriented Gradient (HOG) describing their geometric properties. Nonetheless, the downside of this approach is that the sequences of all motions in the experiment are cut to have an equal unit length in order to get the fixed window size for the recognition. Other works that use dimensionality reduction techniques for human motion recognition are, for instance, the approach in [5] that presents human motion recognition from 3D motion data using PCA. The works of [24] and [9] apply PCA to recognize human activities to 2D videos.

Our main contributions in this work are i) we demonstrate the robustness of the proposed feature extraction technique in combination with various dimensionality reduction methods using untrained angles for unknown subjects. In addition, we show that this technique is also robust against data loss. ii) we compare different dimensionality reduction techniques with popular classification methods. iii) we show that a local anatomical normalization using an in-frame reference can lead to an improved recognition of raw data.

## 3. FEATURE VECTORS

In this section, we discuss two ways to obtain feature vectors for classification: i) using a direct naive approach in order to get features in space frame by frame as they are, and ii) using the proposed method to get features in a subspace.

### 3.1 Feature Vectors as Naive Approach

For each video sequence, we compute the feature vectors  $V_m \in \mathbb{R}^{N_{f,m}} \times \mathbb{R}^{(N_j \cdot N_d)}$ , where  $N_{f,m}$  is a number of frames of video  $m$ , and  $m$  is the video index in  $\{1..M\}$ . Let  $M$  be the number of training videos, and  $N_j$  the number of skeleton joints. Here, we merge the MoCap markers to obtain fixed  $N_j = 15$  joints.  $N_d$  is the number of the dimension in  $\{2, 3\}$  for 2D and 3D projection, respectively. To feed data

to a classifier, all  $M$  videos have each frame stacked on top of one another as shown in Figure 1. The classifier can process a total number of frames  $N_M$  as  $N_M = \sum_{m=1}^M N_{f,m}$ .

For testing, the video with index  $m$  is used to obtain pre-

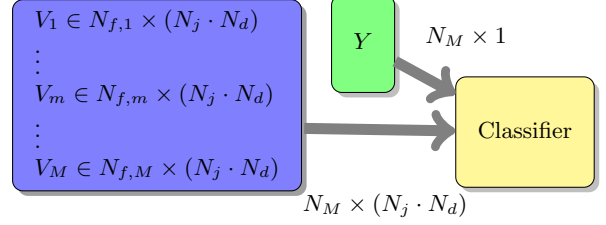


Figure 1: Feature vectors using the naive approach

dicted output  $y_m \in \mathbb{R}^{N_{f,m}}$  (where  $m$  is an arbitrary index of video either in the training or test dataset). To choose the motion class  $c^*$  of the video with index  $m$ , we use the majority voting of  $y_m$  for all frames  $N_{f,m}$  in that video.

$$c^* = \max_{c \in \{1..C\}} \sum_{i=1}^{N_{f,m}} y_{i,c} \quad (1)$$

### 3.2 Feature Vectors in Subspaces

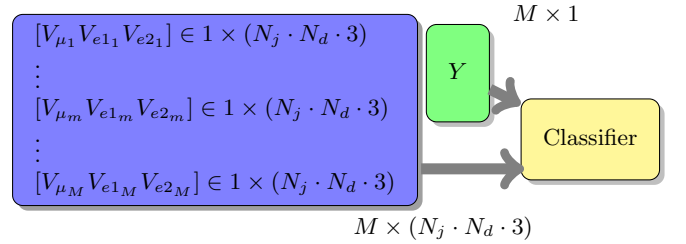
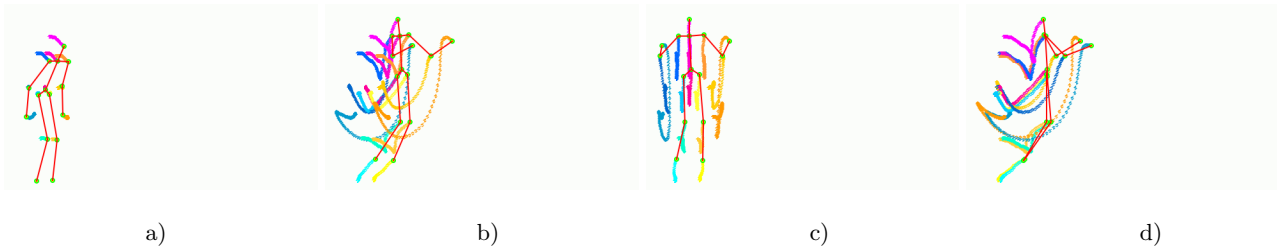


Figure 2: Feature vectors when two basis vectors ( $N_c = 2$ ) are selected. The number of training data is very small compared to the naive approach and is equal to the number of training videos.

To build feature vectors using dimensionality reduction, we follow the approach suggested by Körner and Denzler [18]. This technique normalizes data points in an anatomically fashion by computing a zero-mean skeleton configuration at each frame. For each video  $m$ , we compute the mean average of the joints from all frames for each dimension  $d$  of  $L_{m,d}$  as:

$$L_{m,d} = \frac{1}{N_{f,m}} \sum_{i=1}^{N_{f,m}} \ell_i \quad (2)$$

where  $\ell_i = ((x_1, y_1), \dots, (x_{N_j}, y_{N_j}))_i$  is a joint matrix at frame  $i$  for 2D projection ( $N_d = 2$ ),  $d \in \{x, y\}$ . Thus, for video  $m$ , we have the mean average of the joints  $L_m \in \mathbb{R}^{(N_j \cdot N_d)}$ . Next, we normalize the data in order to compute the eigenvectors. The normalized vector of the video  $m$  is defined as  $V_{\mu,m} \in \mathbb{R}^{N_{f,m}} \times \mathbb{R}^{(N_j \cdot N_d)}$  where the dimensions are still maintained at this step. For each dimension  $d$  of  $V_{\mu,m}$ ,  $V_{\mu,m,d}$  is computed using:



**Figure 3: Tracking of each skeletal joint locations for  $N_j = 15$  on 2D projection. A subject performs *jumping forward* from different angles: a)  $-45^\circ$  at frame 150 (The subject stands still for the first 100 frames) b)  $-45^\circ$  at frame 250 c)  $0^\circ$  at frame 250 d)  $-90^\circ$  at frame 250.**

$$V_{\mu,m,d} = V_{m,d} - L_{m,d} \quad (3)$$

$L_m$  is applied frame-wise. For the number of chosen principle components  $N_c$ , we obtain basis vectors  $V_{e_c,m}$  from a manifold transformation of  $V_{\mu,m}$ . Each  $V_{e_c,m}$  has size of  $\mathbb{R}^{(N_j \cdot N_d)} \times \mathbb{R}^{N_c}$ . Therefore, the feature vectors of video  $m$  can be obtained from the mean average vector and the normalized basis vectors as  $[L_m; V_{e_c,m}]$  with the shape of  $N_j \cdot N_d \cdot (N_c + 1)$  as shown in Figure 2.  $V_{\mu,m}$  can be transformed using various dimensionality reduction techniques to get the best learning performance. The results of applying different manifold learning methods are illustrated in section 4.3 and 4.4.

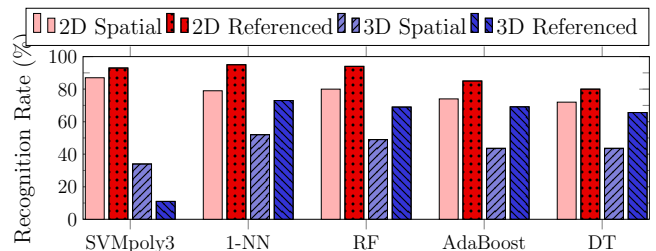
Extracting feature vectors using the proposed dimensionality reduction has the benefit that the extracted features are very small no matter how long the videos are, since the number  $M$  of videos is much less than the number of total frames of all videos  $N_M$  ( $M \ll N_M$ ).

## 4. EXPERIMENTAL SETUP AND RESULT

### 4.1 Experimental Setup

We choose ten actions from the CMU MoCap dataset which are *bending* (subjects bend to pick up objects either with both hands or with one hand from the ground and sometimes put them over their head), *boxing*, *golf swing*, *jumping forward*, *marching*, *running*, *salsa dance*, *cross-crunch* exercise (written shortly as *crunch*), *side-twist* exercise (or *twist*) and *walking*. The number of markers on MoCap are reduced to 15 representing the joints of a skeleton. Each training and test set consist of five videos. However, some actions such as *side-twist* and *cross-crunch* are limited to two subjects performing only a few trials for a long period of time. Therefore, we cut these long videos in order to obtain ten short videos for the training and test set. Subjects in the training set are excluded from the test set. The exception is *salsa dance* that has only one subject performing on all ten trials. For the training set, we apply five camera angles  $\{-90, -45, 0, 45, 90\}$  to each video. A sample of these videos is shown in Figure 3. For the test set, we use twenty-one angles in  $\{-100, -90, \dots, 90, 100\}$ . Therefore, we have  $10 \times 5 \times 5$  videos ( $M = 250, N_M = 122645$ ) for training data, and  $10 \times 5 \times 21$  videos ( $M = 1050, N_M = 570759$ ) for testing recognition of untrained subjects and, from untrained camera angles. In addition to unmodified *spatial* video data, we apply *local normalization* with respect to an *in-frame reference point* to the data. This means the spatial data of each frame has each joint subtracted from the

reference. Here, we pick the joint that laid at the center of the torso in a skeleton as a reference. It is important to note that we do not adopt the normalization with a zero-mean and standard deviation of one, because the assumption of normal distribution worsens the performance of sequential classification in our prior test.



**Figure 4: Recognition rate from 3D and 2D projection using the naive approach with five classifiers. The recognition rates from four classifiers are improved by using in-frame referenced data compared to raw spatial data**

### 4.2 Classifications using Naive Approach

In this section, we extract features using the naive approach discussed in section 3.1. Several classification methods with the same configurations will be referred to for the rest of our experiment, for instance, Support Vector Machine (SVM) with polynomial kernel degree 3 (SVMpoly3), k-Nearest Neighbor (k-NN) with  $k=1$ , Decision Tree (DT), AdaBoost with DT and Random Forests (RF) using Gini impurity with 50 trees. Two common approaches typically used for human motion recognition and general time-series classification are k-NN [18, 3, 6] and SVM [15]. Especially, k-NN gains favors in many types of research for time-series classification with distance measures [3] such as Euclidean Distance (ED), Dynamic Time Warping (DTW) [17] and Edit Distance on Real sequence (EDR) [6]. Nonetheless, our experimental results show that apart from k-NN and SVM polynomial (degrees 2,3,4,5 give only a few percent difference), RF is one of the best classifiers comparable to 1-NN as shown in Figure 4. Furthermore, the recognition rates in Figure 4 also depict the influence of using normalized data by in-frame reference. The average of using in-frame reference is about 10% better than using raw spatial data for most classifiers.

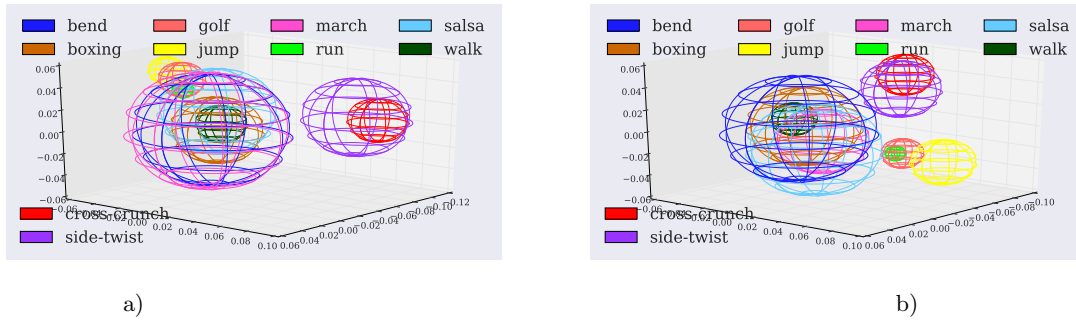


Figure 5: The largest areas of distribution of motions from the basis vector with the largest eigenvalue using PCA. a) Sphere distributions of 3D skeletons at 0° and b) at 90°.

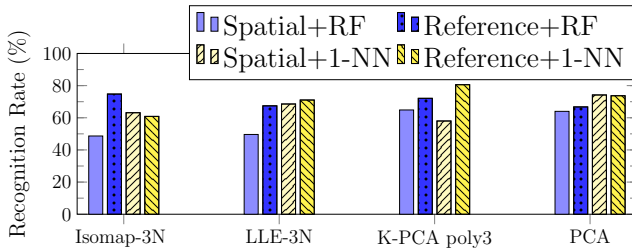


Figure 6: Recognition rates from 3D using dimensionality reduction from spatial vs. in-frame reference data. The results are from several manifold learning techniques using 3 principal components in combination with two classifiers, RF and 1-NN.

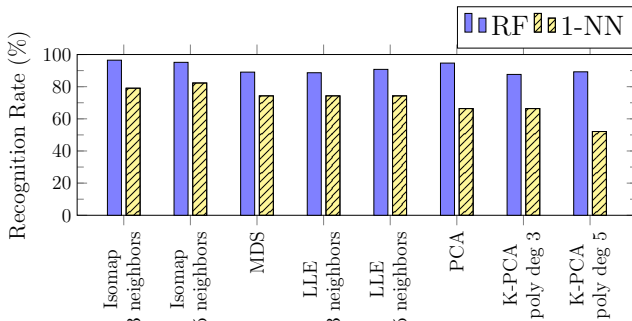


Figure 7: Recognition rates from 2D projection using dimensionality reduction from spatial data. The results are obtained from several manifold learning using 3 principal components in combination with two classifiers, RF and 1-NN.

### 4.3 Dimensionality Reduction

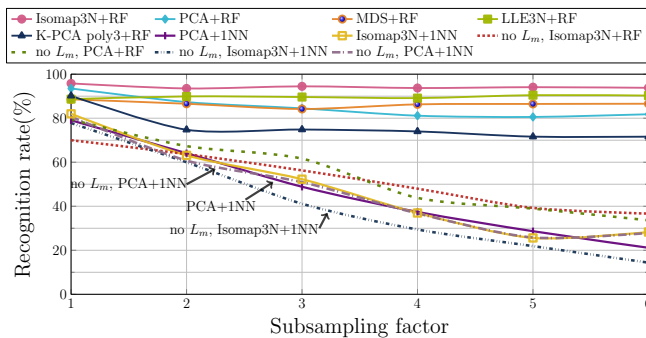
For subspace analysis, we choose several manifold learning algorithms using three principal components, for instance, PCA, Kernel PCA (K-PCA) with third order polynomial (In [18], order 5 and 9 are used to obtain the best performance), LLE with 3 neighbors (LLE-3N), Isomap with 3 neighbors (Isomap-3N) and metric Multidimensional scaling (MDS). Figure 5 shows the largest areas of distribution of ten mo-

tions from the basis vector with the largest eigenvalue using PCA. The rotation in 3D data after the transformation still maintains the largest area of distribution from all actions as depicted in Figure 5 a) and b). The first basis vector  $V_{e_1,m}$  is used to compute the sphere distribution. The dimensions in  $N_d$  are preserved in the basis vector and displayed as the axes in the figures. The radii of the spheres are calculated from the maximum distance of  $|V_{e_1,m} - V_{e,\mu}|$  in all dimensions, where  $V_{e,\mu}$  is an average of  $V_{e_1,m}$  of the corresponding dimension. The results of 3D data from several manifold learning methods using three basis vectors in combination with two classifiers RF and 1-NN, are shown in Figure 6. The in-frame reference data can improve the performance of 3D data but does not show any improvement for 2D data which is transformed to the screen coordinates. The outputs from applying similar configurations to 2D data are displayed in Figure 7. The best results from dimensionality reduction for 2D are obtained from spatial data employing Isomap with 3 and 5 neighbors getting 96.48%, 95.14%, and PCA 94.67% in combination with RF, respectively. The results of using subspaces yield even better performance than the naive approach. For testing, employing subspace has a speed-up of 1.5 for 2D data and takes 6.7 seconds for 1050 samples using a single process on Intel i7-3770 processor. In a case of applying 1-NN as a classifier, computing based on subspace for test data can obtain average speed 23 times faster than using the naive approach for both 2D and 3D data.

### 4.4 Data Loss

Further, we extend our experiment to demonstrate the stability with respect to data loss by subsampling the original test data using subsampling factors of 2,3,4,5 and 6; whereas the training data still remains the same. The subsampling factor of 2 will take only 50% of the original data, meaning that every 2 frames of the test data will be taken instead of each single frame (factor of 1). Hence, using subsampling factor of 6 as shown here leads to 83% data loss. Figure 8 illustrates the stability of applying dimensionality reduction when data is missing. The dashed and dotted lines show the results of feature extraction excluding the mean average of video  $L_m$  introduced in section 3.2. The results show that without mean average  $L_m$ , the classification is quite sensitive to data loss. In addition, our proposed feature extraction together with RF apparently gives stable outcome regardless of subsampling factors comparing to 1-





**Figure 8: Recognition rates from resampling 2D data with different sub-sampling factors using 3 basis vectors. The methods with “no  $L_m$ ” are the methods using feature vectors without the mean average of the videos in equation 2.**

NN. Isomap with 3 neighbors (Isomap-3N) in combination with RF presents the best performance, LLE and MDS also gain stable good performance. On the contrary, PCA, which is a favored dimensionality reduction approach, is pretty sensitive to data loss comparing to other dimensionality reduction methods.

## 4.5 Discussion

For 3D data, we get the best performance 80.6% using K-PCA and 1-NN and we did not try to optimize further, whereas [18] gains 94.34% for 8 actions with one view, [12] 98.29% for 3 actions with one view. Close to our experiment setup is [15], using 13 joints for 12 actions with fixing data at 164 frames for each motion on 6 camera angles, yields 90.5%. However, its testing angles are also similar to the training angles. None of these works has a constraint on the separation of test subjects from training subjects with untrained angles. Table 1 displays the confusion matrix from 2D data using our default setup for 21 angles having a recognition rate of 96.5%. The worst performance comes from *salsa dance* with 80%. This action is a complex action type which means one action is a combination of several untrained short movements. The misclassification of *salsa* as *boxing* might be because hands stretching out are captured during the long movement. (*Salsa* in the training set has the longest movement average about 1800 frames where the other 9 actions have an average in the range of 160-550 frames.) The other misclassified actions are from *marching* which is mistaken as *walking* 3.8% and *crossing-crunch* 2.8%; *running* is mistaken for *walking* 2.8%; and *walking* mistaken for *running* 3.8%. The most confusions of actions are between *running*, *marching* and *walking* which both legs, hands and the body joints have quite similar trajectories. Figure 9 (a)-(d) demonstrate two subjects performing two actions, *walking* and *running* from camera angles  $0^\circ$  and  $-45^\circ$ . The pictures show pretty close similarities of joint’s trajectories from two angles of two actions, especially from  $0^\circ$ . Hence it is not a trivial task to classify these two actions in subspaces for the untrained camera angles.

## 5. CONCLUSION

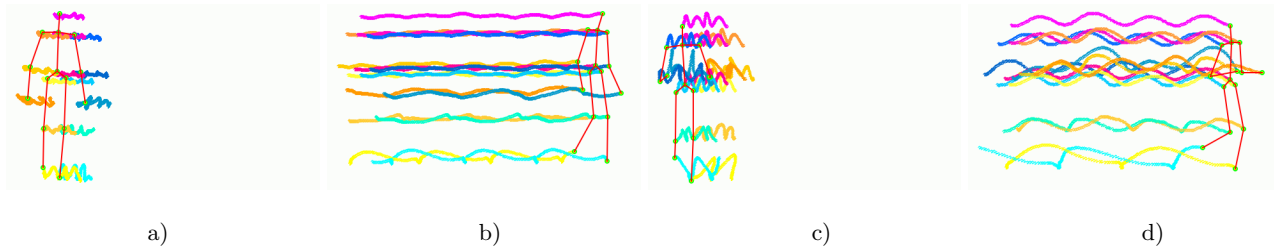
	bend	box	golf	jump	march	run	salsa	crunch	twist	walk
bend	100	0	0	0	0	0	0	0	0	0
box	0	100	0	0	0	0	0	0	0	0
golf	0	0	100	0	0	0	0	0	0	0
jump	0	0	0	100	0	0	0	0	0	0
march	0	0	0	0	93.4	0	0	2.8	0	3.8
run	0	0	0	0	0	96.2	0	0	0	3.8
salsa	0	20	0	0	0	0	80	0	0	0
crunch	0	0	0	0	0	0	0	100	0	0
twist	0	0	0	0	0	0	0	0	100	0
walk	0	2	0	0	0	2.8	0	0	0	95.2

**Table 1: Confusion matrix of classifying 1050 test samples of the 2D skeleton using *Isomap-3N+RF*.**

We demonstrated the robustness of using the proposed scheme of representing human motion sequences in subspaces. Our results show that the presented algorithm is quite stable against data loss and yields high recognition rates even with untrained subjects and untrained angles. The normalization with in-frame reference data can improve the recognition by more than 10% in most cases. We also show that the Isomap which performs MDS in the geodesic space of nonlinear data with RF yields the best result. Using PCA for the case of data in sequence loss is not suitable. Additionally, although 1-NN performs relative well but the testing is slower than RF with 50 trees and it is not robust against data loss. The proposed design features could be further extended for other applications of time series classification. Training and testing are fast. They can be used with classifiers that support on-line learning for real-time applications and to analyze multidimensional data for other time series data.

## 6. REFERENCES

- [1] J. Aggarwal and M. Ryoo. Human activity analysis: A review. *ACM Comput. Surv.*, 43(3):16:1–16:43, Apr. 2011.
- [2] 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*, 2016.
- [3] A. Bagnall, L. M. Davis, J. Hills, and J. Lines. Transformation based ensembles for time series classification. In *Proceedings of the Twelfth SIAM International Conference on Data Mining*, pages 307–318, 2012.
- [4] G. E. A. P. A. 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*.
- [5] A. Bottino, M. D. Simone, and A. Laurentini. Recognizing human motion using eigensequences. *Journal of WSCG*, 15(1-3):135–142, 2007.
- [6] L. Chen, M. T. Özsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*.
- [7] Y. Chen, E. Keogh, B. Hu, N. Begum, A. Bagnall, A. Mueen, and G. Batista. The UCR time series classification archive, 2015. [http://www.cs.ucr.edu/~eamonn/time\\_series\\_data/](http://www.cs.ucr.edu/~eamonn/time_series_data/).



**Figure 9: Two actions from two camera angles that often misclassified in subspaces. a) walking at  $0^\circ$  for 493 sequences. b) similar action as a) at  $-45^\circ$ . c) running at  $0^\circ$  for 189 sequences, and d) similar action as c) at  $-45^\circ$ .**

- [8] CMU Graphics Lab. CMU Motion Capture Database. <http://mocap.cs.cmu.edu/>.
- [9] A. Diaf, B. Boufama, and R. Benlamri. A compound eigenspace for recognizing directed human activities. In *Proceedings of the 9th International Conference on Image Analysis and Recognition - Volume Part II, ICIAR'12*, pages 122–129, 2012.
- [10] J. P. Eckmann, O. S. Kamphorst, and D. Ruelle. Recurrence plots of dynamical systems. *Europhysics Letters*, 4, Nov. 1987.
- [11] L. Han, W. Liang, X. Wu, and Y. Jia. Human action recognition using discriminative models in the learned hierarchical manifold space. In *8th IEEE International Conference on Automatic Face Gesture Recognition*, pages 1–6, 2008.
- [12] L. Han, X. Wu, W. Liang, G. Hou, and Y. Jia. Discriminative human action recognition in the learned hierarchical manifold space. *Image Vision Comput.*, 28(5):836–849, May 2010.
- [13] M. Höferlin, B. Höferlin, and D. Weiskopf. Video visual analytics of tracked moving objects. In *Proceedings of 3rd Workshop on Behaviour Monitoring and Interpretation*, 2009.
- [14] M. Höferlin, B. Höferlin, D. Weiskopf, and G. Heidemann. Uncertainty-aware video visual analytics of tracked moving objects. *J. Spatial Information Science*, 2(1):87–117, 2011.
- [15] I. N. Junejo, E. Dexter, I. Laptev, and P. Perez. View-independent action recognition from temporal self-similarities. *IEEE Trans. Pattern Anal. Mach. Intell.*, 33(1):172–185, Jan. 2011.
- [16] E. Keogh. Exact indexing of dynamic time warping. In *Proceedings of the 28th International Conference on Very Large Data Bases, VLDB '02*, pages 406–417. VLDB Endowment, 2002.
- [17] E. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. *Knowl. Inf. Syst.*, 7(3):358–386, Mar. 2005.
- [18] M. Körner and J. Denzler. Analyzing the subspaces obtained by dimensionality reduction for human action recognition from 3d data. In *IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, pages 130–135, 2012.
- [19] V. A. Kuhlmeier, N. F. Troje, and V. Lee. Young infants detect the direction of biological motion in point-light displays. *Infancy*, 15(1):83–93, 2010.
- [20] C. Li, P. R. Kulkarni, and B. Prabhakaran. Segmentation and recognition of motion capture data stream by classification. *Multimedia Tools Appl.*, 35(1):55–70, 2007.
- [21] M. Livne, L. Sigal, N. F. Troje, and D. J. Fleet. Human attributes from 3d pose tracking. *Computer Vision and Image Understanding (CVIU)*, 2012.
- [22] L. E. Miller and A. P. Saygin. Individual differences in the perception of biological motion: Links to social cognition and motor imagery. *Cognition*, 128(2):140–148, 2013.
- [23] M. Müller and T. Röder. Motion templates for automatic classification and retrieval of motion capture data. In *Eurographics Symposium on Computer Animation*, pages 137–146, 2006.
- [24] M. A. Nael, M. M. Abdelwahab, and M. El-Saban. Multi-view human action recognition system employing 2dpc. In *WACV*, pages 270–275. IEEE Computer Society, 2011.
- [25] P. Tanisaro and G. Heidemann. Time series classification using time warping invariant echo state networks. In *15th IEEE International Conference on Machine Learning and Applications, (ICMLA 2016)*.
- [26] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *SCIENCE*, 290:2323–2326, 2000.
- [27] B. Schölkopf, A. Smola, and K.-R. Müller. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10(5), July 1998.
- [28] E. Shechtman and M. Irani. Matching local self-similarities across images and videos. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2007.
- [29] P. Tanisaro, J. Schöning, K. Kurzhals, G. Heidemann, and D. Weiskopf. Visual analytics for video applications. *it-Information Technology*, 57:30–36, 2015.
- [30] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319, 2000.
- [31] N. F. Troje. Biological motion perception. *The senses: A comprehensive reference*, 2:231–238, 2008.
- [32] 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.*, 26(2):275–309, 2013.

## 2.4 Classifying Bio-Inspired Model in Point-Light Human Motion Using Echo State Network

**Abstract:** We introduce a feature extraction scheme from a biologically inspired model using receptive fields (RFs) to point-light human motion patterns to form an action descriptor. The Echo State Network (ESN) which also has a biological plausibility is chosen for classification. We demonstrate the efficiency and robustness of applying the proposed feature extraction technique with ESN by constraining the test data based on arbitrary untrained viewpoints, in combination with unseen subjects under the following conditions: i) lower sub-sampling frame rates to simulate data sequence loss, ii) remove key points to simulate occlusion, and iii) include untrained movements such as drunkard's walk.

**Originally published in:** The 26th International Conference on Artificial Neural Networks (ICANN), 2017. Springer International Publishing.

**DOI:** [http://dx.doi.org/10.1007/978-3-319-68600-4\\_11](http://dx.doi.org/10.1007/978-3-319-68600-4_11)

46 2.4 • Classifying Bio-Inspired Model in Point-Light Human Motion Using Echo State Network

# Classifying Bio-Inspired Model of Point-Light Human Motion using Echo State Networks

Pattreeya Tanisaro, Constantin Lehman, Leon Sützelfeld,  
Gordon Pipa and Gunther Heidemann

Institute of Cognitive Science, Osnabrück University, Germany

**Abstract.** We introduce a feature extraction scheme from a biologically inspired model using receptive fields (RFs) to point-light human motion patterns to form an action descriptor. The Echo State Network (ESN) which also has a biological plausibility is chosen for classification.

We demonstrate the efficiency and robustness of applying the proposed feature extraction technique with ESN by constraining the test data based on arbitrary untrained viewpoints, in combination with unseen subjects under the following conditions: i) lower sub-sampling frame rates to simulate data sequence loss, ii) remove key points to simulate occlusion, and iii) include untrained movements such as *drunkard's walk*.

**Keywords:** Echo State Network, Motion Capture, Motion Recognition, Biological Motion Perception, Bio-Inspired Model

## 1 Introduction

Human motion recognition has a large variety of applications, for example, in safety and surveillance such as access control and congestion analysis, abnormal behavior detection [12], and in behavioral biometrics including gesture and posture recognition for human computer interaction (HCI) [11]. These applications employ different representations and recognition techniques; however, the representation and recognition methods are usually mutually reliant. As suggested in [7], the human motion representation can be categorized into two models: humanoid body model and humanoid image model. The humanoid body model uses structural representations from joint-positions of 2D or 3D points in space simulating point-light display. This point-light display can be seen as the model of a stick figure which can be used to estimate human body parts in the humanoid body model. The earliest application of using such point-light displays of motion patterns was introduced for studying biological motion recognition on human visual perception mechanisms [5]. The experiment reveals that the movement of 10-12 bright spots attached to human body parts is sufficient for humans to distinguish the actions. The point-light display of human motion has later been widely used for subsequent studies on human behavior in psychology and cognitive science because human motion also conveys information about emotions or mental states, personality traits and biological attributes [15], [8], [9].

Our objective in this study is to demonstrate the possibility of applying biologically inspired models for both feature representation and recognition of fuzzy human motion. This study was conducted using motion capture (MoCap) from the CMU MoCap database [1]. The 3D data was projected onto a 2D plane and transformed to screen coordinates simulating a 2D point-light video. Afterwards, the 2D coordinate-space of each video frame was enlarged or shrunk to fit inside a grid. This idea was inspired by human grid cells for the formation of environment maps in the hippocampus. The receptive fields (RFs) resembling wavelets in the retina and primary visual cortex (V1) are generated inside the grid. With these techniques, we combined the trajectory-based approach from a kinematic structure of 2D point-lights with a pattern-based approach. The proposed feature extraction technique was tested under new angles of new subjects.

## 2 Temporal Pattern Classification using an ESN

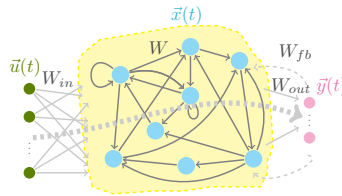


Fig. 1: Architecture of an ESN. The dashed lines denote the connections which are not compulsory.

An ESN [4] is a type of RNNs of which the weights are left untrained. Only the output weights are trained for the desired target at the readout connection where no cyclic dependencies are created. The work of [10] presents an ESN as a framework for neurodynamical models of working memory. It illustrates ESN properties for storing, maintaining, retrieving and removing data that are similar to functions of the brain. A general ESN architecture is shown in Figure 1.

Consider a discrete time neural network with input dimensionality  $N_u$ , neurons in the reservoir  $N_x$ , and output dimensionality  $N_y$ . Let  $\mathbf{u}(t) \in \mathbb{R}^{N_u}$ ,  $\mathbf{x}(t) \in \mathbb{R}^{N_x}$  and  $\mathbf{y}(t) \in \mathbb{R}^{N_y}$  denote the vectors of input activities, internal state and output unit activity for time  $t$  respectively. Further, let  $W_{in} \in \mathbb{R}^{N_x \times N_u}$ ,  $W \in \mathbb{R}^{N_x \times N_x}$  and  $W_{out} \in \mathbb{R}^{N_y \times N_x}$  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  $W_{fb} \in \mathbb{R}^{N_x \times N_y}$ . The internal unit activities  $\mathbf{x}$  in Figure 1 are updated from time step  $t - 1$  to time  $t$ , where  $t = 1, \dots, T$ , by

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

$f(\cdot)$  is an activation function of the neurons, a common choice is  $\tanh(\cdot)$  applied element-wise. The *leaky integration rate*  $\alpha \in (0, 1]$  is the leakage rate determining

the speed of the reservoir update dynamics. The update rule for the internal units is extended to

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

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

$$\mathbf{y}(t) = f_{out}(W_{out}[\mathbf{u}(t); \mathbf{x}(t)]), \quad (3)$$

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

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

The output class for testing the input sequences  $\mathbf{u}(t)$  is then computed by

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

where  $y_k(t)$  is the corresponding output of class  $k$ , and  $\tau$  is the length of time series of input  $u(t)$ .

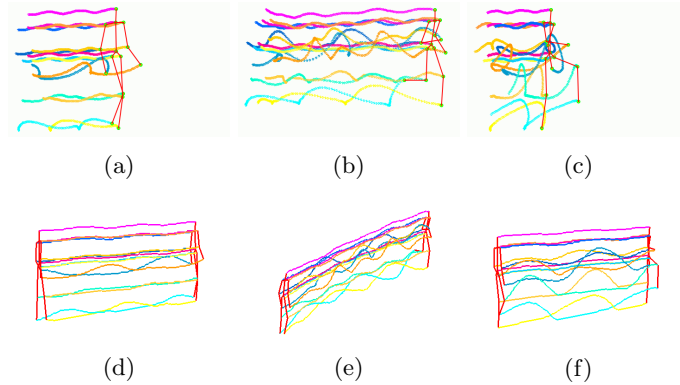


Fig. 2: Top: Three actions for 1.5 seconds (180 frames) at  $-45^\circ$ . *Walking*, *running* and *marching* are shown in a), b) and c), respectively. Bottom: The arbitrary views of corresponding trajectories of figures a), b) and c) are extended in time-scale in a three dimensional space shown in d), e) and f), accordingly.

### 3 Experimental Setup and Feature Representation

#### 3.1 Dataset

Nine actions ( $N_y = 9$ ) from the CMU MoCap database were chosen for the experiment. They were *bending* (i.e. subjects bend to pick up objects from the

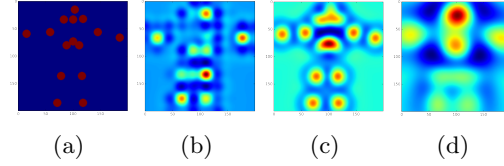


Fig. 3: Point-light figures with a diameter of 15 pixels at each joint. a) The point-light is stretched and filled in the grid of  $200 \times 200$  pixels. b) The grid is mapped on the RFs of size  $N_{RF} = 10 \times 10$  with a Marr wavelet of  $\sigma = 10$ . c)  $N_{RF} = 20 \times 20$ ,  $\sigma = 10$  and d)  $N_{RF} = 10 \times 10$ ,  $\sigma = 20$ , where there are overlaps of the RFs in the setting. This setting is based on a preferred bio-inspired model.

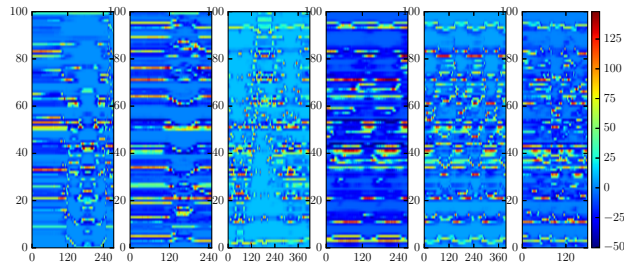


Fig. 4: The feature vectors  $\mathbf{u}(t)$  of six videos of different classes at  $0^\circ$  are shown from left to right: *golfing*, *bending*, *crunching*, *walking*, *marching* and *running*. The x-axis indicates the varying frame numbers of the video, whereas the y-axis has the fixed number of RFs.

ground and sometimes put them over their heads), *boxing*, *golfing swing*, *jumping forward*, *marching*, *running*, *standing cross-crunch exercise* (written shortly as *crunching*), *standing side-twist exercise* (or *twisting*), and *walking*. The markers on MoCap were reduced to 15 representing the joints of a skeleton. Each training and test set consists of five videos of different subjects. For some actions such as *twisting* and *crunching*, there are only a few subjects, but the videos are long; therefore, we cut these long videos in order to obtain ten short videos. It is important to note that subjects in the training set are excluded from the test set. For the training set, we apply five camera angles  $\{-90, -45, 0, 45, 90\}$  to each video. Three samples of these videos are shown in Figure 2. For the test set, we use twenty-one angles in  $\{-100, -90, \dots, 90, 100\}$ . Therefore, we have  $9 \times 5 \times 5$  videos for training data, and  $9 \times 5 \times 21$  videos for testing the recognition of new subjects and, from unseen camera angles.

### 3.2 Feature Representation

The 2D coordinates of each video frame were stretched to fit inside a  $200 \times 200$  pixels grid as shown in Figure 3 a). The grid has a fixed number of RFs producing



an input feature vector  $\mathbf{u}(t) \in \mathbb{R}^{N_{RF} \times N_F}$ , where  $N_{RF} = N_{RF_x} \times N_{RF_y}$  is the total number of RFs in a rectangular grid.  $N_F$  is the number of frames in a video. In our experiment, we chose  $N_{RF} = 10 \times 10$  and adjusted the  $\sigma$  of Marr wavelet in order to design the RFs in the way that the RFs overlapped one another as shown in Figure 3 b)-d). Examples of feature vectors of six videos representing different actions are displayed in Figure 4. The two leftmost figures are the *golfing* and *bending*, where there is no repetition of the action pattern. Next to them is the pattern of one and a half cycle of *crunching*. The last three images are *walking*, *marching* and *running* showing periodic patterns for about 2-3 cycles. The first 100 frames of the *golfing* and *bending* videos reveal very smooth patterns, indicating no significant movement of the agents in these two videos. This is typical for some actions such as *golfing*, *bending* and *jumping forward*. By contrast, actions such as *running*, *marching* and *walking* exhibit a very short onset of action and can complete one cycle in a very short time. In comparison, *running* is the shortest video with about 140 frames, while the other actions have an average in the range of 160-550 frames.

### 3.3 ESN Configurations

We set up a moderate reservoir size of  $N_x = 500$  having sparsely connected neurons with 10% connectivity similar to [13]. The weight matrices,  $W$  and  $W_{in}$ , are random values uniformly distributed in the range  $[-1, 1]$ . The spectral radius  $\rho(W)$  can be considered as the scaling factor of the weight matrix  $W$ . The desired spectral radius can be simply computed from the ratio of the desired value and the maximum of the absolute eigenvalues of weight matrix. For a long short-term memory network, [3] shows that the peak performance in the setup has the spectral radius set to one. The only parameter that would be varied in our experiment is the leaky rate ( $\alpha$ ), which can be regarded as a time warping of the input signal. All results in our experiment use the average of 4 runs of a randomly initialized ESN networks with the same configuration.

## 4 Experimental Results

### 4.1 Data sequence loss and redundancy as variations in speed

We subsampled the original test data using subsampling factors of 1,2,4,6,8, and 10, whereas the training data still remained the same. The subsampling factor of 2 means that every 2<sup>nd</sup> frame of the data will be taken instead of each single frame (factor of 1). We evaluated our result using ESN with three leaky rates  $\alpha = 0.1, 0.5$  and  $0.9$  comparing with two methods, 1-Nearest Neighbor (**1-NN**) and Random Forest (**RdF**) which use 15 joint-positions in videos obtained directly from MoCap **without RFs**. We used both a naive approach and a dimensionality reduction method to extract feature vectors for these two classifiers. The feature vectors for the naive approach are obtained by simply stacking all video frames on top of each other for training the classifiers. The voting majority of the frames in a target video is counted for the classification. For the dimensionality

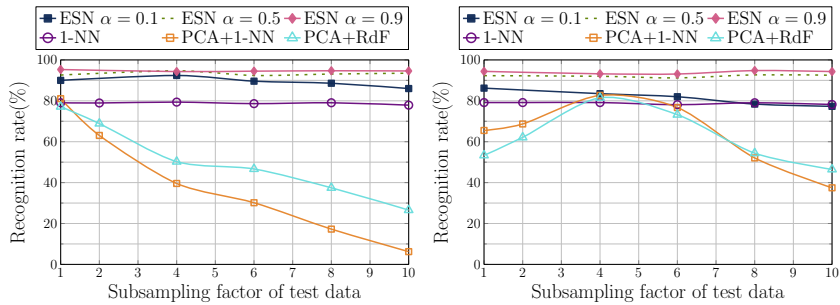


Fig. 5: Recognition rates from various classification approaches shown in y-axis. Subsampling factors of test data shown in x-axis. Left: Training factor of 1. Right: Training factor of 5.

reduction method, the feature vectors are acquired from PCA employing three principal components in combination with 1-NN and RdF. Figure 5-Left shows the results of recognition rate using test data from twenty-one untrained angles with respect to data sequence loss by using a training factor of 1. Figure 5-Right shows the result of recognition rate using a training factor of 5 as a testing for data redundancy. Both figures reveal that the ESN with  $\alpha = 0.9$  gives the best performance with robustness against data sequence loss and redundancy yielding a recognition rate of 95% even with large training and test subsampling factors. The good performance of ESN using  $\alpha = 0.9$  which can handle the variations in speed might be explained by the behavior of a long short-term memory in ESN which is demonstrated in [3]. Classifying data in space using 1-NN as the naive approach also gains a stable outcome of about 80%. In contrast, classifying data in subspaces using PCA is sensitive to data sequence loss and only gains good outcomes when the frequencies of training and test data sequences are about the same.

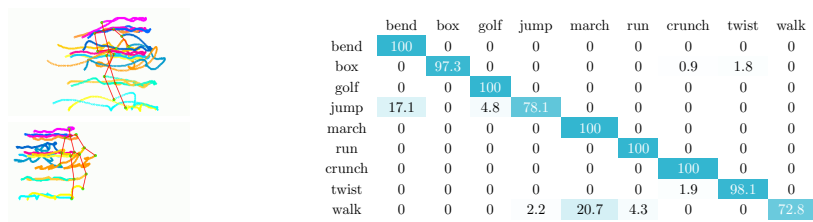


Fig. 6: Left: Two subjects perform *drunkard's walking*. Right: Confusion matrix of test samples by substituting *drunkard's walk* for *walk*.

## 4.2 Removing key points and drunkard's walks

We furthered the experiment by removing key points from the test data simulating occlusion from all frames in videos. Removing a *wrist* from a skeleton

does not affect the recognition rate, while removing an *ankle* from a skeleton makes *running*, *crunching* and *walking* all mistaken as *marching*, which results in a recognition rate drop to 61%. Furthermore, we extended the test by having three new persons performing four trials simulating *drunkard's walk* from twenty-two untrained angles producing 92 test samples for *walking*, while the other actions remain the same. Two samples of *drunkard's walk* are shown in Figure 6-Left. The confusion matrix in Figure 6-Right reveals that *walking* is misclassified as *marching* for 20.7%. The closed trajectories of these two actions can be inspected from Figure 2 c) and Figure 6-Left.

### 4.3 Discussion of Related Work

One of the earliest promising methods for view-independent recognition of 3D MoCap was introduced by [6]. It applied a non-threshold recurrent plot by computing a similarity matrix of each joint as a sum of squared differences. The benefit of using this method is that the descriptors are stable across view changes. The recognition relies on a Bag-of-Features obtaining from the Histogram of Oriented Gradient. However, the disadvantage of this approach is that the sequences of all motions in the experiment must have an equal length in order to get a fixed window size for recognition. Another study on view-independent recognition of Mocap is [14]. It proposed a feature extraction technique to transform either 2D or 3D data into subspaces to form an action descriptor. The major advantage of this approach is that it yielded a very small fixed data size regardless of video length, as well as very fast computation. The test on projected motion in 2D achieves a recognition rate of 96.5% from 21 untrained angles for 10 actions and it is also very stable for the case of data sequence loss. Other interesting skeleton based action recognition approaches for 3D MoCap are for instance, [2] and [16]. They proposed and compared several deep recurrent neural network architectures with Long Short-Term Memory (LSTM) for classification. The tests were carried out using 65 classes of HDM05 MoCap yielding up to 96.92% and 97.25% recognition rate respectively. Nonetheless, the tests were only performed for one default view.

## 5 Conclusion

We have introduced a feature extraction scheme from a biologically inspired model by applying the concept of receptive field to point light patterns of human motion. Our proposed scheme in combination with ESN which presents itself as a good approximator, yields a good performance and robustness against variations of speed even when the trajectories of motions are fuzzy. This representation could be deployed for human motion classification based on optical flow obtained from standard videos, where the human pose estimation is infeasible. The designed ESN is generic in the sense that it is not specialized to human motion. It also shows a good prediction of the unseen data. Hence, application to other domains of articulated objects in motion is possible. Furthermore, new

technologies such as the IBM TrueNorth chip have introduced a dedicated neuro-inspired hardware that allows modeling hundreds of thousands up to a million of neurons with very low energy. The ESNs, which offer very simple learning mechanisms, can be optimized by local learning rules that scale well even with very large systems. Therefore, ESN is a potential candidate for low energy systems that can be an integral part of sensor technology for the future.

## References

1. CMU Graphics Lab: CMU Motion Capture Database. <http://mocap.cs.cmu.edu/>
2. Du, Y., Wang, W., Wang, L.: Hierarchical recurrent neural network for skeleton based action recognition. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (June 2015)
3. Jaeger, H.: Long Short-Term Memory in Echo State Networks: Details of a Simulation Study. Tech. rep., Jacobs University, Bremen, Germany (Feb 2012)
4. Jaeger, H., Haas, H.: Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. *Science* 304(5667), 78–80 (2004)
5. Johansson, G.: Visual perception of biological motion and a model for its analysis. *Perception and Psychophysics* 14(2), 201–211 (1973)
6. Junejo, I.N., Dexter, E., Laptev, I., Perez, P.: View-independent action recognition from temporal self-similarities. *IEEE Trans. Pattern Anal. Mach. Intell.* 33(1), 172–185 (Jan 2011)
7. Kale, G.V., Patil, V.H.: A study of vision based human motion recognition and analysis. *IJACI* 7(2), 75–92 (2016)
8. Livne, M., Sigal, L., Troje, N.F., Fleet, D.J.: Human attributes from 3d pose tracking. *Computer Vision and Image Understanding (CVIU)* (2012)
9. Miller, L.E., Saygin, A.P.: Individual differences in the perception of biological motion: Links to social cognition and motor imagery. *Cognition* 128(2), 140 – 148 (2013)
10. Pascanu, R., Jaeger, H.: A neurodynamical model for working memory. *Neural Netw.* 24(2), 199–207 (2011)
11. Rautaray, S.S., Agrawal, A.: Vision based hand gesture recognition for human computer interaction: a survey. *Artificial Intelligence Review* 43(1), 71 (2015)
12. Tanisaro, P., Schöning, J., Kurzhals, K., Heidemann, G., Weiskopf, D.: Visual analytics for video applications. *it-Information Technology* 57, 30–36 (2015)
13. Tanisaro, P., Heidemann, G.: Time series classification using time warping invariant echo state networks. In: 15th IEEE International Conference on Machine Learning and Applications, (ICMLA) (2016)
14. Tanisaro, P., Mahner, F., Heidemann, G.: Quasi view-independent human motion recognition in subspaces. In: Proceedings of 9th International Conference on Machine Learning and Computing (ICMLC) (2017)
15. Troje, N.F., Sadr, J., Geyer, H., Nakayama, K.: Adaptation aftereffects in the perception of gender from biological motion. *Journal of Vision* 6(8), 7 (2006)
16. Zhu, W., Lan, C., Xing, J., Zeng, W., Li, Y., Shen, L., Xie, X.: Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence. pp. 3697–3703 (2016)

## 2.5 An Empirical Study on Bidirectional Recurrent Neural Networks for Human Motion Recognition

**Abstract:** The deep recurrent neural networks (RNNs) and their associated gated neurons, such as Long Short-Term Memory (LSTM) have demonstrated a continued and growing success rates with researches in various sequential data processing applications, especially when applied to speech recognition and language modeling. Despite this, amongst current researches there are limited studies on the deep RNNs architectures and their effects being applied to other application domains.

In this paper, we evaluated the different strategies available to construct bidirectional recurrent neural networks (BRNNs) applying Gated Recurrent Units (GRUs), as well as investigating a reservoir computing RNNs, i.e., Echo state networks (ESN) and a few other conventional machine learning techniques for skeleton based human motion recognition. The evaluation of tasks focuses on the generalization of different approaches by employing arbitrary untrained viewpoints, combined together with previously unseen subjects. Moreover, we extended the test by lowering the sub-sampling frame rates to examine the robustness of the algorithms being employed against the varying of movement speed.

**Originally published in:** The 25th International Symposium on Temporal Representation and Reasoning (TIME) 2018. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik GmbH, Wadern/Saarbruecken, Germany.

**DOI:** <http://dx.doi.org/10.4230/lipics.time.2018.21>

# An Empirical Study on Bidirectional Recurrent Neural Networks for Human Motion Recognition

**Pattreeya Tanisaro**<sup>1</sup>

Institute of Cognitive Science  
[University of Osnabrück, Germany]  
pattanisaro@uni-osnabrueck.de

**Gunther Heidemann**<sup>2</sup>

Institute of Cognitive Science  
[University of Osnabrück, Germany]  
gheidema@uni-osnabrueck.de

---

## Abstract

The deep recurrent neural networks (RNNs) and their associated gated neurons, such as Long Short-Term Memory (LSTM) have demonstrated a continued and growing success rates with researches in various sequential data processing applications, especially when applied to speech recognition and language modeling. Despite this, amongst current researches there are limited studies on the deep RNNs architectures and their effects being applied to other application domains.

In this paper, we evaluated the different strategies available to construct bidirectional recurrent neural networks (BRNNs) applying Gated Recurrent Units (GRUs), as well as investigating a reservoir computing RNNs, i.e., Echo state networks (ESN) and a few other conventional machine learning techniques for skeleton based human motion recognition. The evaluation of tasks focuses on the generalization of different approaches by employing arbitrary untrained viewpoints, combined together with previously unseen subjects. Moreover, we extended the test by lowering the sub-sampling frame rates to examine the robustness of the algorithms being employed against the varying of movement speed.

**2012 ACM Subject Classification** Mathematics of computing → Time series analysis, Computing methodologies → Supervised learning by classification, Computing methodologies → Activity recognition and understanding, Computing methodologies → Neural networks

**Keywords and phrases** Recurrent Neural Networks, Human Motion Classification, Echo State Networks, Motion Capture, Bidirectional Recurrent Neural Networks

**Digital Object Identifier** 10.4230/LIPIcs...21

## 1 Introduction

The recurrent neural networks, whose structures are similar to those of multilayer perceptrons (MLPs) have been widely used for sequential data processing. Nonetheless, they are different from the MLPs by allowing connections among hidden units, therefore the networks can retain information of past inputs as a vector of activation for each time step which makes RNNs exceedingly deep. Their depth, however, makes them difficult to train because the update of the weight matrices with a gradient-based approach such as Backpropagation Through Time (BPTT) leads to exploding and vanishing gradient problems [1]. Many techniques have been

<sup>1</sup>   
<sup>2</sup> 

introduced in order to solve these two issues, especially for the vanishing gradient problem, but training RNNs was still a very difficult task and the applications were limited.

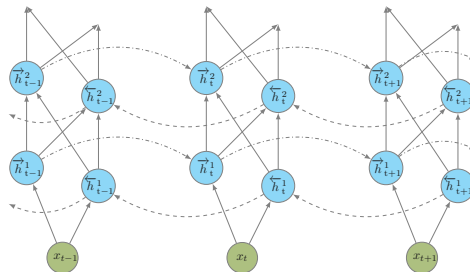
Since the emergence of special architecture for gradient-based methods called LSTM [17], training RNNs has become easier and more successful in numerous tasks such as speech recognition [12], acoustic modeling [24], sequence labeling in speech recognition [13], handwriting recognition [15], language modeling and machine translation [30, 31, 4], prediction of successful shooting in basketball [28], analyzing motion patterns in autonomous driving [11], image caption [36] and learning of video representation [29]. The LSTM is designed to solve the vanishing gradient problem whereas truncating the gradient is harmless to the networks because an LSTM can enforce a constant error flow within special units to bridge time lags. Unlike the traditional recurrent unit which calculates the weighted sum of inputs and directly applies the activation function, the LSTM unit contains a memory cell. Furthermore, there are several studies such as [26] and [18] that reported their achievement of an improvement of the output performance by introducing the depth to the RNNs. Typically, any RNNs when unfolded in time might be considered deep themselves, because the input to output in a given time span has crossed several nonlinear layers as computational paths [26]. Nonetheless, the *depth* of neural networks is usually defined by the number of feedforward neural layers. Most studies in deep RNNs concentrate on sequence-to-sequence modeling, particularly for language modeling for instance [30, 16, 18, 26, 6].

In our study, we focused on solving a classification problem using a special type of deep RNNs, called bidirectional RNNs (BRNNs) which were first introduced by [27] in the late 1990s. Nevertheless, they started to attract attention many years later after a groundbreaking achievement of sequence labeling in speech recognition by [13]. The BRNN is an RNN which contains a separation of a forward and backward pass for positive and negative time direction. Therefore, it is able to store the past and future context, whereas a conventional RNN can only partially achieve this by delaying the output by certain time steps. Our study is set up by employing more than 300 configurations for deep BRNNs after the preliminary tests, of which we inspect how the classification performance is affected by changing the *width* and the *depth* of the hidden layers. The designed networks are set up in a generic sense by simply stacking multilayer RNNs to have the required depth. The evaluation is based on three Motion Capture datasets for comparing BRNNs with ESNs [19, 20] and traditional machine learning techniques. Motion Capture (MoCap) is a marker-based system which by its high-dimensional nature, nonlinearities and long-range dependencies make it ideal for studying the limitations of time series models [35]. Although the focus of our study is on this particular domain, the design of BRNNs is not just solely specific for MoCap datasets. We are convinced that the study is also applicable to other high dimensional time series data.

## 2 Related Work

Many studies have demonstrated a superior functionality of applying deep RNNs when compared to shallow networks, for instance, [30] introduced a new architecture called multiplicative RNNs by using multiplicative connections to allow the current input character for the character-level in language modeling to determine the hidden-to-hidden weights. However, this model was trained with Hessian-Free optimizer (HF) instead of gradient descent. The work of [16] focused on a hierarchy of RNNs for character-level language modeling using stochastic gradient descent. It proposed two alternative architectures which are deep MLPs with three hidden layers stacked from one layer on top of each other with temporal feedback loops. One architecture uses feedback loops from output but with the

last hidden layer contributing to the output layer, while another architecture allows all the connections from each hidden layer contributing to the output layer. Four other different models to construct deep RNNs have been proposed by [26] for three language models. Quite recently, a hierarchical multiscale RNN model has been presented by [6]. It shows that the proposed network architecture can learn the latent hierarchical multiscale structure from temporal data for character-level language modeling. A similar study to our work which employs bidirectional RNNs for classification tasks has been discussed in [14]. It used five hidden layers of BRNNs with LSTM to classify 61 phoneme outputs in which each layer consists of  $2 \cdot 250$  cells. A comprehensive comparative study of deep RNNs has been revealed in [12]. It demonstrated the results of using from one to five hidden layers while fixing the number of neurons for all hidden layers. The results from this experiment exhibited that: i) LSTM works better than the typical  $\tanh$  neuron, ii) bidirectional RNNs with LSTM also give better output performances than typical unidirectional RNNs with LSTM units, and iii) the depth size is more important than the width size. In addition, by fixing the number of neurons of each hidden layer, the networks with three hidden layers work as well as those with five layers, while the number of weights of five hidden layers is almost twice their number for three layers. Furthermore, the evaluation in [18] also confirms that shallow BRNNs outperform shallow unidirectional RNNs on extracting sentence-level opinion expression. It concludes that, for a large network, three hidden layers provide the best output performance for their tasks. In case of a small network, two, three and four hidden layers show equally good performance for certain sizes. By adding more layers, its performance decreases. Further, the study suggests that in conventional stacked deep learners, every hidden layer conceptually lies in a different representation space, and establishes a more abstract and higher-level representation of the input. By taking these findings as our guidelines, we then hypothesized that the activities at each layer could represent some forms of the action descriptors.



■ **Figure 1** A deep BRNN with two hidden layers following [14]. The *dashed-dotted lines* indicate the **forward** direction depicted by  $\vec{h}_t$  and the *dashed lines* indicate the **backward** direction  $\overleftarrow{h}_t$ .

### 3 Classification Approaches

#### 3.1 Deep Bidirectional RNNs

Generally, the BRNN has been applied for sequence-to-sequence learning, particularly for Natural-Language Processing (NLP) tasks. The structure of the BRNN consists of two RNNs, one to compute the forward hidden sequences  $\vec{h}_t$  and the second is to compute backward hidden sequences  $\overleftarrow{h}_t$ . Figure 1 shows an architecture of the BRNN for two hidden layers. Let  $x = (x_1, \dots, x_T)$  be an input sequence for  $T$  time steps. The final output  $y_t$  is accumulated across the  $T$  frames at the last layer and is classified by the probability of



human action classes using the Softmax function. We computed the output sequence at time  $t$  of  $y$  according to [14] as:

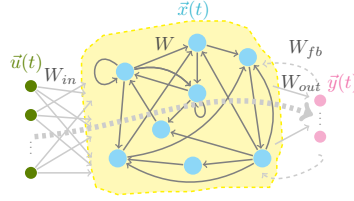
$$y_t = g(W_{\vec{h}^n y} \vec{h}_t + W_{\leftarrow h^{\leftarrow n} y} \bar{h}_t + b_y) \quad (1)$$

$g(\cdot)$  is an activation function,  $W_{\vec{h}^n y}$  and  $W_{\leftarrow h^{\leftarrow n} y}$  are weight matrices at the output layer and  $b_y$  is an output bias. The  $\bar{h}_t$  can be interpreted as a summary of the past from  $t = T$  to 1, whereas  $\vec{h}_t$  is the summary of the future from  $t = 1$  to  $T$ . The activities in forward and backward direction can be written by:

$$\vec{h}_t^n = f(W_{\vec{h}^n \vec{h}^n} \vec{h}_t^{n-1} + W_{\vec{h}^n \vec{h}^n} \vec{h}_{t-1}^n + b_{\vec{h}}^n) \quad (2)$$

$$\bar{h}_t^n = f(W_{\leftarrow h^{\leftarrow n} \leftarrow h^{\leftarrow n}} \bar{h}_t^{n-1} + W_{\leftarrow h^{\leftarrow n} \leftarrow h^{\leftarrow n}} \bar{h}_{t+1}^n + b_{\bar{h}}^n) \quad (3)$$

where  $h^0$  is the input sequence.



■ **Figure 2** Architecture of an ESN. The *dashed lines* denote the connections which are not compulsory.

### 3.2 Echo State Networks

An ESN shown in figure 2 is a type of RNN, whose design does not depend on updating weights by gradient computation, but, instead it creates a random *dynamical reservoir RNN*. The reservoir is then driven by the training data and leaves the weights untrained. The output weights are computed at the readout connection using a linear regression of  $y(t)$ . The internal unit activities  $\vec{x}$  in figure 2 can be updated by:

$$\vec{x}(t) = f(W_{in} \vec{u}(t) + W \vec{x}(t-1) + W_{fb} \vec{y}(t)) \quad (4)$$

$f(\cdot)$  is an activation function of the neurons, a common choice is  $\tanh(\cdot)$  applied element-wise. By employing the time warping of the input signals, the *leaky integration rate* [21]  $\alpha \in (0, 1]$  is adopted to determine the speed of the reservoir update dynamics. The update rule for the internal units is extended to:

$$\vec{x}_{leaky}(t) = (1 - \alpha) \vec{x}(t-1) + \alpha \vec{x}(t). \quad (5)$$

Applying a simple linear regression at the readout layer leads to output  $\vec{y}(t)$ :

$$\vec{y}(t) = W_{out} [\vec{u}(t); \vec{x}(t)] \quad (6)$$

### 3.3 Traditional Machine Learning Methods

Most work on time series classification for example, the well-known UCR archive, of which all datasets are of one dimensional feature, focuses on an adaptation of distance measures using 1-Nearest Neighbor (1-NN) with Euclidean Distance (ED) and Dynamic Time Warping (DTW). Both techniques have proven to perform very well on the UCR archive, especially DTW. Nevertheless, DTW has limited its applications only on the fixed length data (that is one must extrapolate the shorter sequence in order to have an equal length between two sequences) and gives rise to some issues for the case of multi-dimensional data e.g., the computational complexity and the selection of the dependent or independent warping distance function [32].

We adopted two transformation approaches to extract the feature vectors in our experiments. They are **i) a naïve method** by stacking each frame on top of one another, and using majority voting to decide the best course of action, and **ii) a dimensionality reduction technique** of the zero-mean skeleton configuration for feature vectors demonstrated in [34, 23]. The two best classifiers for MoCap classification, cited in [34], k-NN and Random Forest (RF), were chosen for both transformation approaches.

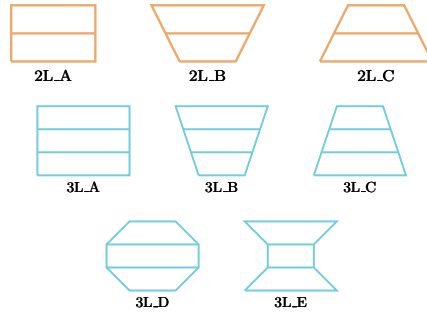
## 4 Configurations

### 4.1 BRNN architectures

The RNNs with one hidden layer ( $\mathbf{1L}$ ) represent shallow networks, while those with more than one hidden layers represent the deep RNNs. According to [14] and [18], three hidden layers are sufficient to achieve the best performance, while adding more hidden layers worsens the output performance. On this account, in our experiment, we concentrated on evaluating geometries of the networks with two and three hidden layers as illustrated in figure 3. RNNs with three stacked hidden layers numbered from bottom to top labeled as ( $\mathbf{L}_1 \cdot \mathbf{L}_2 \cdot \mathbf{L}_3$ ) will be referred to throughout the experiment indicating the number of units in *one direction*. (We use the term cell or unit instead of neuron to emphasize the use of gated units in RNNs.) Model  $2L\_A$ , as seen in the figure 3 is depicted for the two hidden layers in which the number of cells of the two hidden layers is almost equal. The model  $2L\_B$  is for two hidden layers, of which the number of neurons of  $L_2$  (top) is at least double the size of  $L_1$  (bottom), and vice versa for model  $2L\_C$ . Note that the first hidden layer ( $L_1$ ) connects to the input nodes laying at the bottom. For models with three hidden layers, we extended the geometries to five architectures as illustrated in figure 3. Here, we combined cell numbers in  $\{50, 100, 150, 200, 250, 300, 350, 400\}$  to form these geometries which have a limited total amount of cells from  $2 \cdot 200$  up to  $2 \cdot 600$  units.

### 4.2 BRNN configurations and hyperparameters

In order to verify the impact of the width and the depth on the network as well as to simplify the experiment, several parameters in the experiment had been previously investigated. They are: **i) Cell type**. In our experiment, instead of a well-known LSTM, we replaced each cell unit at  $h$  with a gated recurrent unit called GRU [4]. GRU is a variation of a gating mechanism and is comparable to LSTM and has been primarily used in machine translation as encoder-decoder models. It is similar to LSTM in that the gating units modulate the flow of information inside the unit, but without memory cells. Comparative studies of using traditional recurrent unit  $\tanh$ , LSTM and GRU, found in [7] and [22] have shown that the gated units, both the LSTM and GRU outperform the conventional recurrent unit. To



■ **Figure 3** Different geometries representing eight models used in our experiment with varying width of two and three hidden layers.

achieve certainty, we had examined these three cells in BRNNs in our preliminary tests. The networks with GRU cells significantly outperformed the other two cells by more than 5% in all experiments. Therefore, our classification results were from applying the GRU units to the networks. **ii) Optimizer.** In contrast to the optimizer benchmark for RNNs in Penn TreeBank language modeling [8] which mentions that Adam and RMSProp do not work well with RNNs, we found that in our case, both of them converged very quickly even with a very small learning rate and gave good output performance. The primary test was carried out for a shallow BRNN. The selected learning rates for each optimizer here were the recommended values in the papers based on MNIST dataset. For the rest of the experiments, we chose RMSProp as our default optimizer which outperformed all other optimizers. **iii) Regularization.** Because of the limited amount of data, we cannot take out some data for validation. Nonetheless, we added a regularization term  $L^2$  to the objective function with a fixed regulation  $\lambda = 0.02$ . It is interesting to note that increasing the regularization parameter from 0.01 to 0.02 increases the recognition rate by about 3-5% in most models. We applied the norm clipping with a maximum gradient norm limited to one, and no *dropout* was applied.

### 4.3 ESN configurations

The ESN configurations in the experiments follow the guidelines suggested in [33] which demonstrated the influence of the ESN settings on various datasets on the UCR archive. Several key parameters are: **i) Sparsity of the reservoir.** In corresponding with BRNN networks which have the networks size in  $2 \cdot \{200, \dots, 600\}$ , we set the reservoir size using only half of BRNN with connectivity of 10, 30, 50 and 70% respectively. **ii) Spectral radius** which is considered to be big for the tasks that require an extensive history of an input, while one is served as a reference point. We picked 5.0 from [33]. **iii) Leaky rate** which can be regarded as time warping of the input signals was fixed at 0.1 for all configurations. **iv) Input scaling:** was set to 2.0 similar to [33] and **v) regularization coefficient** was fixed at 0.1. Moreover, the network weight was set to have a uniform distribution in the range of  $[-0.5, 0.5]$  and no feedback connection was considered here.

### 4.4 Traditional machine learning configurations

For the direct naïve method, we employed two popular classifiers for classification, 1-NN and RF with 75 trees, which yielded best output performance in [34]. To prove the case of a combined manifold learning with classification, we chose the PCA with two and three

components associated with RF and 1-NN.

## 5 Datasets and Experimental Setups

### 5.1 Datasets

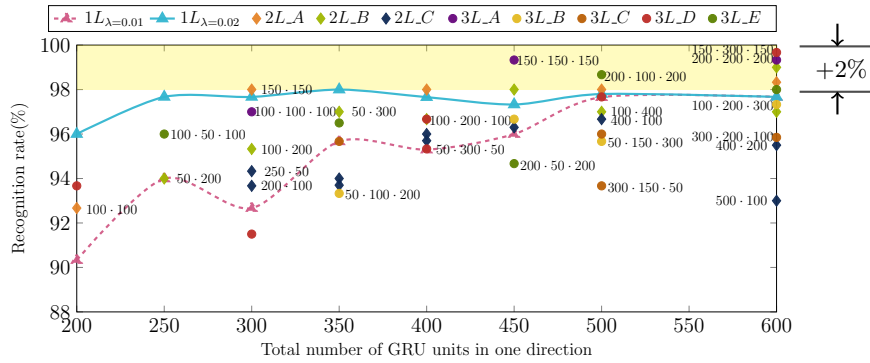
We evaluated the proposed techniques as described in section 3 and 4 using three MoCap datasets, MHAD-27, MHAD-10 and HDM05.

**MHAD-27**<sup>3</sup> [2] consists of 27 different actions performed by eight subjects. Each subject repeated the same action four times. The dataset contains a total of 861 data sequences, where three sequences were corrupted and removed from the dataset on the official website. The training was performed on six subjects, and two subjects were left out for the test. The recognition rate was reported on an average of 28 combinations. This dataset was recorded using 20 markers. Therefore, we have the feature vector which is captured in 3D space for BRNN of size  $(3 \cdot 20) \times 120$ , where 120 is the amount that is close to the maximal sequence length of this dataset. Zeros were appended to the shorter sequences in BRNNs. It is important to note that ESN can handle different lengths of the data sequences, so the data is trained with the original length.

**MHAD-10** [2] is a 3D MoCap dataset of six subjects performing 10 different hand gestures tracked with 25 markers. The four additional markers comparison with MHAD-27 were put on the left and right hand and a thumb, and one additional marker was put on a spine. Each subject repeats an action 5 times (trials). Therefore, we have a total of 300 videos with various sequence lengths. For BRNN, we chose to fix the number of sequences to 150 which is close to the maximal length of the sequence in the dataset. This makes a feature vector size of  $(3 \cdot 25) \times 150$ . The training is performed on five subjects from all trials and an unseen subject is left out for testing. Hence the recognition rate is an average of six-fold cross-validation.

**HDM05** [25] was originally made up of 130 classes consisting of five subjects performing actions with and without repeating the same cycles separately. This created a total of 2343 sequences. We followed [5, 9, 38] in grouping non-repetitive and repetitive motions together yielding 65 actions. There were about 20 actions which have samples less than 20 i.e., *throwBasketball*, *throwFarR* and *jumpDown* having only 14 trials each, while actions such as *walk*, *elbowToKnee* and *runOnPlaceStart* have 94, 80, 74 trials, respectively. This leads to an unbalancing of data and causes a huge bias towards a particular action. Nonetheless, since we focused on the action recognition of unseen subjects, therefore four subjects were used in the training set and one subject was for the test. We reduced the original number of markers to 19, where some nearby sensors e.g., on the spine as used in MHAD were merged. The average sequence length is 261 with the maximum of 901. With the limitation of our computational capacity, we set the data using the fixed length of 400 for BRNNs. Therefore, the feature vectors of BRNNs have a size of  $(3 \cdot 19) \times 400$ .

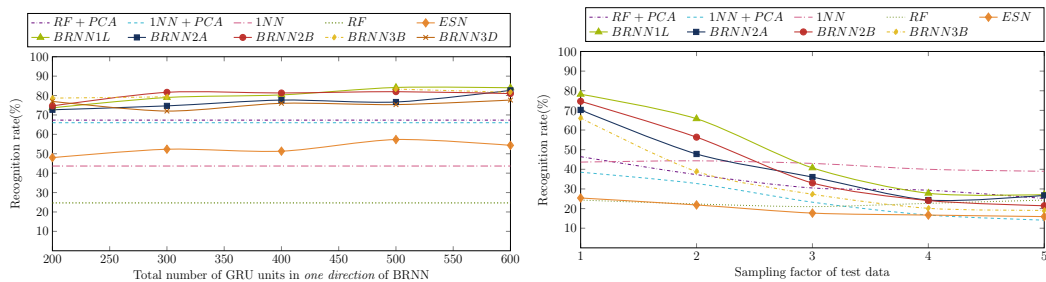
<sup>3</sup> This dataset, MHAD-27, is the first version of UTD-MHAD, where MHAD-10 uses a second generation of Kinect camera which came out later.



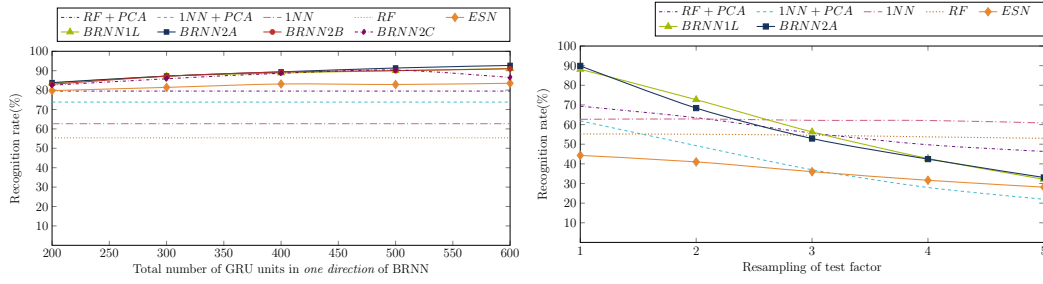
■ **Figure 4** The recognition rates of MHAD-10 from designated network architectures using five-fold cross validation. The setup does not condition on the separation of test subjects from the training set.

## 5.2 Experimental Setups

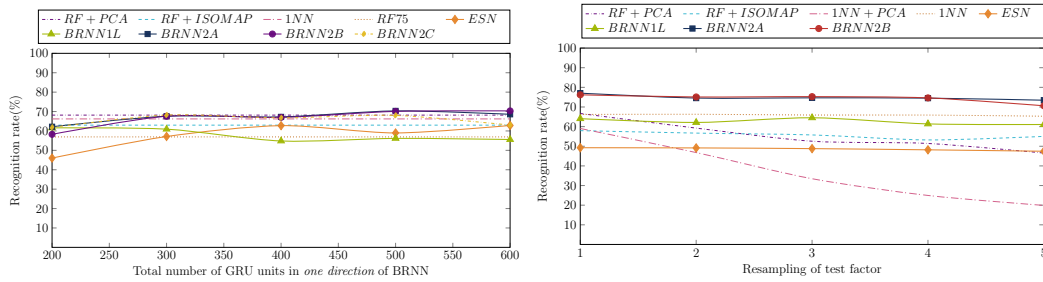
We normalized each sequence with respect to its in-frame reference of that dataset, where the reference is the joint that laid at the center of the skeletal torso. The evaluations were composed of three experiments: **i) Having some insights of deep networks strategies by varying the width and height of the network.** In this experiment, we did not impose conditions on separation of test subjects from the training set. **ii) Finding a few good models by varying the number of cells in the networks using unknown subjects.** The experiment was set up in a way to find a few good models of each dataset by varying the number of cells in the networks in 200-600 units (one direction in BRNN and the total units in a reservoir for ESN), where the test subjects were excluded from the training set. More than 300 configurations for BRNN were constructed to obtain the best output and created some insights of construction strategies for deep networks. **iii) Extending the test using untrained viewpoints with the variations of speed.** We enhanced the experiment by extending the training set to have five camera angles  $\{-90, 45, 0, 45, 90\}$ , whereas test angles are in  $\{-80, -70, \dots, 70, 80\}$ . Moreover, we subsampled the original test data using a subsampling factor of 1, 2, 3, 4 and 5, while the training data still remained the same. The subsampling factor of 2 means that every  $2^{nd}$  frame of the data will be taken instead of each single frame (factor of 1).



■ **Figure 5** The recognition rates of MHAD-10 by excluding test subjects from the training set. **Left)** Changing the total number of GRU units in one direction of BRNN or the reservoir size of ESN. **Right)** Extending the test on the left by changing subsampling factors of test data of untrained viewpoints.



■ **Figure 6** The recognition rates of MHAD-27 by excluding test subjects from the training set. **Left)** Changing the total number of GRU units in one direction of BRNN or the reservoir size of ESN. **Right)** Extending the test on the left by changing subsampling factors of test data of untrained viewpoints.



■ **Figure 7** The recognition rates of HDM05 by excluding test subjects from the training set. **Left)** Changing the total number of GRU units in one direction of BRNN or the reservoir size of ESN. **Right)** Extending the test on the left by changing subsampling factors of test data of untrained viewpoints.

## 6 Experimental Results

### 6.1 Discussion of results

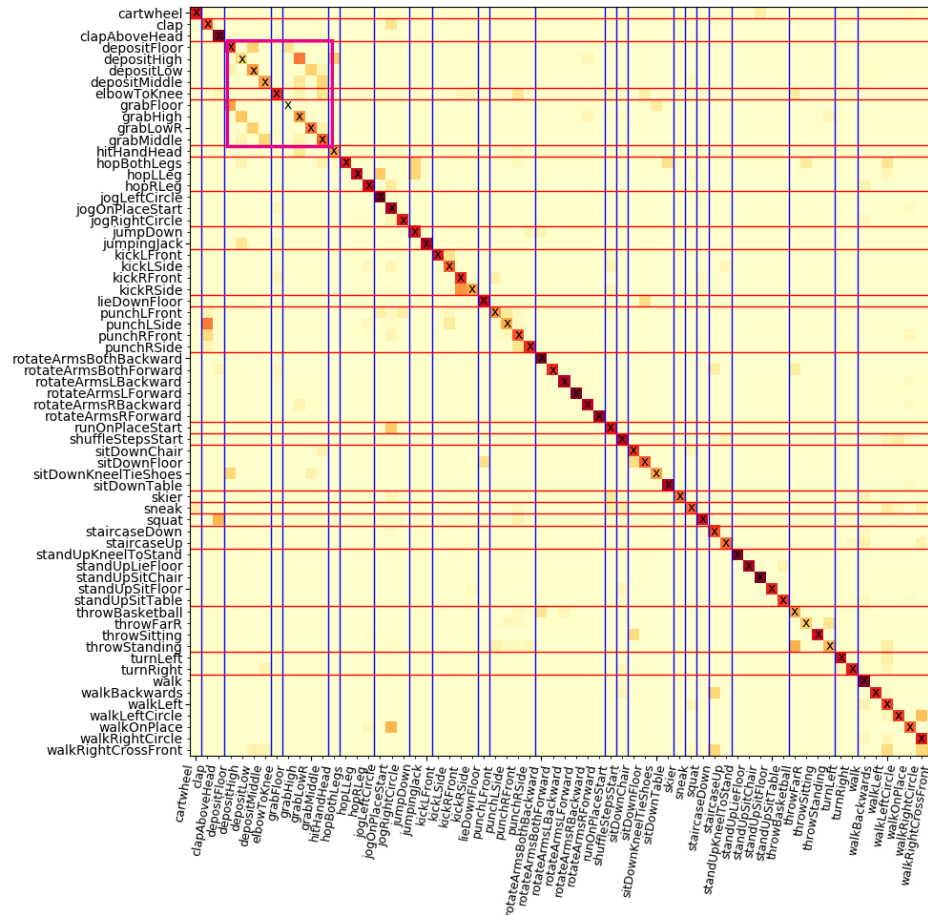
Firstly, we investigated the effects of geometries of BRNNs following the **construction strategies** depicted in Figure 3. The recognition rates of MHAD-10 performing on average of five-fold cross-validation using shallow and deep BRNNs are shown in Figure 4. Moreover, the recognition rates of each strategy are the average of two runs with the standard deviation of  $\pm 3\%$ . These test results gain very high recognition rates because they are not based on the separation of test subjects from the training set. It is obvious that the recognition rates which are better than 98% (the yellow shaded area in Figure 4) can only be achieved when the networks are relatively large i.e., the number of cells is greater than  $2 \cdot 400$ . Furthermore, the models which have any layer containing 50 cells yield output worse than others. This might be because the input feature of MHAD-10 has a size of 75 and any form of dimensionality reduction at any hidden layer in RNNs by shrinking the network's width is not suitable. Hence, by the experimental results, we conclude that the width of a hidden layer next to the input layer in one direction should be larger than the size of the input features.

We enhanced the experiment by **excluding test subjects from the training set to examine the generalization of the models**. The recognition rates of three datasets for two experimental setups of MHAD-10, MHAD-27 and HDM05 can be found in Figure 5, 6 and 7 on the left and right of the figures respectively. In these figures, only a few best models of BRNN were chosen from various configurations based on five geometries in Figure 3.

Furthermore, the results also demonstrate the output of the shallow networks, the output of ESNs in section 3.2 and the output of machine learning approaches from section 3.3.

The results of the first experiment displayed on the left of Figure 5, 6 and 7 show that the shallow and deep BRNNs significantly outperform other methods in the MHAD-10 and MHAD-27 dataset. The deep BRNNs are slightly better than the other methods on large networks for HDM05. In addition, there are some considerable differences in recognition rates among both datasets and algorithms, especially for MHAD-10. Even though MHAD-10 consists of only 10 actions, nine out of these actions are the movements of solely the right hand. On the contrary, even though MHAD-27 consists of more actions than MHAD-10, the actions also involve a variety of movements of hands and legs which leads to overall better recognition rates of all methods. The dimensionality reduction techniques in combination with RF outperform other machine learning techniques and are close to the winner of HDM05 using deep networks, *BRNN2C* using the total of  $2 \cdot 600$  cells. The confusion matrix of HDM05 using *BRNN2B* model on Figure 7-left is depicted in Figure 8. Actions with one hundred percent recognition rate (recall) from all runs (filled with dark brown in Figure 9) are for instance, *clapAboveHead*, *jogLeftCircle*, *rotateArmsBothBackward*, *standUpKneelToStand* and so forth. The most common misclassified actions in all classifiers are between *deposit* and *grab* as the trajectories of actions drawn in Figure 9. It is difficult to track the trajectories in 3D of a stationary image by eye; hence, we projected a 3D image onto a 2D plane and as we can see, the trajectories from these two groups cannot be distinguished. Therefore, when we allow classification using top three correctness, the recognition rates increase by 12-15% in all methods. For HDM05, there are no significant differences among different classification methods. When applying subsampling factors to **simulate the changes of movement speed** for the test data of MHAD-10 and MHAD-27 using untrained viewpoints, then increasing the subsampling factor decreases the recognition rate. Interestingly, however, this effect does not apply to HDM05. This might be because only HDM05 consists of non-repetitive and repetitive sequences in one action which allows the networks to easily capture the changes of patterns of the action as varying of movement speeds, while MHAD-10 and MHAD-27 only consist of one periodic movement in each action. Besides, by increasing the number of viewpoints in training deep BRNNs, the recognition rates of HDM05 have been increasing by approximately 10% on arbitrary untrained viewpoints.

The results also reveal that deep BRNNs using two layers for the total number of cells greater than  $2 \cdot 500$  units such as *BRNN2A* and *BRNN2B* surpass all other models for all three datasets, including three layers of BRNNs. The shallow BRNNs work equally well or even better than the deep BRNN for MHAD-10 and MHAD-27 but not for HDM05. It is important to note that models with three hidden layers do not perform better than models with two hidden layers, while training such gradient-based approaches requires a large amount of computation time on GPU. The computation time and cost of training and testing BRNNs is much greater than training ESNs, which the training is only performed for the output weights at the readout where there is no cyclic dependency. Training and testing using dimensionality reduction methods demand the least time and computational power. Considering the time complexity for the gradient-based learning by BPTT, it must be analyzed in terms of space for the number of values stored and the time complexity in terms of the number of arithmetic operations required [37]. Therefore, measuring architecture complexity of RNNs is not a trivial task. Nonetheless, for our configurations when the network is fully connected and all weights are adaptable, if the shallow network requires time  $\mathcal{T}$  to complete the task, the deep network can be expected to complete the task in about  $\mathcal{L} \cdot \mathcal{T}$ , where  $\mathcal{L}$  is the number of hidden layers.



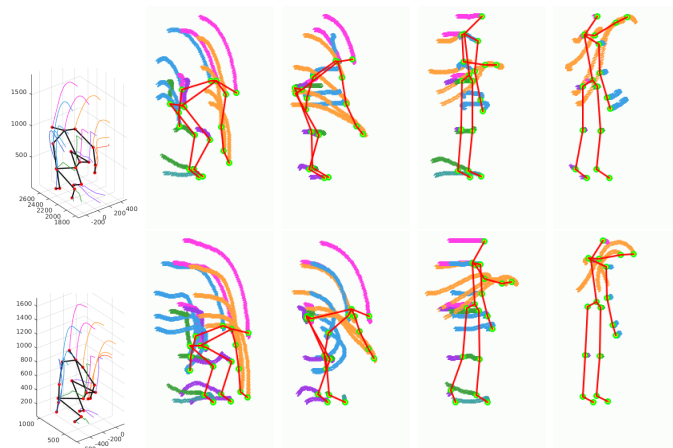
■ **Figure 8** Confusion matrix of HDM05 with 65 actions on average of 2-folds using best model of *BRNN2B* ( $2 \cdot [100 \cdot 400]$ ). The weighted colors are computed from the percentage of the total number of that action. The thick pink rectangle at the left corner shows a group of actions which significantly misclassified in all methods. The red horizontal and vertical blue lines are drawn to highlight groups of the actions.

**A Comparison.** Other studies which resemble our first experiment use only one default view and do not exclude test subjects from training data. Furthermore, some approaches apply some prior filters before passing data to the networks, for instance, [3] proposed a hybrid MLPs which reported the recognition rate with an accuracy of 95% on HDM05 on 10-fold evaluation. Next, [9] introduced deep BRNN by stacking BRNNs using LSTM units on each skeleton part yielding the best result of 96.92% for HDM05. Followed by [38] which use deep BRNNs with LSTM units on body parts similar to [9] but added a so-called co-occurrence matrix and dropout to a three-LSTM layer with two feedback layers. It accounted for the recognition rate of 97.25%.

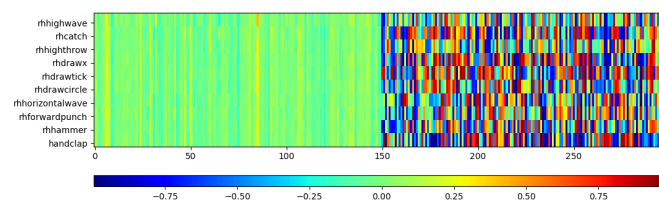
## 6.2 Visualization of BRNN

**Last hidden layer.** For the BRNNs, only half of the output activities of the last hidden layer contribute to the output layer as can be seen in Figure 10. This figure could explain

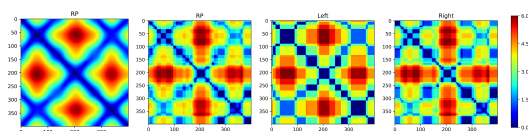




■ **Figure 9** Most common misclassified patterns are the confusion between “*deposit*” and “*grab*”. Top) From left to right: 3D projection of *depositFloor* of default view, and the rotated 2D projections of *depositFloor*, *depositLow*, *depositMiddle* and *depositHigh*, Bottom) *grabFloor* in 3D and 2D projections of *grabFloor*, *grabLowR*, *grabMiddle* and *grabHigh*.



■ **Figure 10** The activities from the last hidden layer with  $2 \cdot 150$  cells of MHAD-10 dataset. The left side shows the activities from forward and the right side from the backward direction.



■ **Figure 11** The Euclidean distance matrices of a subject performing *elbowToKnee*. From left to right: i) input ii) the retrieved activities from the first hidden layer of combined directions, iii) forward and iv) backward direction.

why the number of cells in one direction at the last hidden layer needs to be much greater than the number of the output classes which are required by the Softmax function at the output.

**Visualization of input and other hidden layers.** Multidimensional time-series data cannot be directly visualized, therefore investigating its behavior is very difficult. One common approach that is normally used in order to get some insight into high dimensional time-series data is by examining their distance matrix. One benefit of using distance matrices, such as Euclidean distance is that we can further analyze the matrix using recurrence plots (RPs) [10] by applying a threshold distance and the Heaviside function. The RPs can tell

when the phase space trajectory of the dynamic system re-occur roughly in the same area in the phase space. Figure 11 shows the Euclidean distance matrices of a subject in HDM05 performing an *elbowToKnee*. The left-most of the figure shows the distance matrix of the input which reveals a few harmonic oscillations that can be observed by the checkerboard structures. The next three figures are the activities from the first hidden layer  $L_1$  for the combination of both directions, for forward and backward direction, respectively. We can infer from the changes of one state of learning to another that the networks opt to differentiate their output activities at each layer. At the upper layer, the scale of the differentiation is larger. The very dark blue corresponds to distance zero and red to the maximum distance between the features in this time span.

## 7 Conclusion

During the course of conducting our research, we have demonstrated the various influences of various geometries of the deep BRNN's upon human motion recognition. It is crucial to have some empirical insights to amend and influence both the width and depth of the model to suit the research requirements and objectives. The evaluations of the classifications were performed by focusing on the generalization and the robustness of the models by testing on unseen subjects with the variation of movement speeds. The results showed that BRNNs outperformed ESNs and other conventional classification techniques. Correspondingly, we discovered that any form of dimensionality reduction, caused by reducing the width of the hidden layers to less than the number of input features or reducing the width of last hidden layer in one direction less than the output units is unsatisfactory. The shallow networks should be included and examined in the experiment as they may not only demonstrate good performance for some datasets, but also provide some insights into the impact of hyper parameters. Nonetheless, to achieve the best outcomes, based on our research, we strongly recommend that deep RNN's as the method of choice for researchers to employ.

---

## References

- 1 Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *Trans. Neur. Netw.*, 5(2):157–166, March 1994. doi:10.1109/72.279181.
- 2 Chen Chen, Roozbeh Jafari, and Nasser Kehtarnavaz. *UTD-MHAD: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor*, volume 2015-December, pages 168–172. IEEE Computer Society, 12 2015. doi:10.1109/ICIP.2015.7350781.
- 3 Kyunghyun Cho and Xi Chen. Classifying and visualizing motion capture sequences using deep neural networks. *CoRR*, abs/1306.3874, 2013. URL: <http://arxiv.org/abs/1306.3874>.
- 4 Kyunghyun Cho, B van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. *On the properties of neural machine translation: Encoder-decoder approaches*. 2014.
- 5 Kyunghyun Cho, Bart van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. Association for Computational Linguistics, October 2014.
- 6 Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. Hierarchical multiscale recurrent neural networks. *ICLR 2017 conference*, September.

- 7 Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. *Empirical evaluation of gated recurrent neural networks on sequence modeling*. 2014.
- 8 Timothy Dozat. Incorporating nesterov momentum into adam. 2015.
- 9 Yong Du, Wei Wang, and Liang Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.
- 10 J. P. Eckmann, Oliffson S. Kamphorst, and D. Ruelle. Recurrence plots of dynamical systems. *Europhysics Letters*, 4, November 1987.
- 11 Mona Fathollahi and Rangachar Kasturi. Autonomous driving challenge: To infer the property of a dynamic object based on its motion pattern using recurrent neural network. *CoRR*, abs/1609.00361, 2016.
- 12 A. Graves, A. r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, May 2013. doi:10.1109/ICASSP.2013.6638947.
- 13 Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd International Conference on Machine Learning, ICML '06*, pages 369–376, New York, NY, USA, 2006. ACM. URL: <http://doi.acm.org/10.1145/1143844.1143891>, doi:10.1145/1143844.1143891.
- 14 Alex Graves, Navdeep Jaitly, and Abdel-rahman Mohamed. Hybrid speech recognition with Deep Bidirectional LSTM. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pages 273–278. IEEE, December 2013. URL: <http://dx.doi.org/10.1109/asru.2013.6707742>, doi:10.1109/asru.2013.6707742.
- 15 Alex Graves, Marcus Liwicki, Santiago Fernández, Roman Bertolami, Horst Bunke, and Jürgen Schmidhuber. A novel connectionist system for unconstrained handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, 31(5):855–868, May 2009. URL: <http://dx.doi.org/10.1109/TPAMI.2008.137>, doi:10.1109/TPAMI.2008.137.
- 16 Michiel Hermans and Benjamin Schrauwen. Training and analysing deep recurrent neural networks. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 190–198. Curran Associates, Inc., 2013. URL: <http://papers.nips.cc/paper/5166-training-and-analysing-deep-recurrent-neural-networks.pdf>.
- 17 S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies. In Kremer and Kolen, editors, *A Field Guide to Dynamical Recurrent Neural Networks*. IEEE Press, 2001.
- 18 Ozan İrsoy and Claire Cardie. Opinion mining with deep recurrent neural networks. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 720–728, 2014. URL: <http://aclweb.org/anthology/D14-1080>.
- 19 Herbert Jaeger. Adaptive nonlinear system identification with echo state networks. In *Advances in Neural Information Processing Systems 15, NIPS 2002*, pages 593–600, 2002.
- 20 Herbert Jaeger and Harald Haas. Harnessing nonlinearity: predicting chaotic systems and saving energy in wireless telecommunication. *Science*, 304(5667):78–80, 2004.
- 21 Herbert Jaeger, Mantas Lukoševičius, Dan Popovici, and Udo Siewert. Optimization and applications of echo state networks with leaky-integrator neurons. *Neural Networks*, 20(3):335–352, 2007.
- 22 Rafal Jozefowicz, Wojciech Zaremba, and Ilya Sutskever. An empirical exploration of recurrent network architectures. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2342–2350, Lille, France, 07–09 Jul 2015. PMLR. URL: <http://proceedings.mlr.press/v37/jozefowicz15.html>.

- 23 Marco Körner and Joachim Denzler. Analyzing the subspaces obtained by dimensionality reduction for human action recognition from 3d data. In *IEEE International Conference on Advanced Video and Signal-Based Surveillance (AVSS)*, pages 130–135, 2012.
- 24 Yajie Miao and Florian Metze. On speaker adaptation of long short-term memory recurrent neural networks. In *in Sixteenth Annual Conference of the International Speech Communication Association (INTERSPEECH) (To Appear)*. ISCA, 2015.
- 25 M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. Technical Report CG-2007-2, Universität Bonn, June 2007.
- 26 Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. *How to construct deep recurrent neural networks*. 2014.
- 27 M. Schuster and K.K. Paliwal. Bidirectional recurrent neural networks. *Trans. Sig. Proc.*, 45(11):2673–2681, November 1997. URL: <http://dx.doi.org/10.1109/78.650093>, doi: 10.1109/78.650093.
- 28 Rajiv Shah and Rob Romijnders. Applying deep learning to basketball trajectories. *KDD 2016, Large Scale Sports Analytic Workshop*, 2016.
- 29 Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In David Blei and Francis Bach, editors, *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pages 843–852. JMLR Workshop and Conference Proceedings, 2015. URL: <http://jmlr.org/proceedings/papers/v37/srivastava15.pdf>.
- 30 Ilya Sutskever, James Martens, and Geoffrey E. Hinton. Generating text with recurrent neural networks. In Lise Getoor and Tobias Scheffer, editors, *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024, New York, NY, USA, 2011. ACM. URL: [http://www.icml-2011.org/papers/524\\_icmlpaper.pdf](http://www.icml-2011.org/papers/524_icmlpaper.pdf).
- 31 Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems, NIPS’14*, pages 3104–3112, Cambridge, MA, USA, 2014. MIT Press. URL: <http://dl.acm.org/citation.cfm?id=2969033.2969173>.
- 32 Preetreya Tanisaro and Gunther Heidemann. Time series classification using time warping invariant echo state networks. In *15th IEEE International Conference on Machine Learning and Applications, (ICMLA)*, 2016.
- 33 Preetreya Tanisaro, Constantin Lehman, Leon Sütfeld, Gordon Pripa, and Gunther Heidemann. Classifying bio-inspired model in point-light human motion using echo state network. In *The 26th International Conference on Artificial Neural Networks (ICANN), 2017*, Lecture Notes in Computer Science. Springer, 2017.
- 34 Preetreya Tanisaro, Florian Mahner, and Gunther Heidemann. Quasi view-independent human motion recognition in subspaces. In *Proceedings of 9th International Conference on Machine Learning and Computing (ICMLC)*, ICMLC 2017, pages 278–283. ACM, 2017. URL: <http://doi.acm.org/10.1145/3055635.3056577>, doi:10.1145/3055635.3056577.
- 35 Graham W. Taylor, Geoffrey E. Hinton, and Sam T. Roweis. Two distributed-state models for generating high-dimensional time series. *J. Mach. Learn. Res.*, 12:1025–1068, July 2011. URL: <http://dl.acm.org/citation.cfm?id=1953048.2021035>.
- 36 Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3156–3164, 2015.
- 37 Ronald J. Williams and David Zipser. Gradient-based learning algorithms for recurrent networks and their computational complexity, 1995.
- 38 Wentao Zhu, Culing Lan, Junliang Xing, Wenjun Zeng, Yanghao Li, Li Shen, and Xiaohui Xie. Co-occurrence feature learning for skeleton based action recognition using regular-

ized deep lstm networks. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, pages 3697–3703, 2016.

## A Lists of Complete Actions

Action	#Trials	minLength	maxLength	meanLength
RHHighWave	30	55	161	85
RHCatch	30	38	75	54
RHHighThrow	30	44	78	56
RHDrawX	30	55	81	64
RHDrawTick	30	43	72	57
RHDrawCircle	30	54	89	67
RHHorizontalWave	30	52	139	70
RHForwardPunch	30	42	71	55
RHHammer	30	51	85	65
HandClap	30	47	68	57

■ **Table 1** Ten hand gestures in MHAD-10 from default view at 0°.

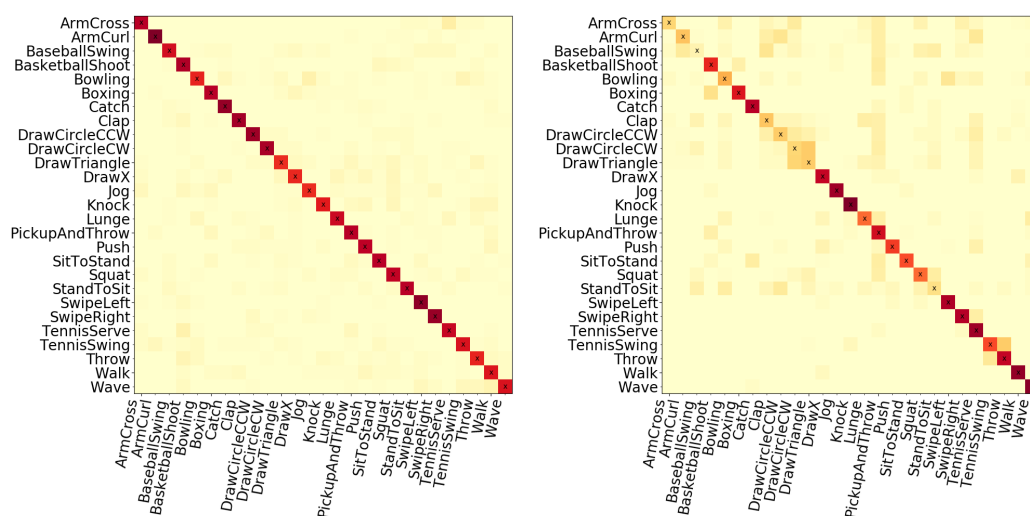
Action	#Trials	minLength	maxLength	meanLength
ArmCross	32	50	86	64
ArmCurl,	32	42	86	59
BaseballSwing	32	63	90	74
BasketballShoot	32	46	81	60
Bowling	32	64	101	76
Boxing	32	51	92	68
Catch	32	41	74	59
Clap	32	51	78	61
DrawCircleCCW	32	64	98	74
DrawCircleCW	32	66	95	75
DrawTriangle	32	61	106	77
DrawX	31	55	81	66
Jog	32	51	82	67
Knock	32	53	95	67
Lunge	32	63	103	80
PickupAndThrow	32	68	125	87
Push	32	47	80	62
SitToStand	32	47	69	54
Squat	31	50	116	82
StandToSit	32	46	71	57
SwipeLeft	32	48	76	61
SwipeRight	32	47	75	59
TennisServe	32	52	94	67
TennisSwing	32	44	87	64
Throw	32	44	70	58
Walk	31	60	104	76
Wave	32	49	81	65

■ **Table 2** 27 actions in MHAD-27 from default view at 0°.

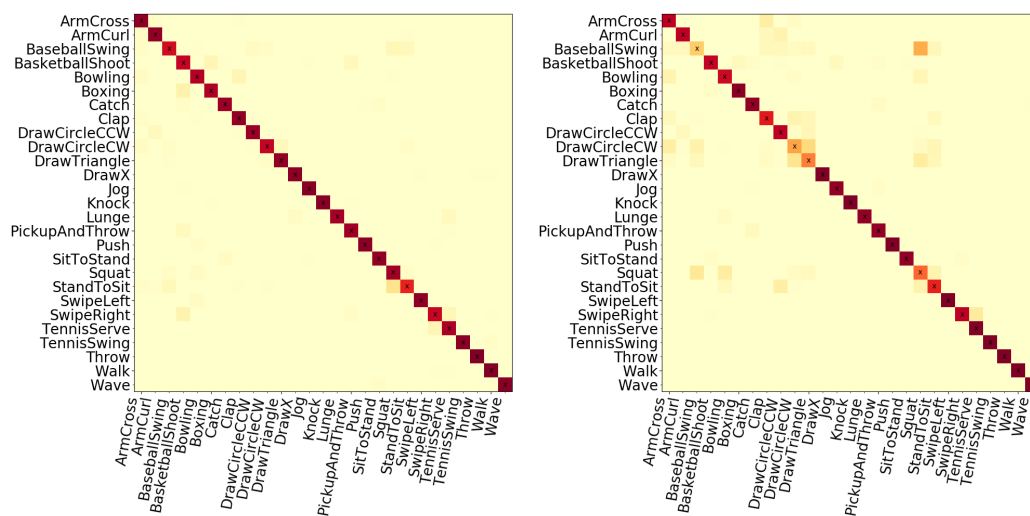
Action	#Trials	minLength	maxLength	meanLength
cartwheel	28	281	701	450
clap	31	32	211	109
clapAboveHead	31	59	657	248
depositFloor	32	181	641	363
depositHigh	28	129	509	245
depositLow	28	196	481	277
depositMiddle	29	178	662	270
elbowToKnee	80	93	574	243
grabFloor	16	186	401	268
grabHigh	29	170	460	258
grabLowR	29	191	544	294
grabMiddle	28	112	352	210
hitHandHead	13	141	281	226
hopBothLegs	55	56	432	151
hopLLeg	64	62	254	119
hopRLeg	65	58	246	116
jogLeftCircle	32	197	400	292
jogOnPlaceStart	70	80	241	147
jogRightCircle	33	190	441	288
jumpDown	13	177	381	288
jumpingJack	65	116	484	201
kickLFront	43	129	841	294
kickLSide	39	131	721	315
kickRFront	45	121	668	296
kickRSide	44	127	740	310
lieDownFloor	20	301	901	655
punchLFront	45	119	761	263
punchLSide	45	90	721	235
punchRFront	45	138	761	286
punchRSide	42	97	662	242
rotateArmsBothBackward	32	62	649	214
rotateArmsBothForward	32	62	739	230
rotateArmsLBackward	32	57	708	215
rotateArmsLForward	32	55	739	215
rotateArmsRBackward	32	54	649	210
rotateArmsRForward	32	54	733	213
runOnPlaceStart	74	58	182	100
shuffleStepsStart	51	161	540	319
sitDownChair	20	154	441	318
sitDownFloor	20	224	601	407
sitDownKneelTieShoes	17	425	825	645
sitDownTable	20	162	401	270
skier	40	123	459	202
sneak	63	164	751	372
squat	65	136	823	271
staircaseDown	15	139	319	222
staircaseUp	27	164	444	292
standUpKneelToStand	17	100	301	182
standUpLieFloor	20	279	703	525
standUpSitChair	20	176	441	295
standUpSitFloor	20	167	641	403
standUpSitTable	20	121	454	250
throwBasketball	14	281	721	407
throwFarR	14	361	600	524
throwSitting	28	188	404	282
throwStanding	28	242	541	353
turnLeft	30	119	281	196
turnRight	30	135	260	196
walk	94	122	369	214
walkBackwards	30	158	433	299
walkLeft	32	277	659	411
walkLeftCircle	37	261	560	397
walkOnPlace	60	121	400	233
walkRightCircle	27	246	542	381
walkRightCrossFront	29	195	701	434

■ **Table 3** 65 actions in HDM05 from default view at  $0^\circ$ .

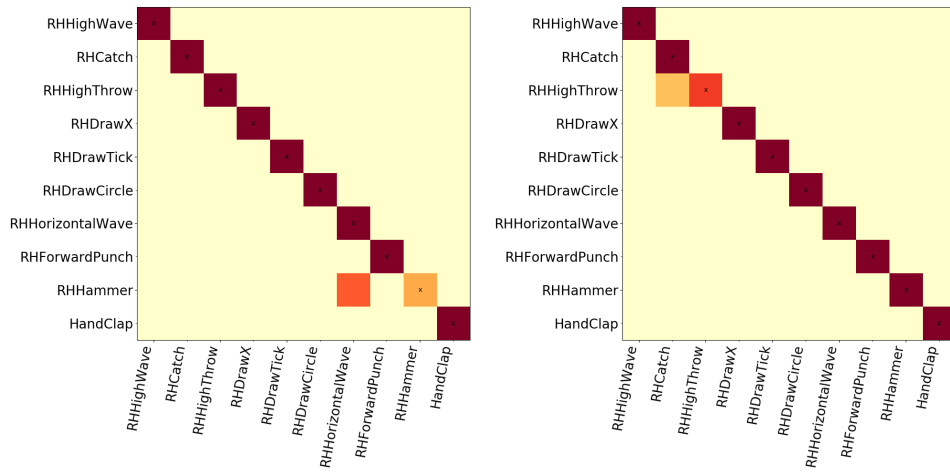
## B Confusion Matrices from Various Models



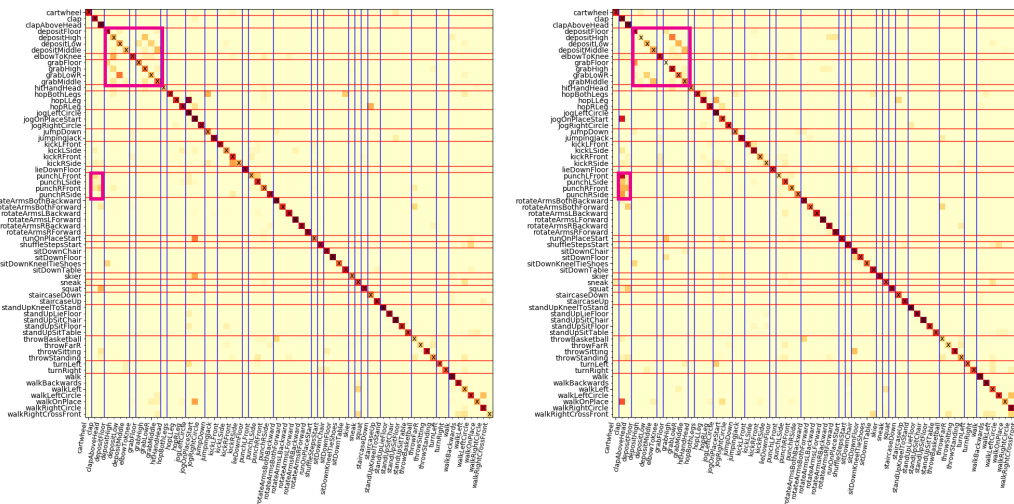
■ **Figure 12** Confusion Matrices of MHAD-27 of the first testing fold from two classification methods. Left) Subspaces employing RF with PCA. Right) Majority vote using 1-NN.



■ **Figure 13** Confusion Matrices of MHAD-27 of 27 folds. Left) BRNN of  $2 \cdot [300 \cdot 300]$  cells. Right) ESN with network size of 600 neurons.



■ **Figure 14** Confusion Matrices of MHAD-10 with the same configurations of BRNN employing  $2 \cdot [50 \cdot 150 \cdot 250]$  cells, but testing on different subjects shown on the left and the right figure.



■ **Figure 15** Confusion Matrices of HDM05 of all folds in each configuration. Two common misclassified groups of actions are highlighted in the pink color. Left) BRNN of  $2 \cdot [200 \cdot 400]$  cells. Right) BRNN of  $2 \cdot [250 \cdot 250]$  cells.



## 2.6 Dimensionality Reduction for Visualization of Time Series and Trajectories

**Abstract:** Visualization is essential for data analysis and it is particularly challenging for data in high-dimensional space, especially for temporal information. Many techniques have been employed in an attempt to transform multivariate time series data to one-dimensional data by reducing the number of features in order to visualize their time-dependent behaviors. However, the applicability of these approaches is restricted to a limited number of data instances that can be visualized simultaneously.

We present a technique to visualize time series and trajectories that overcomes these limitations by transforming these data into subspaces which allows data analysts to easily select the instance of interest from a bunch of data. The benefits of our proposed method are threefold: it provides i) a visual representation of time-dependent data in a massive amount simultaneously, ii) a very concise feature representation and iii) an easy identification of anomalies. The results are demonstrated by employing this technique to various data traits from public archives, they are i) univariate time series data from the UCR archive, ii) multivariate time series data from several sources, and iii) human motion trajectories from two motion capture (MoCap) datasets.

**Originally published in:** The 21st Scandinavian Conference on Image Analysis (SCIA), 2019. Springer Lecture Notes in Computer Science (LNCS) series.

**DOI:** [https://doi.org/10.1007/978-3-030-20205-7\\_21](https://doi.org/10.1007/978-3-030-20205-7_21)

# Dimensionality Reduction for Visualization of Time Series and Trajectories

Pattreeya Tanisaro and Gunther Heidemann

Institute of Cognitive Science, University of Osnabrück

**Abstract.** Visualization is essential for data analysis and it is particularly challenging for data in high-dimensional space, especially for temporal information. Many techniques have been employed in an attempt to transform multivariate time series data to one-dimensional data by reducing the number of features in order to visualize their time-dependent behaviors. However, the applicability of these approaches is restricted to a limited number of data instances that can be visualized simultaneously.

We present a technique to visualize time series and trajectories that overcomes these limitations by transforming these data into subspaces which allows data analysts to easily select the instance of interest from a bunch of data. The benefits of our proposed method are threefold: it provides i) a visual representation of time-dependent data in a massive amount simultaneously, ii) a very concise feature representation and iii) an easy identification of anomalies. The results are demonstrated by employing this technique to various data traits from public archives, they are i) univariate time series data from the UCR archive, ii) multivariate time series data from several sources, and iii) human motion trajectories from two motion capture (MoCap) datasets.

**Keywords:** Dimensionality Reduction · Visualization · Time Series

## 1 Introduction

Visualization techniques help us to understand data by observing its patterns. A common approach to meet this challenge is to transform the high-dimensional data representation into lower dimensions. High-dimensional datasets typically have certain traits that can be captured after a transformation to a different coordinate system, and become directly visible to the human eye when mapped to two or three-dimensional space. Although there are numerous dimensionality reduction techniques available; nevertheless, there are much fewer dimensionality reduction approaches that are practical enough for the general application of which take the temporal information into account. Some interesting techniques for visualization of time-dependent data, for example [4,7,9] are tied to a domain-specific application and their data are not publicly accessible; therefore, it raises a question of whether to adopt those techniques to a new domain application even though the data may be categorized in the same time characteristics.

Generally, a known method for visualizing time series data is by using a line graph which is widely used in the monitoring of vital signs in order to detect the

abnormal exceeding of a specified threshold. However, the line graph does not work well for multivariate time series, and it is especially difficult to visualize a large collection of signals. On top of that, for the case of a multivariate time series where the interdependencies occur among many variables, detecting the outliers happens to be much more difficult and cannot be solved by visualizing each data feature on the timeline. A conventional approach to visualize high-dimensional time series data is to examine their distance matrices. One benefit of using a distance matrix such as Euclidean distance is that we can further analyze the matrix using a recurrence plot (RP) by applying a threshold distance and the Heaviside function. The RP is well-known for the visualization of time series because it allows any high-dimensional phase space trajectories to be visualized in subspaces through a two-dimensional representation of its recurrences.

Unlike other traditional dimensionality reduction approaches for temporal data which allow a user to analyze signals over time, **our technique allows the user to inspect the structure of a bunch of time series data or trajectories simultaneously and to detect the outliers**. Particularly for multivariate time series which have many practical usages in real-world applications, there is no general solution to compare a bunch of information at the same time. We demonstrate that it is possible to apply conventional dimensionality reduction approaches from a non-time-series to transform high-dimensional time series into low-dimensional subspaces **by neglecting time information in the display**. Although the dimensionality reduction techniques themselves are not new, to the best of our knowledge, such an attempt to visualize the time-dependent data by neglecting the time axis has never been investigated before. The advantages of our approach are threefold. They allow the data analyst: *i) to visualize the large-scale time series data clusters simultaneously, ii) to get a very concise feature representation of the time series regardless of its length and the number of features, and iii) to detect an anomaly in the data.*

## 2 Data Transformation To Subspaces

Let  $p$  be the total number of instances in the dataset. For any given instance  $i$ , each individual instance is specified by  $\{X^i\}$  where  $i \in \{1, \dots, p\}$ . For any high-dimensional data sequence  $X^i$  with a fixed number of features  $m$  and arbitrary length  $T_i$ , we can be interpreted  $X^i$  as a real-valued matrix  $X$  with a dimension  $m \times T_i$  as illustrated in Figure 1a. Pick the number of selected components  $c_n$  for any transformation  $F$  to the matrix  $X^i$  where  $n$  is the number of dimensionality reduction technique used. For the chosen first principal components  $c_1$  at  $n=1$ , we obtain  $F_1(X^i)$  as illustrated in Figure 1b where  $T_i \geq c_1$ . Hence, to apply  $n$ -times of dimensionality reduction of  $F$  to  $X^i$  for  $c_n$  components, namely  $F_n(F_{n-1}(\dots F_1(X^i)\dots))$  as shown in Figure 1c, requires:

$$T_i \geq c_1 \quad \forall i \in \{1, \dots, p\} \quad \text{and} \quad (m \cdot c_1) \geq c_2 \dots \geq c_n \quad (1)$$

Usually, the sequence length of any signal instance is much larger than the selected number of principal components, that is  $T_i \gg c_1 \forall i \in \{1, \dots, p\}$ . Before

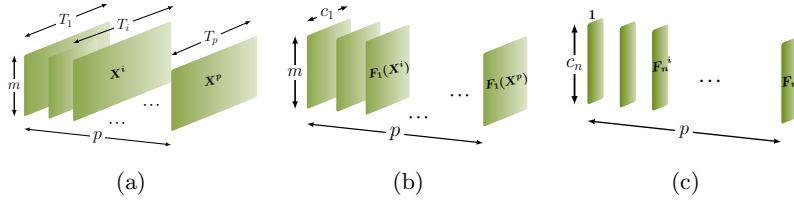


Fig. 1: Transformation of time-dependent data into subspaces. a)  $p$  instances of time-dependent data of  $m$  features with arbitrary sequence lengths  $T_i$ . b) Results after the first transformation of  $F_1$ . From this point onward, the arbitrary size of “time dimension”  $T_i$  has become all equal with the selected principal components  $c_1$ . c) The data after an arbitrary  $n^{\text{th}}$  transformation giving each signal instance of size  $c_n$  which can portray a small feature representation of the data.

applying the first order transformation,  $n = 1$ , we may build a feature vector by normalizing each  $X_{j,k}^i$  where  $j \in \{1, \dots, m\}$  and  $k \in \{1, \dots, T_i\}$  as:

$$X_{j,k}^i \Leftarrow X_{j,k}^i - \bar{X}_j^i \quad (2)$$

where  $\bar{X}_j^i$  is the average over the sequence length  $T_i$  of feature  $j$ . Likewise, for the case of the trajectories of MoCap dataset, we first normalize the stick-figure’s joint positions which were computed by the marker positions following [12] by subtracting the position of the center of the torso from each joint position. The normalization by subtracting the mean is optional but is proved to enhance the visualization in many cases. For the case of different scaling of features, the rescaling prior to applying the dimensionality reduction transformation can be beneficial. However, normalizing time series data by dividing it by its standard deviation does not improve our visualization in general. Similar evidence was reported in [11] for human motion classification. After applying the first transformation of  $F_1$  on each normalized  $X^i$ , the data sequence  $X^i$  can be newly represented as  $F_1(X^i) \in \mathbb{R}^{m \times c_1}$  as depicted in Figure 1b. The time axis now has been replaced with the number of principal components of the first transformation. The feature vector for the second order transformation may be arranged using a concatenation of an average vector to  $F_1(X^i)$  as  $[\bar{X}_j^i; F_1(X^i)]$ . After a second order transformation,  $F_2(F_1(X^i))$ , the new matrix can be shortly written as  $F_2^i \in \mathbb{R}^{1 \times c_2}$  which is depicted in Figure 1c.

### 3 Datasets

The datasets in this paper were selected by considering the number of data classes up to twelve classes which can be easily identified by different colors. There is no restriction on the number of features nor sequence lengths.

### 3.1 Univariate Time Series

The UCR archive [3] contains 85 datasets of univariate time series. It is target for benchmarking time series classification. Each dataset consists of a separate training and test set of a fixed sequence length which was already normalized to have zero mean and a standard deviation of one. We picked five datasets from this archive and merged the training and test data together because we only focus on the visualization of data and not on classification. These datasets are *Plane*, *ItalyPowerDemand*, *Wafer*, *CBF* and *ECG5000*.

### 3.2 Multivariate Time Series

We selected three datasets which are frequently used in benchmarking classification tasks as found in [1, 5, 6]. These three datasets were: *JapaneseVowel*, *NetworkFlow* and *Wafer* (It has the same dataset name but different set from the UCR). The *JapaneseVowel* dataset was a collection of nine male speakers for a total of 640 sequences. Each utterance forms a sequence with lengths in the range of 7-29 and consists of 12 features each. The *Networkflow* dataset represents a network traffic protocol of a total of 1337 sequences with the sequence length of 50-997 where a series of network packets define a sequence. Each packet consists of four attributes which are used to identify the applications that generated the traffic flow. These attributes are a packet size, transfer direction, payload, and the duration. The *Wafer* stands for silicon wafer in semiconductor manufacture. The dataset contains 1194 sequences with the sequence length of 104-198. Each sequence consists of six measurement variables recorded during the etch process. Each wafer is marked as normal or defective.

### 3.3 Motion Capture

We chose two different MoCap datasets, the UTD-MHAD [2] and the HDM05 [8] to demonstrate the effectiveness of our proposed technique.

The **UTD-MHAD** consists of 27 different actions performed by eight subjects. Each action was recorded using 20 markers in 3D coordinates, resulting in the total number of 60 features. Each subject repeated the same actions four times (trials) for only one cycle, and each action trial has different sequence length. Hence, we have only 32 sequences for each action in total. This small amount of data is statistically not interesting. Based on the results of a quasi-view independent of human movement in 3D described in [12], an eigenvector of the largest eigenvalue still maintains its projection size even when a subject performs the same action facing in a different direction. Therefore, we rotated the actor’s default view by 10, 20 and 30 degrees in order to obtain more samples.

The **HDM05** was originally made up of 130 classes consisting of five subjects, called “bd”, “bk”, “dg”, “mm” and “tr”, performing actions with and without repeating the same cycles separately. Following [10], we grouped the non-repetitive and repetitive motions together yielding 65 actions, resulting in a various number of trials in each action. Some actions e.g., *walk* consists of four types of

walk, they are *walk2StepsL*, *walk2StepsR*, *walk4StepsL*, and *walk4StepsR*. In this dataset, each user had more freedom to perform the action, for example, the numbers of repetitions and the directions of movement were not fixed.

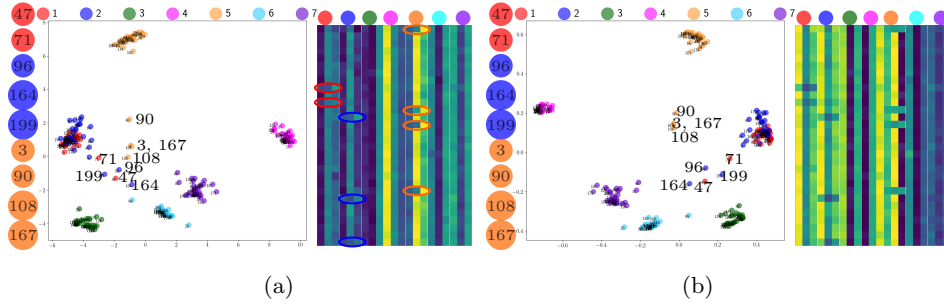


Fig. 2: Results of the dimensionality reduction of 210 sequences of “Plane” using (a) PCA and (b) Kernel PCA with RBF kernel. Nine outliers listed on the left side of each image (a) and (b) can be easily identified in the two-dimensional views (That is only the first two principal components are drawn). The right side of each image shows the feature representation map of the corresponding algorithm from three principal components of a matrix size  $30 \times (3 \cdot 7)$ , where 30 signals of each data class are in the rows and three components of seven classes are in the columns. The irregular patterns in each class found in the feature representation map highlighted in ellipses at (a)-right can be found in the same positions at the (b)-right. The feature vectors of the data class ● and ● as seen in the feature maps are very similar.

## 4 Visualization Results

In this section, we will examine the outputs from projecting data on 2D and 3D space after applying the transformations discussed in section 2. Without prior knowledge of the characteristics of the data, the dimensionality reduction techniques were randomly chosen from two characteristics, linear projections i.e., PCA and nonlinear projections such as kernel PCA with nonlinear kernels and t-SNE. We selected some interesting outputs to be demonstrated here.

### 4.1 Visualization Results of The Univariate Time Series

The results of applying two dimensionality reduction techniques to *Plane* in the UCR archive are displayed in Figure 2a and Figure 2b. Not only do our outputs exhibit to the viewer the intrinsic properties of each data cluster, but they also

6 Pattreeya Tanisaro and Gunther Heidemann

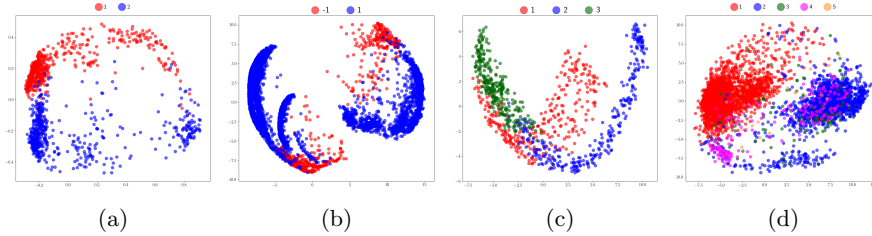


Fig. 3: Results of the dimensionality reduction of selected datasets in the UCR archive. a) 1097 sequences of “ItalyPowerDemand” using Kernel PCA. b) 7174 sequences of “Wafer” using PCA. c) 930 sequences of “CBF” using PCA. d) 5000 sequences of “ECG5000” using PCA. This dataset has much of unbalancing in the data in each class. The number of signals in each class are: (●) 2919, (●) 1767, (●) 96, (●) 194 and (●) 24.

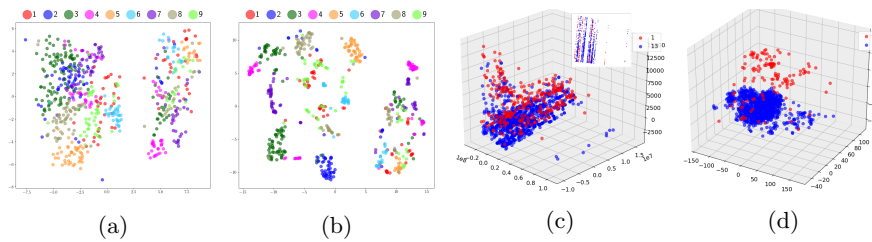


Fig. 4: Results of the dimensionality reduction of multivariate time series. a) “JapaneseVowel” with PCA followed by t-SNE ( $n = 2$ , the perplexity of t-SNE 40). b) “JapaneseVowel” with PCA followed by double t-SNE ( $n = 3$ , the perplexity of t-SNE 40 and 30, respectively). c) “NetworkFlow” using PCA and t-SNE. d) “Wafer” using PCA and t-SNE.

reveal the outliers that laid afar from their groups. It is obvious that using just three principal components as illustrated in Figure 2a and Figure 2b can make a time series much easier to interpret. Even employing different dimensionality reduction techniques, we can easily spot the same outliers from the feature representation maps on Figure 2b-right (no highlight) at the same locations on Figure 2a-right (highlighted with small ellipses). Furthermore, the outputs of four other datasets in the same archive, namely, *ItalyPowerDemand*, *Wafer*, *CBF* and *ECG5000* were illustrated in Figure 3a-3d. As the figures show, several thousands of time series can be displayed simultaneously in the same plane. The unbalanced data, the data clusters, and anomalies can be easily identified.

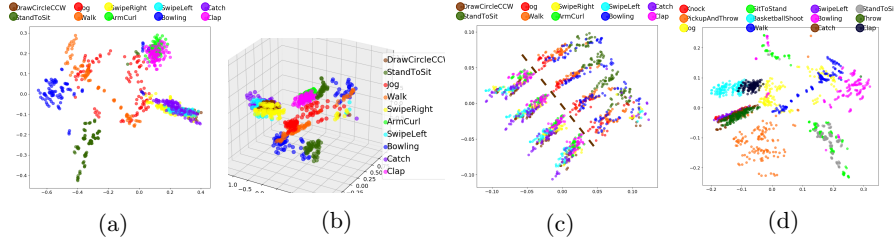


Fig. 5: Results of the dimensionality reduction of human movements in the UTD-MHAD. a) Ten actions using PCA followed by Kernel PCA. b) 3D projection of the same action set as (a) using Kernel PCA and PCA. c) The same action set as (a) using Kernel PCA with two kernels, RBF and polynomial, accordingly. The dashed line was drawn to separate another view. d) Twelve actions using PCA and Kernel PCA.

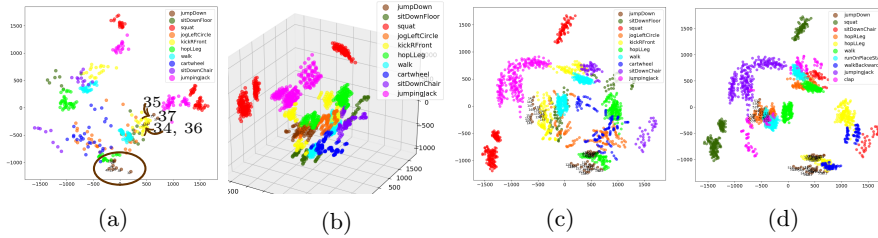


Fig. 6: Results of the dimensionality reduction of human movements in the HDM05. a) 446 number of actions from a default view using Kernel PCA and PCA. Three movements of “jumpDown” (●) of user “dg” (34, 35, 36) and one movement of user “mm” (37) are laid far away from the others (in the brown ellipse). b) Three-dimensional projection of 1784 trials from ten actions as (a) with additional movements of rotating subjects by 10, 20 and 30 degrees. c) Two-dimensional projection of (b). d) A similar effect in a different subset of motions of four viewpoints comparing to (c) depicted in different colors from (d) → (c) as the following: “jumpDown” (●), “squat” (● → ●), “jumpingJack” (● → ●), “hopLLeg” (● → ●), “sitDownChair” (● → ●) and “walk” (● → ●).

## 4.2 Visualization Results of The Multivariate Time Series

For multivariate time series, the result of applying PCA followed by t-SNE, namely using  $n = 2$  to *Japanese Vowel* is displayed in Figure 4a. Figure 4b is the output of adding a second t-SNE ( $n = 3$ ) with smaller perplexity to Figure 4a. Generally, applying just two appropriate transformations should be sufficient to capture an important structure of the data, however in this case we prefer to have more compact clusters. Figure 4c displays two patterns of traffic data flow from the *NetworkFlow*. An inset in the figure shows a two-dimensional projection. The characteristics of the data are very different in large scale, that is the direction



8 Pattreeya Tanisaro and Gunther Heidemann

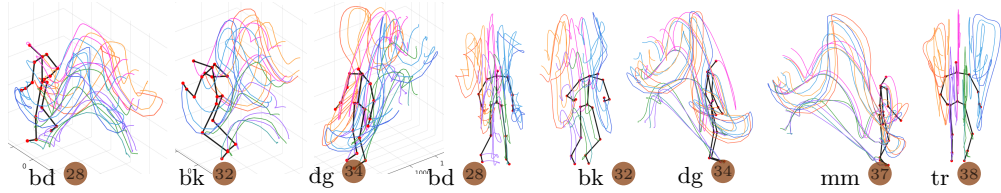


Fig. 7: Five subjects performs “JumpDown” (●) from the default view as seen in Figure 6a. From left to right: the first three images are the 3D projection of three subjects. The next five images are the simplified 2D projections in which the trajectories are much easier to be observed. The subjects “dg” and “mm” face 90° opposing to the direction of “bd”, “bk”, and “tr” causing different patterns of the trajectories and are considered as another group as seen in Figure 6a.

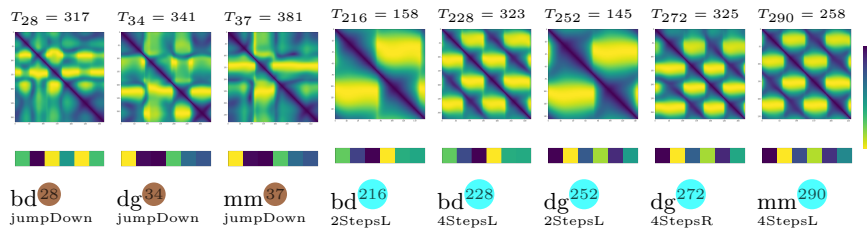


Fig. 8: A comparison of eight trials in the HDM05 between applying the unthresholded RP and our proposed algorithm with six principal components lying below each corresponding RP. The first three images on the left are from the action “jumpDown” (●) as depicted in Figure 7. The next five images are obtained from four types of “walking” (●). Notice the time length ( $T_i$ ) where the repetitions of the walking cycle occur.

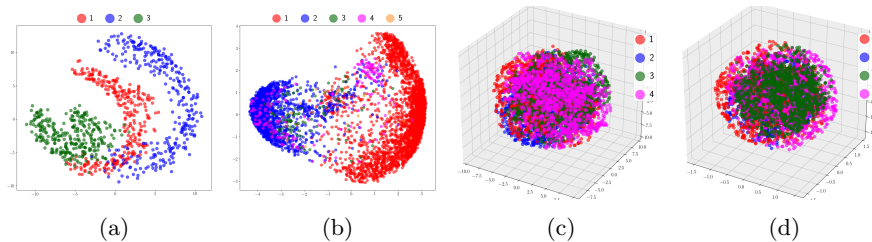


Fig. 9: Variations of transformation. a) “CBF” using MDS. b) “ECG5000” using Kernel PCA with polynomial kernel. c) “TwoPatterns” in the UCR archive using PCA. d) “TwoPatterns” using Kernel PCA with third-degree polynomial.



of the traffic is just either 1 or 0, whereas the payload sizes are about a few thousand of units. Nevertheless, our approach allows a user to easily spot an

outlier. An output of *Wafer* where the wafers were marked either defective (●) or not defect (●) from 1194 signals has been revealed in Figure 4d.

### 4.3 Visualization Results of The Motion Capture

**The UTD-MHAD dataset.** The outputs of applying the proposed method to the UTD-MHAD can be seen in Figure 5a–5d. Ten actions from Figure 5a are *DrawCircleCCW*, *StandToSit*, *Jog*, *Walk*, *SwipeRight*, *ArmCurl*, *SwipeLeft*, *Bowling*, *Catch*, and *Clap*. Four actions which involve only one arm movement, *SwipeLeft*, *SwipeRight*, *Catch*, and *DrawCircleCCW* are drawn very close to each other in two-dimensional space; while two-hand movements, *ArmCurl* and *Clap* plots are also close to one another. The actions concerning moving hands and feet such as *Jog* and *Walk* can be related to each other in the plot. The rest of the actions, *StandToSit* and *Bowling* are obviously distinct from other actions. When the transformation algorithms have been changed, this yields different patterns as found in Figure 5b–5c. It is not easy to interpret the data by simply inspecting Figure 5c without understanding the underlying patterns. There are two viewpoints to be used to interpret this figure: i) two groups of actions are separated by the dashed line and ii) four groups of actions which are perpendicular to the dashed line. The first view divides actions into two groups, one group involving actions with just “hands” moving lies on the left side of the dashed line and another group on the right involves actions in which both, “hands” and “legs” move. From the second viewpoint, the data can be seen as four clusters, as data lies perpendicular to the dashed line which is the consequence of rotating subjects around an axis by viewing the same action from four angles (0, 10, 20, and 30 degrees). As a result, these four groups look quite similar to each other but just shifted along the same axis. Moreover, the results of the transformation of twelve actions are displayed in Figure 5d. We take out three actions, *DrawCircleCCW*, *SwipeRight* and *ArmCurl* and add five actions, *Knock*, *PickupAndThrow*, *SitToStand*, *BasketballShoot*, and *Throw* to the plot. The action *StandToSit* now gets closer to the new action *SitToStand*, whereas the actions about one arm movement *Knock*, *Throw*, *SwipeLeft* and *Catch* lie close to each other. Some data of *Jog* and *Walk* lay close to each other, and the rest of actions such as *BasketballShoot*, *Clap*, *Bowling*, *PickupAndThrow* are properly clustered.

**The HDM05 dataset.** The results of applying similar techniques as to the UTD-MHAD to the **HDM05** can be found in Figure 6a–6d. Figure 6a shows 446 trials from ten actions using the default view (0°) of MoCap. The action *jumpDown* (●) has the minimum number of trials because it consists of only 13 trials, of which four trials are from the subject “bd”, three trials from the subject “dg” and another three trials from the subject “tr”, two trials from the subject “bk” and one trial from the subject “mm”. Notice that four trials of the *jumpDown* from the subject “dg” and “mm”, 34, 35, 36 and 37 lay far away from the others in the brown ellipse. When we looked closely to these particular sequences as depicted in Figure 7, we found that the subjects “dg” and “mm” have completely different trajectories compared to the subjects “bd”, “bk”, and “tr”. This is because these two subjects turn completely by 90 degrees difference

to the camera, or in perpendicular to the other three subjects “bd”, “bk”, and “tr”. This is the reason why those four trials of *jumpDown* are laid afar from the group. Next, when we slightly rotated these five subjects in all trials by 10, 20, and 30 degrees, the patterns of each action in these four viewpoints can be observed in Figure 6b- 6d. Figure 6b shows the first three principal components of these four views from the rotations ( $0^\circ$ ,  $10^\circ$ ,  $20^\circ$ ,  $30^\circ$ ). After exchanging four of ten actions, while six actions are kept as depicted in Figure 6c and Figure 6d, **the shapes and distributions of these actions still remain the same in the subspaces**. Notice the distributions of *jumpDown*, *squat*, *jumpingJack*, *hopLLeg*, *sitDownChair* and *walk* in Figure 6c comparing to Figure 6d. Additionally, eight images in Figure 8 show the comparison of the visualization of *jumpDown* and *walk* between the unthresholded RPs and our feature representations. The RPs are symmetric matrices of size  $T_i \times T_i$  but we fit them to the same image size for a comparison. By comparing our extracted features with six components which are laid below each corresponding RP, we can easily spot two distinct groups of action *jumpDown* and *walk*. The RP of action *jumpDown* from the subject “dg”  has a distinctive pattern other than the subject “mm” , whereas the extracted features from our method for both actions show about the same features which are in accordance to the trajectories illustrated in Figure 7. The action *walk* consists of four types of walking, “2StepsL”, “2StepsR”, “4StepsL”, and “4StepsR”. The number “2” and “4” indicate the walking steps, while the “L” and “R” indicate whether left or right leg starts. By comparing the RPs of *walk* in Figure 8, the repetitive actions such as two steps (2Steps) or four steps (4Steps) by our method yield similar fixed small features. The “4Steps” *walk* in the RP reveals two harmonic oscillations or two complete cycles, while “2Steps” shows one cycle of action. Furthermore, an output from the subject “bk” have a distinct feature but quite close to “bd”, while the features of “dg” and “mm” are about the same. These results comply with Figure 6, Figure 7 and Figure 8. The trivial changes e.g., either left leg (L) or right leg (R) starting first have no significance considering from the involved markers on one leg versus the markers on the whole body. The concept of principal components makes the results robust to noise and the trivial changes.

## 5 Discussion

### 5.1 Stability and Robustness

The current trends of time series and trajectory classification are to use deep networks by fine-tuning millions of parameters to achieve the best output performance. However, it lacks an explanation of why a particular signal fails. Furthermore, some outliers may lead to overfitting of the training data. Yet, our approach can complement this deficiency by offering a concise feature representation which can give data analyst an understanding of the underlining patterns. By projecting the extracted features onto a two- or three-dimensional space, it provides a visual representation for a data analyst to have an overview of a bunch of data simultaneously. The results are robust to small changes as seen in

Figure 6c, Figure 6d and Figure 8. Five images in Figure 8 show similar feature representations for different types of *walk* (and three images of *jumpDown*) from various subjects leading to the same direction. In addition, even though we employed different transformation techniques, we can still spot the same outliers in different visual representations as the results shown from “Plane” dataset.

## 5.2 Limitations and Variations of Transformation

Our proposed technique has a few limitations. First and foremost, the dimensionality reduction methods employed in our experiment were selected from several known techniques based on the assumption of linear subspace embedding and nonlinear manifold learning of the data. Therefore, we are not able to tell which technique is the best choice. For instance, the output of employing Multidimensional Scaling (MDS) on *CBF* (previously PCA was employed) now can be seen in Figure 9a. This new figure shows better separation of three data clusters comparing to Figure 3c. Changing the transformation of *ECG5000* from PCA (depicted in Figure 3d) to Kernel PCA in Figure 9b gives us an interesting alternative viewpoint. Nevertheless, we have had several cases of failure, for instance, *TwoPatterns* in the UCR archive which composes of 5000 sequences. The data clusters cannot be clustered in the way we had expected. The results are shown in Figure 9c-9d. Another failure was also found in case of visualizing the UTD-MHAD dataset when all the actions in a subset involved only hand gestures. This may be because the changes in the movement due to one arm gestures with three corresponding markers (of dimension  $3 \times 3$ ) were considered insignificant when compared with the movement of a whole body consisting of 20 markers (of dimension  $20 \times 3$ ). Considering PCA which we employed Singular Value Decomposition (SVD) for a matrix decomposition, for any given element  $i$ ,  $X^i$  has an arbitrary size of  $m \times T_i$ . We may assume that the best rank  $r$  of the matrix  $X^i$  is the number of crucial time points  $c_1$ . The matrix  $X^i$  is just a product of two matrices  $U$  and  $V$  where  $U$  is an  $m \times c_1$  matrix expressing the weighted factor, and  $V$  is a  $c_1 \times T_i$  matrix. We keep the most highest weighted factors in  $U$  and repeat the process for  $n$  times. Hence, the principal components  $c_n$  are to be in the final results.

## 6 Conclusion

In this paper, we have presented a new approach for time series and trajectory visualization by employing existing well-known non-time-series dimensionality reduction techniques. Our proposed methodology does not seek to make an interpretation of an individual signal nor to inspect the changes of data over time. Nonetheless, we can reveal some meaningful information such as the overview of data clusters. Moreover, outliers of each data class can be easily identified. By integrating this technique into a visual analytic pipeline in visualization tools, it can take the load off a data analyst in order to investigate any anomalies presented in the large data size. The datasets applied in the experiments were

selected from diverse sources to demonstrate and enforce the robustness of our proposed technique. This technique is not tailored to any particular data type, hence it can be integrated into any application domain. In addition, our approach is very simple to implement and lightweight as well as reproducible across different runs. Finally, it is to be noted that good clustering depends on the inter-relationship of the data structure and the correctly applied manifold learning method to achieve the optimum results.

## References

1. Baydogan, M.G., Runger, G.: Time series representation and similarity based on local autopatterns. *Data Mining and Knowledge Discovery* **30**(2), 476–509 (Mar 2016). <https://doi.org/10.1007/s10618-015-0425-y>, <https://doi.org/10.1007/s10618-015-0425-y>
2. Chen, C., Jafari, R., Kehtarnavaz, N.: UTD-MHAD: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor, vol. 2015-December, pp. 168–172. IEEE Computer Society (12 2015). <https://doi.org/10.1109/ICIP.2015.7350781>, <http://www.utdallas.edu/~kehtar/UTD-MHAD.html>
3. Chen, Y., Keogh, E., Hu, B., Begum, N., Bagnall, A., Mueen, A., Batista, G.: The UCR time series classification archive (2015), [http://www.cs.ucr.edu/~eamonn/time\\_series\\_data/](http://www.cs.ucr.edu/~eamonn/time_series_data/)
4. Jäckle, D., Fischer, F., Schreck, T., Keim, D.A.: Temporal mds plots for analysis of multivariate data. *IEEE Transactions on Visualization and Computer Graphics* **22**(1), 141–150 (Jan 2016). <https://doi.org/10.1109/TVCG.2015.2467553>
5. Karim, F., Majumdar, S., Darabi, H., Harford, S.: Multivariate lstm-fcns for time series classification (2018)
6. Luczak, M.: Combining raw and normalized data in multivariate time series classification with dynamic time warping. *Journal of Intelligent and Fuzzy Systems* **34**(1), 373–380 (2018). <https://doi.org/10.3233/JIFS-171393>
7. Martin, S., Quach, T.T.: Interactive visualization of multivariate time series data. In: *Proceedings, Part II, of the 10th International Conference on Foundations of Augmented Cognition: Neuroergonomics and Operational Neuroscience - Volume 9744*, pp. 322–332. Springer-Verlag New York, Inc., New York, NY, USA (2016)
8. Müller, M., Röder, T., Clausen, M., Eberhardt, B., Krüger, B., Weber, A.: Documentation mocap database hdm05. Tech. Rep. CG-2007-2, Universität Bonn (June 2007)
9. Nguyen, M., Purushotham, S., To, H., Shahabi, C.: m-tsne: A framework for visualizing high-dimensional multivariate time series. *CoRR* **abs/1708.07942** (2017)
10. Tanisaro, P., Heidemann, G.: An empirical study on bidirectional recurrent neural networks for human motion recognition. In: *The 25th International Symposium on Temporal Representation and Reasoning (TIME)* (2018)
11. Tanisaro, P., Lehman, C., Sütfeld, L., Pripa, G., Heidemann, G.: Classifying bio-inspired model in point-light human motion using echo state network. In: *The 26th International Conference on Artificial Neural Networks (ICANN)*, 2017 (2017)
12. Tanisaro, P., Mahner, F., Heidemann, G.: Quasi view-independent human motion recognition in subspaces. In: *Proceedings of 9th International Conference on Machine Learning and Computing (ICMLC)* (2017), feb 24-26, 2017, Singapore

88 2.6 • Dimensionality Reduction for Visualization of Time Series and Trajectories

## 2.7 A Very Concise Feature Representation For Time Series Classification Understanding

**Abstract:** One major problem of time series analysis, particularly of a multivariate time series, is to find their feature representations. Especially, with the emerging of deep recurrent neural networks (RNNs), researchers opt to train the networks with raw signals by using an end-to-end framework to achieve the highest classification accuracy. Their works focus on modifying the network models and fine-tuning millions of hyperparameters; however, they lack the required level of understanding of the intrinsic properties of the data. In our work, we adopted a technique for dimensionality reduction of non-time-series to transform the time series data into small sets of feature representations. Our proposed technique allows the analyst to easily visualize the feature representations of the data and detect an instance which has the potential to cause a test failure. We demonstrated the robustness of our technique by subjecting the extracted features to a conventional classification approach such as Random Forest. The datasets used for the evaluation are from the known benchmarking of 15 multivariate time series datasets and two Motion Caption datasets of 27 and 65 actions. The classification results were compared with the outputs from the Echo State Networks (ESNs) and the deep Bidirectional Neural Networks (BRNNs).

**Originally published in:** The 16th International Conference on Machine Vision Applications (MVA), 2019. IAPR

**DOI:** <https://doi.org/10.23919/MVA.2019.8757981>

# A very concise feature representation for time series classification understanding

Pattreeya Tanisaro  
University of Osnabrück, Germany

Gunther Heidemann  
University of Osnabrück, Germany

## Abstract

*One major problem of time series analysis, particularly of a multivariate time series, is to find their feature representations. Especially, with the emerging of deep recurrent neural networks (RNNs), researchers opt to train the networks with raw signals by using an end-to-end framework to achieve the highest classification accuracy. Their works focus on modifying the network models and fine-tuning millions of hyperparameters; however, they lack the required level of understanding of the intrinsic properties of the data. In our work, we adopted a technique for dimensionality reduction of non-time-series to transform the time series data into small sets of feature representations. Our proposed technique allows the analyst to easily visualize the feature representations of the data and detect an instance which has the potential to cause a test failure. We demonstrated the robustness of our technique by subjecting the extracted features to a conventional classification approach such as Random Forest. The datasets used for the evaluation of this task are from the known benchmarking of 15 multivariate time series datasets and two Motion Caption datasets of 27 and 65 actions. The classification results were compared with the outputs from the Echo State Networks (ESNs) and the deep Bidirectional Neural Networks (BRNNs).*

## 1 Introduction

With the emergence of various deep neural network frameworks, the classification of time series has become more efficient than ever. However, a challenge in time series classification is to find a data representation which can be interpreted or explained to an audience when the test fails. For time series analysis this is a key issue which allows an analyst to detect the anomalies instead of obscuring them by permitting the models to overfit the data. A general approach to visualize a time series is to employ a line graph. However, the line graph does not work well for multivariate time series where inter-dependencies between many variables exist. A much more complex situation occurs if the data instances are not of equal length for many of the data features in a large dataset. There is only one known technique that lets a data analyst inspect the feature representation of the multivariate time series, namely the unthresholded recurrence plots (RPs) [4]. The RP or distance plot is heavily used for the visual-

ization of time series because it allows any high dimensional phase space trajectories to be visualized in subspaces through a two-dimensional representation. It exhibits reoccurring phase space trajectories of dynamic systems. This technique has been employed as an action descriptor for view-independent action recognition in combination with the Bag-of-Features obtained from the Histogram of Oriented Gradients (HOG) [5]. Nevertheless, the downside of this approach is that the lengths of all motion recordings in the experiment must be truncated to an equal unit length in order to get the fixed window size. Therefore, it is not suitable for data with unequal lengths. In addition, RPs cannot be viewed together in the same coordinate system and it requires a lot of work to examine each data sequence individually.

In our work, we demonstrate a representation of a time-dependent data to be captured in a lower dimensional space where it can be understood by a traditional classification approach such as Random Forest (RF), and is easily perceived by a data analyst. *Although the applied dimensionality reduction techniques themselves are not new, however to the best of our knowledge, there was no attempt to express the temporal information in a way which allowed the time series to be inspected simultaneously in the normal Cartesian coordinate system.* The classification outputs are tested on two kinds of datasets i) general multivariate time series of 15 datasets and ii) motion capture (MoCap) of two datasets for action recognition used in [11]. Since the evaluation of this tasks should focus on the generalization of action recognition, therefore we exclude the test subjects from the group of training subjects. The other existing works, for instance, [3, 14] did not condition on separation of the subjects in their experiments; hence, we implemented two types of RNNs, a reservoir computing RNN: the ESN, and a gradient-based RNN: the BRNN, for the comparison.

## 2 Dimensionality Reduction of Time Series

Let  $p$  be the total number of data instances in the dataset, and for any given data instance  $i$ , the set of individual data sequences is specified by  $\{X^i\}$  where  $i \in \{1, \dots, p\}$ . For any high-dimensional data sequence  $X^i$  with a fixed number of features  $m$  and arbitrary length  $T_i$ , we can interpret  $X^i$  as a real-valued matrix  $X$  with a dimension  $m \times T_i$  as illustrated in Figure 1a. Pick the number of selected components  $c_n$  for



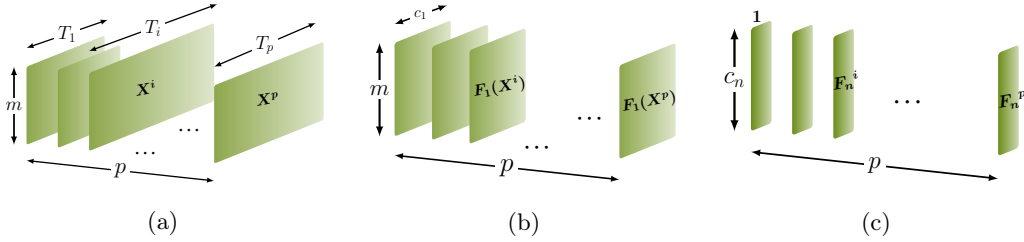


Figure 1: Transformation of time-dependent data into subspaces. (a)  $p$  instances of time series of  $m$  features with arbitrary sequence lengths  $T_i$ . (b) Results after the first transformation of  $F_1$ . From this point onward, the arbitrary size of “time dimension”  $T_i$  has become all equal with the selected principal components  $c_1$ . This new feature representation of  $X^i$  can be understood by **the conventional classification approach**. (c) The data after an arbitrary  $n^{\text{th}}$  transformation giving each signal instance of size  $c_n$  which can portray a small feature representation and its first two or three components can be projected onto the XY- or XYZ-plane.

any manifold learning  $F$  to matrix  $X^i$  where  $n$  is the number of manifold learning technique used. For the chosen first principal components  $c_1$  at  $n=1$ , we obtain  $F_1(X^i)$  as illustrated in Figure 1b where  $\{T_i\} \geq c_1$ . Hence, to apply  $n$ -times of dimension reduction of  $F$  to  $X^i$  for  $c_n$  components, namely  $F_n(F_{n-1}(\dots F_1(X^i)\dots))$  as shown in Figure 1c, requires:

$$T_i \geq c_1 \forall i \in \{1, \dots, p\} \text{ and } (m \cdot c_1) \geq c_2 \dots \geq c_n \quad (1)$$

Usually, the sequence length of any signal instance is much larger than the selected number of principal components, that is  $T_i \gg c_1 \forall i \in \{1, \dots, p\}$ . Before applying the first order transformation,  $n = 1$ , we may build a feature vector by normalizing each  $X_{j,k}^i$  where  $j \in \{1, \dots, m\}$  and  $k \in \{1, \dots, T_i\}$  as:

$$X_{j,k}^i \leftarrow X_{j,k}^i - \bar{X}_j^i \quad (2)$$

where  $\bar{X}_j^i$  is the average over the sequence length  $T_i$  of feature  $j$ . Likewise, for the case of the trajectories of MoCap dataset, we first normalize the skeleton’s joint positions which were computed by the marker positions following [13] by subtracting from each joint position the position of the center of the torso. The normalization by subtracting the mean is optional but is proved to enhance the visualization in many cases. For the case of different scaling of features, the rescaling prior to applying the manifold learning can be beneficial. However, normalizing time series data by dividing it by its standard deviation does not improve our visualization in general. Similar evidence was reported in [12] for human motion classification. After applying the first transformation of  $F_1$  on each normalized  $X^i$ , the data sequence  $X^i$  can be newly represented as  $F_1(X^i) \in \mathbb{R}^{m \times c_1}$  as depicted in Figure 1b. The time axis now has been replaced with the number of principal components of the first transformation. The feature vector for the second order transformation may be arranged using a concatenation of an average vector

to  $F_1(X^i)$  as  $[\bar{X}_j^i; F_1(X^i)]$ . After a second order transformation,  $F_2(F_1(X^i))$ , the new matrix can be shortly written as  $F_2^i \in \mathbb{R}^{1 \times c_2}$  which is depicted in Figure 1c.

### 3 Experimental Setup

#### 3.1 Multivariate Time Series Datasets.

For our experiment, we took fifteen datasets of the multivariate time series collected in [1]. These datasets were used to benchmark the classification methods in [1, 9, 7, 6]. The characteristics of each dataset are shown in Table 1 are i) the number of the attributes, ii) the lengths of sequences in the dataset, iii) the number of output classes, iv) the number of training data and v) the number of testing data. We grouped these datasets into four categories according to the levels of difficulty based on the classification results from DTW [1] which is the state-of-the-art approach for time series classification. These four levels are i) *very difficult* to solve datasets which have an error rate greater than 20%. These datasets are indicated with the deep blue squares (■) in front of the dataset names. ii) *difficult datasets* are marked with orange circles (●), which have an error rate in the range of (10-20)%. iii) *normal difficulty* with the error rates in the range of (5-10)%. The datasets are marked with the green triangle (▲), and iv) *easy datasets* indicated with pink diamond (◆). They have the error rates not higher than 5%.

#### 3.2 Motion Capture Datasets.

We extended our experiment by selecting two different MoCap datasets, the UTD-MHAD [2] and the HDM05 [8] to demonstrate the effectiveness of our proposed technique. The test subjects were excluded from the training set to examine the generalization of action recognition.

**UTD-MHAD** consists of 27 different actions performed by eight subjects. Each subject repeated the

same action four times. The dataset contains a total of 861 trials or data instances, where three sequences were corrupted and removed from the dataset on the official website. The training was performed on six subjects, and two subjects were left out for the test. The recognition rate was reported on an average of 28 combinations. This dataset was recorded using 20 markers.

**HDM05** was originally made up of 130 classes consisting of five subjects performing actions with and without repeating the same cycles separately. This created a total of 2343 instances. We followed [3, 14, 11] in grouping non-repetitive and repetitive motions together yielding 65 actions. This dataset has been heavily biased on some actions and their lengths. For example, *walk* and *elbowToKnee* contain 94 and 80 trials, respectively; while for about 20 actions have less than 20 trials, i.e., *throwBasketball*, *throwFarR* and *jump-Down* having only 14 trials each. Nonetheless, since we focused on the action recognition of unseen subjects, four subjects were used in the training set and one subject was used for the test. We reduced the original number of markers to 19, where some nearby sensors e.g., on the spine were merged.

### 3.3 Configurations of the Classifiers

We employed two known linear and non-linear manifold learning such as PCA and Kernel PCA in order to get the features  $F_1(X^i)$  in combination with RF with 50 and 100 trees for classification. Our proposed approach is abbreviated as *DRe* in the results. To display all data simultaneously in two- and three-dimensional projections, we select various manifold learning algorithms  $F_2$  for the best visualization. For the setup of RNNs, we adopted many configurations and took the one with best output performance. The ESN configurations in this experiment followed the guideline in [10]. The number of neurons was varied in the range of 200-600 neurons with 10, 30 and 50% sparsity. We also applied the spectral radii of 1.0 and 5.0, a leaky rate of 0.1 and 0.9 and a fixed regularization coefficient of 0.1. For BRNNs, we created more than one hundred configurations with various depths and widths of BRNNs and picked the best models shown in Table 1. Following the setup of deep BRNN geometries in [11], we varied the size of the networks in the range of  $2 \times \{200 - 600\}$  neurons. The model with one layer of BRNNs with  $2 \times 500$  neurons is presented as  $BR^{500}$ , where 500 indicates the number of neurons in one direction. Therefore, for the two hidden layers of BRNNs with 100 neurons in one direction for each layer is written shortly as  $BR^{2L-100}$  and so on. For three hidden layers of BRNNs, we simply took the best outcome from several configurations and referred to it as  $BR3L$ . Furthermore, we experimented using both GRUs and LSTMs as neurons.

## 4 Experimental Results and Discussion

### 4.1 Multivariate Time Series

Table 1 shows the error rates of 15 datasets from ten classifiers. The results from left to right are, *DTW* and *LPS* taken from [1] and our implementation as the following: *ESN*, BRNNs with one hidden layer ( $BR1L$ ), four strategies of BRNNs with two hidden layers ( $BR^{2L-100}$ ,  $BR^{2L-150}$ ,  $BR^{2L-250}$ , and  $BR^{2L-500}$ ), BRNNs with three hidden layers ( $BR3L$ ), and our proposed method (*DRe*). Next to the results of *DRe* appears one of the two symbols,  $\heartsuit$  and  $\spadesuit$ , to indicate whether it makes sense ( $\heartsuit$ ) to apply the dimensionality reduction to the dataset. We prefer ( $\heartsuit$ ) the *DRe* when the following two conditions meet: i) its error rate is lower than two third among *DTW*, *LPS* and *ESN*, and ii) its error rate is less than 10%. The *LPS* generally performs much better than *DTW* and has two outstanding results which are difficult to be solved by other classifiers; they are *Libras* and *UwaveMTS*. The ESN also gives satisfactory results for most datasets and becomes the winner for *CharTrajectories* and *JapaneseVowels*; nonetheless, it has a problem to classify three datasets marked with  $\bullet$ . The  $BRNN1L$  is the winner for most datasets. Interestingly, however, all BRNNs perform worst on the *NetworkFlow*. This is probably caused by the characteristics of the attributes. The *NetworkFlow* represents a network traffic protocol where a series of network packets defines a sequence. Each packet consists of four attributes which are used to identify the applications that generated the traffic flow. These attributes are the packet size, transfer direction (either 0 or 1), payload and the duration. Whereas the payload and packet size are in the magnitude of a few thousands but the direction can be either 0 or 1. Therefore, this might cause a problem for the gradient computation. Furthermore, it is important to note that the  $BR1L$  performance is better than of the deep BRNNs for most datasets here.

For *DRe*, five datasets are considered unsuited ( $\spadesuit$ ) to be processed by *DRe* method. This is due to: i) the transformation of the matrix of  $m \times T_i$  to  $m \times c_1$  which is constrained by  $T_i$  and  $m_i$ , and ii) the characteristic of all data classes which should be captured by  $m \times c_1$ . That is  $m \times c_1$  should be sparse to be captured by RF or specifically ‘‘sufficiently greater’’ than the number of classes. These two restrictions can be explained by the characteristics of datasets, which are: i) restricted by  $T_i$ , i.e., *ArabicDigits* which has the shortest length of 4 yielding a matrix of maximum size  $13 \times 4$  for classifying 10 classes, or ii) restricted by  $m$  if the dataset has a small number of attributes but requires many output classes, for instance, *CharTrajectories* which has 3 attributes for 20 classes, *EGG* which has 2 attributes for 2 classes, *Libras* which has 2 attributes for 15 classes and *UwaveMTS* which has 3 attributes for 8 classes. Hence, we can easily notice that the pleasing results

Dataset	nAttr	Length	nClass	nTrain	nTest	DTW	LPS	ESN	BR1L	BR <sup>2L-100</sup>	BR <sup>2L-150</sup>	BR <sup>2L-250</sup>	BR <sup>2L-500</sup>	BR3L	DRe
▲ArabicDigits	13	4-93	10	6600	2200	0.092	0.029	0.070	0.003	0.010	0.016	0.009	0.010	0.007	0.244 ★
■AUSLAN	22	45-136	95	1140	1425	0.238	0.246	0.094	0.061	0.178	0.109	0.061	0.060	0.095	0.096 ♥
◆CharTrajectories	3	60-182	20	300	2558	0.033	0.035	0.023	0.033	0.043	0.046	0.045	0.042	0.048	0.186 ★
▲CMUsubject16	62	127-580	2	29	29	0.069	0.000	0.000	0.000	0.172	0.379	0.034	0.483	0.000	0.000 ♥
▲DigitsShape	2	30-98	4	24	16	0.069	0.000	0.000	0.000	0.172	0.379	0.034	0.483	0.000	0.000 ♥
●ECG	2	39-152	2	100	100	0.150	0.180	0.270	0.160	0.210	0.200	0.200	0.160	0.210	0.250 ★
■JapaneseVowels	12	7-29	9	270	370	0.351	0.049	0.011	0.016	0.024	0.032	0.041	0.027	0.022	0.043 ★
◆KickvsPunch	62	274-841	2	16	10	0.100	0.100	0.100	0.200	0.500	0.500	0.500	0.500	0.400	0.000 ♥
●Libras	2	45	15	180	180	0.200	0.097	0.206	0.156	0.328	0.272	0.322	0.250	0.256	0.617 ★
■NetworkFlow	4	50-997	2	803	534	0.288	0.032	0.034	0.779	0.779	0.779	0.779	0.779	0.779	0.028 ♥
●PEMS	963	144	7	267	173	0.168	0.156	0.278	0.058	0.191	0.185	0.260	0.231	0.202	0.092 ♥
◆Shapes	2	52-98	3	18	12	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000 ♥
▲UwaveMTS	3	315	8	896	3582	0.071	0.020	0.089	0.039	0.155	0.133	0.051	0.041	0.044	0.551 ★
◆Wafer	6	104-198	2	298	896	-0.040	0.038	0.028	0.035	0.077	0.085	0.057	0.106	0.079	0.020 ♥
◆WalkvsRun	62	128-1918	2	28	16	0.000	0.000	0.000	0.000	0.000	0.188	0.000	0.188	0.000	0.000 ♥

Table 1: The characteristics of 15 datasets and their error rates in ten classifiers.

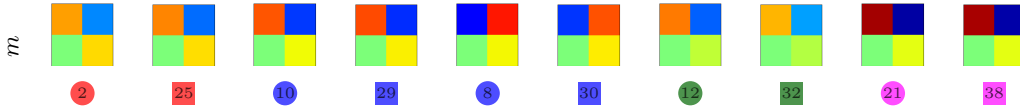


Figure 2: Feature vectors of ten instances of the *DigitsShape* dataset which have two attributes using PCA with two principal components for the classification. Below each feature is the label of the corresponding instance indicated with the number. Each labeled color matches each data class in Figure 3. The training data is presented using circles (●) while the test data is presented using squares (■).

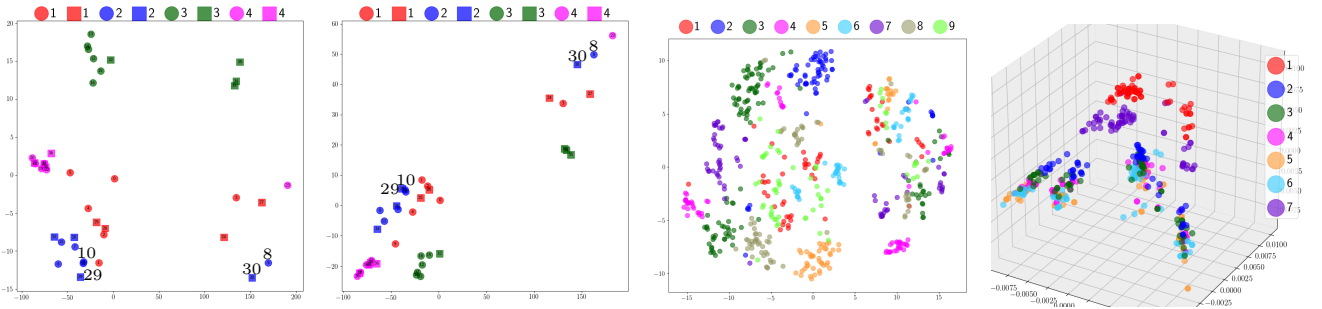


Figure 3: The two left images show the second order transformations of *DigitsShape* using only the first two principal components projected on the two-dimensional plane. The first manifold learning for classification is the PCA followed by two different second transformation approaches “for visualization in 2D projection” using PCA, and MDS, respectively. Four labels, 8, 10, 29, 30 of class “2” (■/●) are enlarged to verify the conformity of their feature representation as shown in Figure 2. The next two images are the projections of 640 instances (of training and test data) of *JapaneseVowels* and 440 instances of *PEMS* (of 963 attributes for each instance).

of *DRe* are obtained for datasets which have many attributes without restriction on the sequence lengths as we can see the outstanding results from *Wafer*, *PEMS* and *NetworkFlow*. Most important, the main benefit of this approach is that we can extract a concise feature which is understandable by the data analyst. Figure 2 shows the feature presentations of ten instances from the *DigitsShape* of 2 attributes to classify 4 classes. Figure 3 shows visualizations on the Cartesian coordinates of *DigitsShape*, *JapaneasesVowels* and *PEMS*. Moreover, we further investigate the three-dimensional projection of 1194 instances of *Wafer* as displayed in Figure 4.

Ten corresponding feature representations of the *Wafer* dataset are depicted next to its 2D projection on the rightmost. The *Wafer* refers to a silicon wafer in a semiconductor manufacture. Each instance consists of six variables recorded during the etching process and is marked as normal (●) or defective (●).

## 4.2 Motion Capture

The error rates of two MoCap datasets of 27 actions of the MHAD and 65 actions of the HDM05 using PCA for the transformation together with RF can be found in Table 2. The outputs from BRNNs were ob-

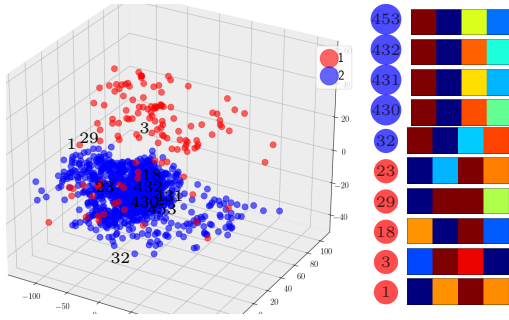


Figure 4: The visualization of projecting the features into 3D space. The image on the left shows 1194 instances of *Wafer*. The feature representations of ten instances of *Wafer* are displayed on the right. The numbers in front of the feature representations are the instance’s ids for tracking them in the 3D space.

Dataset	ESN	BR1L	BR <sup>2L</sup> -100	BR <sup>2L</sup> -250	BR <sup>2L</sup> -300	BR <sup>3L</sup> -200	DRe
MHAD27	0.167	0.100	0.213	0.086	0.127	0.162	0.196
HDM05	0.410	0.254	0.254	0.185	0.187	0.207	0.308

Table 2: The error rates of 27 actions in the MHAD and 65 actions in the HDM05.

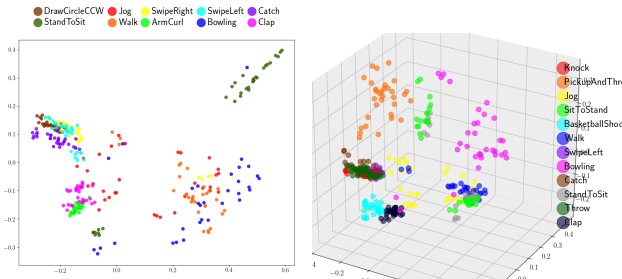


Figure 5: Selected actions of MHAD for visualization. Left image is the result of a 2D projection of ten selected actions and the right image is the result of 3D projection of twelve actions.

tained by fine-tuning hyperparameters for more than one hundred configurations. Although our approach is not better than the BRNNs in general, its performance is comparable. In addition, it took about 10 minutes to complete 28 folds of training and testing the MHAD27, 45 minutes on the ESNs with 500 neurons on Intel i7-3770 CPU 3.40GHz with 16 GB RAM running on one core, while it took three and a half hours on Intel Xeon 3.70GHz 64GB RAM with GeForce GTX TITAN X for one hidden layer of BRNNs of  $2 \times 500$  neurons. The results of *DRe* are reproducible and there is no need to worry about fine-tuning the parameters.

Figure 5 shows the projections of two small subsets of the MHAD. For the image on the left, the ac-

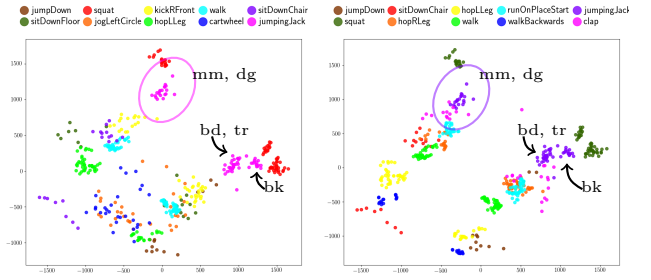


Figure 6: Two-dimensional projections of selected actions in the HDM05. When the actions have changed between the two images on the left to the right, the distributions of the same actions remain the same. These actions are *jumpDown* (●), *squat* (● → ●), *hopLLeg* (● → ●), *walk* (● → ●), *sitDownChair* (● → ●), and *jumpingJack* (● → ●).

tions involve the movements with just an arm, such as *SwipeRight* (●), *SwipeLeft* (●), *Catch* (●) and *DrawCircleCCW* (●) are plotted close to each other. While the actions which engage the movements of arms and legs, for instance, *Jog* (●) and *Walk* (●) are drawn pretty close, the action *StandToSit* (●) and *Bowling* (●) are drawn afar from the other actions. The image on the right shows twelve actions in the 3D space. Similar to the left image, the actions which engage only the movements of one arm such as *Catch* (●), *Knock* (●), *SwipeLeft* (●), and *Throw* (●) are drawn very close to each other. The actions which are sparsely distributed in the plot such as *Bowling* (●), *Jog* (●) and *PickAndThrow* (●) are the actions which have more free movement in space. The *SitToStand* (●) and *StandToSit* (●) are located near each other.

Two images in Figure 6 show the distributions of the same actions on two different subsets (the image on the left vs. the right). Notice the actions which are separated into two groups. This is because two of the subjects “mm” and “dg” are positioned in perpendicular direction to the opposing three subjects “bd”, “bk” and “tr”. Figure 7 can explain why we have such behavior in the plot. From a camera viewpoint, posing actions to 0 and 90 degree creates different trajectories while keeping the correlations of the joints of that movement. The unthresholded RPs of *jumpingJack* and *walking* can be found in Figure 8 on the left and our extracted features are illustrated on the right. The feature representation from our proposed algorithm show that the repetition of the patterns still gives the same fixed concise feature. Moreover, the proposed algorithm is robust against noise as displayed for action indexing [302](#), while the unthresholded RPs shows its sensitivity to this small noise.

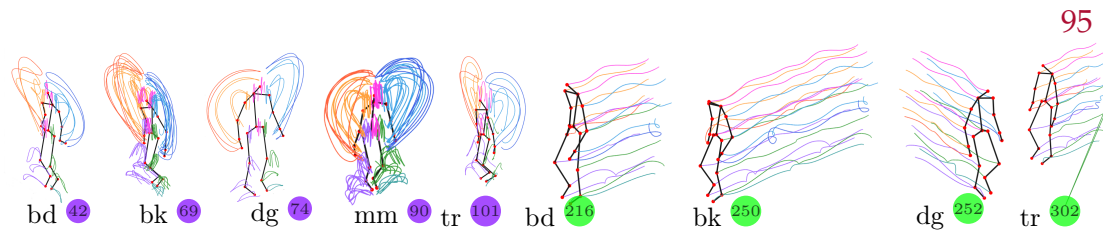


Figure 7: Nine trials of subjects “bd”, “bk”, “dg”, “mm” and “tr” in the HDM05 performing *jumpingJack* and *walk*

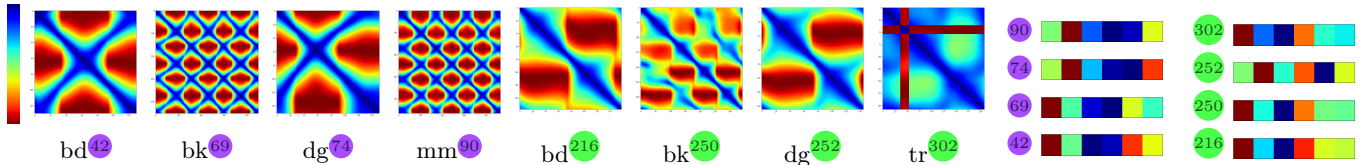


Figure 8: The unthresholded RPs of actions in Figure 7 versus our representations using six components.

## 5 Conclusion

We have presented an approach to represent a time series as a small set of features. We adopted conventional dimensionality reduction techniques such as PCA and KernelPCA to capture the intrinsic properties of the signals. The robustness of our approach has been proven by employing a traditional classifier with these representations. The results were compared using original signals with two types of RNNs with hundreds of configurations. The main benefit of our approach is that regardless of their lengths and the number of features, the time series can be represented in a very concise manner. Furthermore, we can visualize a large amount of time series data simultaneously in a Cartesian coordinate system. An instance which has a unique property would be laid afar from its group. Although our approach has a few limitations, nevertheless its strength lies in the fact that it is very simple to implement and lightweight because the transformation operation is just a matrix decomposition.

## References

- [1] Baydogan, M.G., Runger, G.: Time series representation and similarity based on local autopatterns. *Data Mining and Knowledge Discovery* **30**(2), 476–509 (2016)
- [2] Chen, C., Jafari, R., Kehtarnavaz, N.: Utd-mhad: A multimodal dataset for human action recognition utilizing a depth camera and a wearable inertial sensor. In: *Proceedings of IEEE International Conference on Image Processing*. pp. 168–172 (2015)
- [3] Du, Y., Wang, W., Wang, L.: Hierarchical recurrent neural network for skeleton based action recognition. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015)
- [4] Eckmann, J.P., Kamphorst, O.S., Ruelle, D.: Recurrence plots of dynamical systems. *Europhysics Letters* **4** (Nov 1987)
- [5] Junejo, I.N., Dexter, E., Laptev, I., Perez, P.: View-independent action recognition from temporal self-similarities. *IEEE Trans. Pattern Anal. Mach. Intell.* **33**(1), 172–185 (2011)
- [6] Karim, F., Majumdar, S., Darabi, H., Harford, S.: Multivariate lstm-fns for time series classification (2018)
- [7] Luczak, M.: Combining raw and normalized data in multivariate time series classification with dynamic time warping. *Journal of Intelligent and Fuzzy Systems* **34**(1), 373–380 (2018). <https://doi.org/10.3233/JIFS-171393>
- [8] Müller, M., Röder, T., Clausen, M., Eberhardt, B., Krüger, B., Weber, A.: Documentation mocap database hdm05. Tech. Rep. CG-2007-2, Universität Bonn (2007)
- [9] Schäfer, P., Leser, U.: Multivariate time series classification with WEASEL+MUSE. CoRR **abs/1711.11343** (2017), <http://arxiv.org/abs/1711.11343>
- [10] Tanisaro, P., Heidemann, G.: Time series classification using time warping invariant echo state networks. In: *15th IEEE International Conference on Machine Learning and Applications, (ICMLA)* (2016)
- [11] Tanisaro, P., Heidemann, G.: An empirical study on bidirectional recurrent neural networks for human motion recognition. In: *25th International Symposium on Temporal Representation and Reasoning* (2018)
- [12] Tanisaro, P., Lehman, C., Sütfield, L., Pripa, G., Heidemann, G.: Classifying bio-inspired model in point-light human motion using echo state network. In: *The 26th International Conference on Artificial Neural Networks (ICANN), 2017* (2017)
- [13] Tanisaro, P., Mahner, F., Heidemann, G.: Quasi view-independent human motion recognition in subspaces. In: *Proceedings of 9th International Conference on Machine Learning and Computing (ICMLC)* (2017)
- [14] Zhu, W., Lan, C., Xing, J., Zeng, W., Li, Y., Shen, L., Xie, X.: Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In: *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*. pp. 3697–3703 (2016)

96 2.7 • A Very Concise Feature Representation For Time Series Classification  
Understanding

These sounds that we can't hear, this light that we can't see, how do we even know about these things in the first place? Well, we built tools. We built tools that adapt these things that are outside of our senses, to our human bodies, our human senses. We can't hear ultrasonic sound, but you hook a microphone up to an oscilloscope and there it is. You're seeing that sound with your plain old monkey eyes. We can't see cells and we can't see galaxies, but we build microscopes and telescopes and these tools adapt the world to our human bodies, to our human senses. When Hamming<sup>1</sup> says there could be unthinkable thoughts, we have to take that as "Yes, but we build tools that adapt these unthinkable thoughts to the way that our minds work and allow us to think these thoughts that were previously unthinkable."

Bret Victor, *Media for Thinking the Unthinkable*

---

<sup>1</sup> Richard Hamming, a notable American mathematician