Hybrid Deep-Readout Echo State Network and Support Vector Machine with Feature Selection for Human Activity Recognition

Shadi Abudalfa[1](✉) and Kevin Bouchard[2]

[1] IT Department, University College of Applied Sciences, Gaza, Palestine
sabudalfa@ucas.edu.ps
[2] LIARAlab, Université du Québec à Chicoutimi, Saguenay, Canada
kevin_bouchard@uqac.ca

Abstract. Developing sophisticated automated systems for assisting numerous humans such as patients and elder people is a promising future direction. Such smart systems are based on recognizing Activities of Daily Living (ADLs) for providing a suitable decision. Activity recognition systems are currently employed in developing many smart technologies (e.g., smart mobile phone) and their uses have been increased dramatically with availability of Internet of Things (IoT) technology. Numerous machine learning techniques are presented in literature for improving performance of activity recognition. Whereas, some techniques have not been sufficiently exploited with this research direction. In this paper, we shed the light on this issue by presenting a technique based on employing Echo State Network (ESN) for human activity recognition. The presented technique is based on combining ESN with Support Vector Machine (SVM) for improving performance of activity recognition. We also applied feature selection method to the collected data to decrease time complexity and increase the performance. Many experiments are conducted in this work to evaluate performance of the presented technique with human activity recognition. Experiment results have shown that the presented technique provides remarkable performance.

Keywords: Hybrid technique · Echo state networks · Support Vector Machine · Feature selection · Human activity recognition · Smart system

1 Introduction

Human activity recognition is an important task in developing many smart technologies (e.g., smart homes). Human activity recognition can be used by numerous systems and applications that serve many fields such as healthcare [1], security [2] and physical rehabilitation [3]. Mainly, this task is achieved by employing machine learning techniques [4]. Thereby, using recent machine learning techniques reveal a significant improvement in human activity recognition. This motivated us to explore performance of employing more recent machine techniques with human activity recognition.

Z. Deze et al. (Eds.): BDTA 2020/WiCON 2020, LNICST 371, pp. 150–167, 2021.
https://doi.org/10.1007/978-3-030-72802-1_11

Based on our knowledge, there is an extension of neural networks entitled reservoir computing framework [5] has not been adequately exploited with human activity recognition. Therefore, we present a technique based on such machine learning technique and show its efficiency with this research direction. The presented technique entitled Hybrid Echo State Network and Support Vector Machine (HESNSVM).

The presented technique includes specifically a part of reservoir computing framework which is referred to as Echo State Network (ESN) [6]. ESN [7] mimics a conceptual and practical distinction between a dynamical component entitled reservoir and a feed-forward readout tool. ESN is based on training a deep neural network (recurrent neural network) which causes a high computational time. Thus, we use feature selection [8] for decreasing time complexity. Moreover, the feature selection step improves the overall performance of human activity recognition with the collected dataset used for this work.

Additionally, the presented technique combines ESN with Support Vector Machine (SVM) [9]. As a result of this, the presented technique improves performance of human activity recognition in comparison with ESN. Numerous experiments are conducted in this research to show performance of the presented technique with human activity recognition. Performance of the presented technique has been compared as well with performance of applying ESN, SVM and an extension of ESN called BDESN [10]. Additionally, we compare performance of the presented technique with performance of applying a related work presented by Alalshekmubarak et al. [11].

In this work, the authors focus on developing a supervised learning technique for human activity recognition as an extension to another work conducted for evaluating performance of applying unsupervised and semi-supervised learning techniques [12]. It is worth noting here that this research direction is based on developing activity recognition system that collects data from different inertial central units [13]. Such sensors are located in a smart home and send collected data to an activity recognition system for making suitable decisions that assist patients and elder people. The activity recognition system may be remotely connected to a larger system through Internet of Things (IoT) technology.

The contributions of the present work used to solve the problem of human activity recognition are highlighted below:

- We have designed a hybrid technique named as HESNSVM by combining ESN and SVM models, which has been used for applying human activity recognition to a data collected from different inertial central units.
- In this presented technique, we have used logits resulted from ESN model and probabilities resulted from SVM model. This combination makes our solution unique.
- A comparative study of presented HESNSVM and other models is performed. The reported results aid to confirm the idea of choosing particularly ESN and SVM.
- We have evaluated the presented HESNSVM on a collected dataset and reported results through various perspectives.

The rest of paper is organized as follows: Sect. 2 presents a review for some related studies. Section 3 explains the presented technique. Section 4 describes the experiment environment. Section 5 discusses the experiment results and provides due analysis. Finally, Sect. 6 concludes the paper and reveals some suggestions for future work.

2 Literature Review

A lot of approaches have been presented in the literature for developing human activity recognition systems. Some approaches convert the problem into image processing by collecting data from video camera sensors [14]. While, other approaches [15], similar to our research work, are based on identifying human actions by collecting data from different sensors.

Most of research works are based on employing supervised learning techniques that use only labeled data for human activity recognition [16]. However, some supervised learning techniques have not been adequately exploited in this research direction. Based on our knowledge, limited research works used echo state networks for human activity recognition. Thereby, our work enriches this research direction by showing performance of employing echo state networks with additional methodology in collecting data and recognizing human activities. In the sequel of this section, we present some related works and show the difference in comparison with our research work.

Some research works employed deep learning techniques such as Long Short-Term Memory models (LSTM) and Temporal Convolutional Network (TCN) with human activity recognition [17]. While, other works [18] adopted a deep Convolutional Neural Network (CNN) for improving performance of human activity recognition. Additionally, Wan et al. [19] designed a smartphone inertial accelerometer-based architecture for human activity recognition. They utilized CNN, LSTM, BLSTM, MLP and SVM models.

Miciet al. [20] presented experiment work with echo state networks for classifying human activities extracted from video data. While, our research work is different in using data collected from inertial central units. In the same context, Basterrechand Ojha [21] presented an approach for modeling the human activities by using a temporal learning with ESN. They analyzed a dataset collected with an embedded Gyroscope sensor. Whereas, we presented different learning technique and analyzed different dataset.

Moreover, Gallicchio and Micheli [22] provided an experimental analysis of ESN for ambient assisted living. They analyzed data consists in the 4-dimensional stream of Received Signal Strength (RSS) and sampled at the frequency of 8 Hz. Thereby, it is obvious that they use different evaluation strategy with different dataset.

Based on previous discussion, we conclude that our research work presents additional knowledge for showing performance of applying ESN with human activity recognition. Moreover, there are some unknown factors that were not studied nor evaluated by previous related works. Thereby, our paper fills some gaps and enriches this research direction.

3 Solution Approach

The presented system is firstly based on collecting data from sensors and extracting features from the collected data. Then, a feature selection method is applied for using the most dominant attributes included in the dataset. Next, a data scaling method is applied to normalize attribute values. After that, the normalized data points (attribute values) are used as input to the presented technique HESNSVM. Finally, the output

of HESNSVM model shows the predicted activity expressed in the corresponding data point. All phases of the presented system are shown in Fig. 1.

As shown in the figure, the presented technique HESNSVM combines ESN and SVM models. This technique is inspired by an approach presented by Alalshekmubarak et al. [11]. However, we presented in this work another method for combining ESN and SVM as described below.

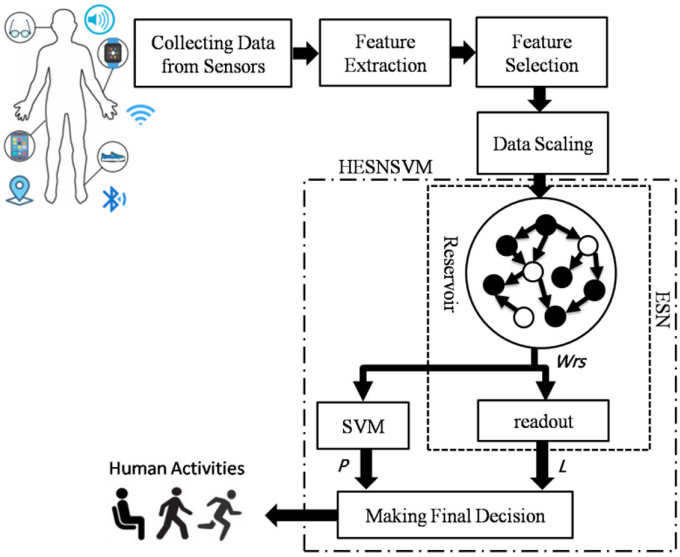

Fig. 1. The presented system structure.

The HESNSVM model should be firstly trained by inputting normalized data points (training set) selected from the dataset. Then, each data point is mapped to higher dimensions by using the reservoir network of ESN. As a result of this, reservoir states *Wrs_train* are computed by using the input data points. Next, the reservoir states *Wrs_train* along with the corresponding target labels (actual human activities) are used to train SVM model and the readout layer of ESN model. After that, the presented system becomes ready to predict human activates.

To evaluate performance of the trained HESNSVM model, another set of normalized data points (testing set) should be selected from the dataset. Then, the corresponding reservoir states *Wrs_test* are computed. Next, the reservoir states *Wrs_test* are classified by using both SVM and readout models. As a result of this, the trained SVM model provides a vector P which denotes a set of class membership probabilities that are corresponding to each input data point. While, the output of trained readout model is a set of logits L [23]. After that, the presented technique makes the final decision by combining the output of both models by multiplying P by L. Applying the multiplication operation increases the confidence of predicting input data points. Finally, the vector O resulted from the multiplication operation ($O = P \times L$) is used as input to *argmax*

function to find the predicted label (predicted human activity). The *argmax* function returns the index (corresponding class label) of the maximum value in vector *O*.

4 Experiment Setup

We conducted many experiments to evaluate performance of the presented technique with human activity recognition. The development tools and hardware platform specifications are described in Table 1 and Table 2 respectively. We used local machine for showing performance of implementing the presented technique with systems that have limited recourses such as smart mobile phones. It is worth noting here that implementing the presented technique does not cause much time delay in the real environment since the training phase can be done offline via another machine. Whereas, we also show effect of applying training phase by using the same local machine to highlight the applicability of the presented technique with more sophisticated systems that may use unsupervised [24] and semi-supervised learning [25].

Additionally, we used Google Colab [26] environment to show performance of the presented technique in comparison with other models. The selected Google Colab environment consists of a Tensor Processing Unit (TPU) with a compiler to support version 3 of Python language. TPU is an AI accelerator Application-Specific Integrated Circuit (ASIC) developed by Google specifically for neural network machine learning and it uses Google's own TensorFlow software.

Table 1. Tools and programs

Tool	Version	Purpose
Python	3.7.1	Building and learning models for developing experiments
Anaconda	4.2.0	Open data science platform powered by Python for providing development environment that facilitates developing our experiments
Spyder	2.3.8	Graphical platform for editing, testing and debugging Python codes
TensorFlow	1.14.0	Open source platform used for implementing echo state networks

Table 2. Platform specifications

Host	Component	Specification
Local	CPU	Intel(R) Celeron (R) N3060 1.6 GHZ
	Memory	4.00 GB
	OS	Windows 10 (64-bit)
Cloud	CPU	Tensor Processing Unit (TPU)
	Memory	12.72 GB

4.1 Dataset

Our experiment work is conducted by using a dataset entitled ActivitiesExercisesRecognition[1] that includes 1150 samples (data points) collected by Kévin Chapron et al. [27] affiliated to the LIARA laboratory at University of Québec in Chicoutimi. The dataset includes ten classes of activities collected from ten participants as shown in Table 3. The table reports same description provided by Chapron et al. In addition, we used same features extracted and reported by Chapron et al. The feature set includes 105 features as shown in Table 4.

For evaluating performance ESN model, we used 70/30 split method for dividing the dataset into training and testing sets. Thereby, the training set (70%) includes 805 samples (data points). While, the testing set includes the rest 345 samples. We also used 10-fold cross validation [28] to show performance of the presented technique in comparison with other models.

Table 3. Activities included in the used dataset

#	Activity	Description
1	ExoFente	Typical Front Lunge
2	ExoMarche	Front Step Down
3	ExoSitUp	Sit-to-Stand
4	ExoSquat	Typical Squat on chair
5	shortRunning	Continuously running for 30 s
6	shortSeat	Staying sit in a chair for 30 s
7	shortSitting	Transition Stand-to-Sit
8	shortStanding	Transition Sit-to-Stand
9	shortSwitch	Fast rotation of the wrist
10	shortWalking	Continuously walking for 30 s

4.2 Evaluation Measures

Empirical results obtained from experiments provide a good way to evaluate performance of the presented technique with human activity recognition. Thereby, we use classification accuracy and F1-score [29] for evaluating the presented model. The accuracy is the ratio of all samples that are classified correctly. While, the F1-score (also known as F-score or F-measure) is the harmonic mean of precision and recall, and its best value is 1 while the worst score is 0.

Recall (which also known as sensitivity or true positive rate) is the ratio of samples which are classified correctly as positive to all positive samples. While, precision is the

[1] https://github.com/LIARALab/Datasets/tree/master/ActivitiesExercisesRecognition.

ratio of samples which are correctly classified as positive to all samples classified as positive. The F1-score is basically applied to binary classification and there are different types of averaging [30] used for applying multiclass classification such as macro, micro, and weighted. In this work, we use weighted average for studying how the system performs across overall sets of data. This method calculates metrics for each label and finds their mean weighted by number of true samples for each label.

Table 4. Feature Set Included in the Used Dataset

Domain	Feature	Count
Temporal	Mean value for each axis	9
	Mean value of mean values	3
	Standard Deviation for each axis	9
	Mean value of standard deviation	3
	Skewness Value for each axis	9
	Mean value of Skewness value	3
	Kurtosis Value for each axis	9
	Mean value of Kurtosis value	3
	Zero Crossing Rate for each axis	9
	Mean value of Zero Crossing Rate	3
	Correlation between every axis	18
Frequential	DC Component for each axis	9
	Energy for each axis	9
	Entropy for each axis	9
	Total	105

4.3 Data Scaling

Applying data scaling is a very important phase that improve significantly performance of the presented technique with the used dataset. In this work we apply data scaling by using a method entitled StandardScaler implemented in Scikit-learn [31] library. StandardScaler purely centers the data by using the following Z-Score formula, where μ is the mean and σ is the standard deviation.

$$Z = (x - \mu)/\sigma \tag{1}$$

4.4 Feature Selection

We evaluated performance of selecting a set of best features that discern robustly all data points existed in the used dataset. For applying feature selection, we firstly removed features that have redundant attribute values. As a result of this, number of remaining

features becomes 35 features. Then, we applied feature selection by using a method entitled SelectKBest implemented in Scikit-learn [31] library. SelectKBest selects the top k features that have maximum score that is relevance to every feature with the target variable. In this work, we selected the best 18 features to evaluate performance of the presented technique.

5 Results and Analysis

We conducted numerous experiments to evaluate performance of applying the presented technique to the used dataset. This section shows experiment results provided when applying the presented technique through various perspectives.

5.1 Feature Selection and Normalization

Applying feature selection makes a significant effect on the shape of activity classes in the multi-dimensional space. Figure 2 visualizes the distributions of features embedded in the used dataset by using t-SNE [32]. Colors of the figure show the ten activity classes included in the dataset.

Figure 2(a) depicts the real data points included in the dataset. Figure 2 (b) visualizes effect of removing features that have redundant attribute values. We note that activity classes have become more clearly. Figure 2 (c) illustrates effect of selecting the best 18 features included in the dataset. It is clear that the ten activity classes have become more separable. Figure 2 (d) shows effect of applying data scaling. Based on this analysis, we conclude that applying feature selection and normalizing data points make an obvious change in shape of activity classes.

5.2 Performance of ESN

In this subsection, we show performance of applying ESN model by using the local host with 70/30 split method for dividing the dataset into training and testing sets. It is necessary to evaluate performance of the ESN model since it is the core of the presented technique. To train ESN model, we should set some hyperparameters. Table 5

Table 5. List of hyperparameters used for training ESN

Hyperparameter	Description
N_r	Size of the reservoir
β	Percentage of nonzero connections in reservoir
ρ	Largest eigen value of the reservoir
ω	Scaling of the input weights
£	Noise in the reservoir state update
λ	Regularization coefficient for ridge regression

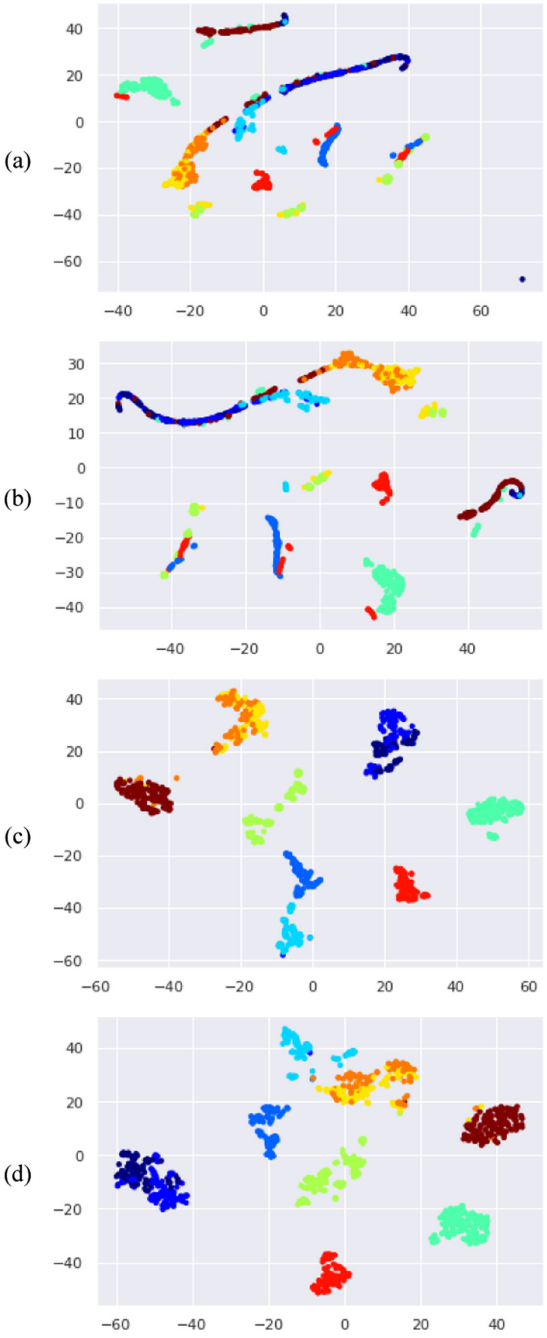

Fig. 2. Visualizing effect of applying feature selection and normalization in terms of t-SNE. (a) The real data points. (b) Removing features that have redundant attribute values. (c) Selecting the best set of features. (d) Applying data scaling.

shows a list of hyperparameters used for setting ESN model. Additionally, we initialized *embedding_method* for ESN as "identity".

For showing performance of applying ESN model to the used dataset, we conducted a sequence of experiments to find the best results. Firstly, we changed value of N_r hyperparameter and fixed value of rest hyperparameters. In this experiment, we set $\beta = 0.33$, $\rho = 1.12$, $\omega = 0.47$, $£ = 0.7$, and $\lambda = 2.12$. It is worth to clarify that running same experiment through training ESN model provides different results when using fixed values of hyperparameters. For taking into consideration this randomness, we repeated each run 3 times and reported the average value with each corresponding value of N_r.

Figure 3 shows classification accuracy (in percent) of changing value of N_r and fixing value of rest hyperparameters through applying ESN with feature selection. We notice clearly that classification accuracy is improved when increasing size of the reservoir network.

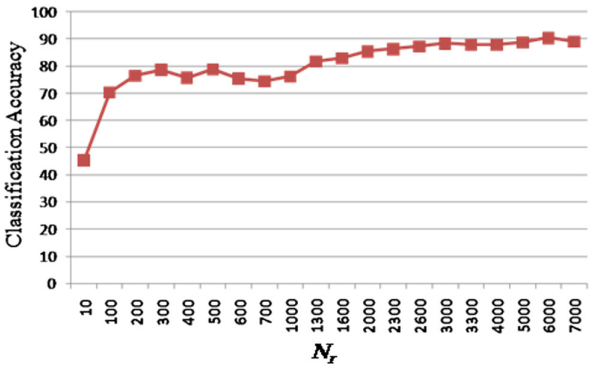

Fig. 3. Performance of changing size of the reservoir (N_r) when training ESN model.

We also evaluated time complexity to show whether the presented technique is applicable to smart systems that have limited recourses such as using smart mobile phones. Figure 4 shows effect of increasing size of the reservoir on computational time consumed when training ESN model. We notice clearly that computational time is sharply grown when increasing size of the reservoir network. Based on these experiment results, we conclude that complexity of ESN model is increased when rising size of the reservoir. We also notice that fixing size of the reservoir to 2000 neurons decreases significantly time complexity with acceptable classification accuracy.

To check effect of setting the rest of hyperparameters, we initialized each hyperparameter with different values and reported classification accuracy. Based on our experiment results, modifying value of β hyperparameter does not provide significant change in performance. While, changing value of ρ hyperparameter affects sharply on the performance. It is interested to clarify that both β and ρ hyperparameters do not significantly affect on computational time.

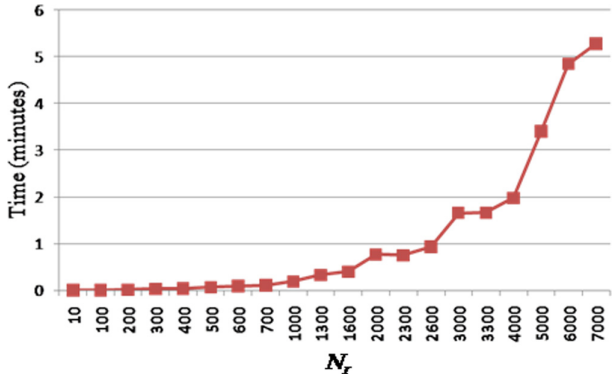

Fig. 4. Time complexity provided with changing size of the reservoir (N_r) when training ESN model.

To show effect of initializing ρ hyperparameter with different values, we used same evaluation strategy by repeating the execution of this experiment three times and reporting the average. We also used same setting with all hyperparameters except Nr hyperparameter which is set to 700 to decrease time complexity. Figure 5 shows the change in classification accuracy along with modifying value of ρ hyperparameter and fixing value of the rest hyperparameters.

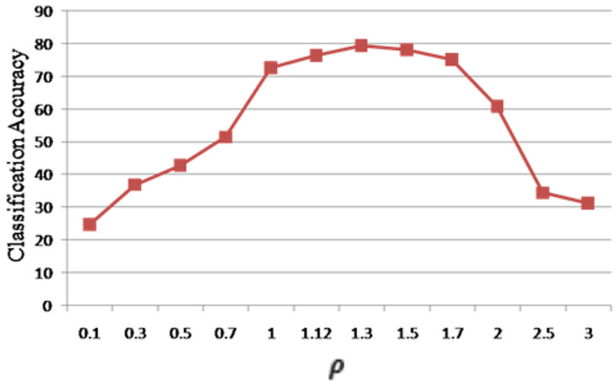

Fig. 5. Effect of changing value of $\boldsymbol{\rho}$ hyperparameter when training ESN model.

Since value 1.3 provides the best accuracy with ρ hyperparameter, we used this value with the rest of experiments. Then, we check effect of changing value of ω hyperparameter. Initializing ω hyperparameter with different values affects significantly on the performance as shown in Fig. 6. After that, we fixed value of ω hyperparameter to 1.7 (achieved best accuracy) and changed value of £ hyperparameter. Figure 7 shows an effect of changing value of £ hyperparameter on classification accuracy.

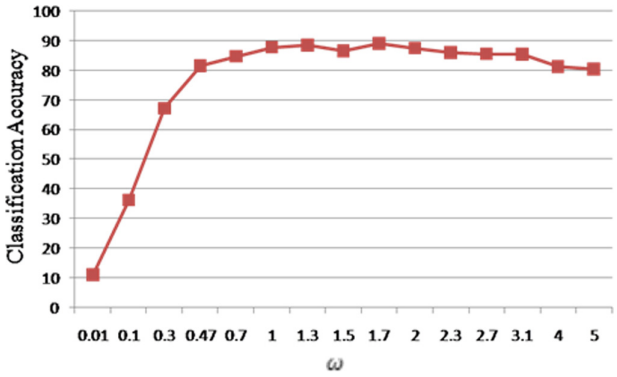

Fig. 6. Effect of changing value of ω hyperparameter when training ESN model.

Finally, we fixed value of £ hyperparameter to 0.00001 and modified value of λ hyperparameter. Figure 8 shows accuracy of initializing λ hyperparameter with different values. It is also worth mentioning that changing value of ω, £ and λ hyperparameters does not significantly affect time complexity. Thereby, N_r is the only hyperparameter that affects significantly time complexity.

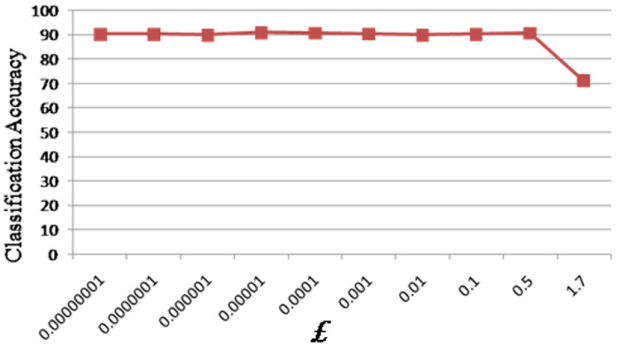

Fig. 7. Effect of changing value of £ hyperparameter when training ESN model.

Based on our experiment results, the best achieved average results are 92.08% and 91.90% for classification accuracy and F1-score respectively. However, better results may be provided if we conduct more experiments with using more settings for the hyperparameters. These results are reported when tuning hyperparameters to $\beta = 0.33$, $\rho = 1.3$, $\omega = 1.7$, £ $= 0.00001$, $\lambda = 15.12$, and $N_r = 700$. The computational time consumed for running this experiment is about 0.09 min (5.4 s). Thereby, we conclude that the presented technique is efficient and can be employed with smart systems that use limited resources (e.g., small memory and slow processor) such as using smart mobile phone.

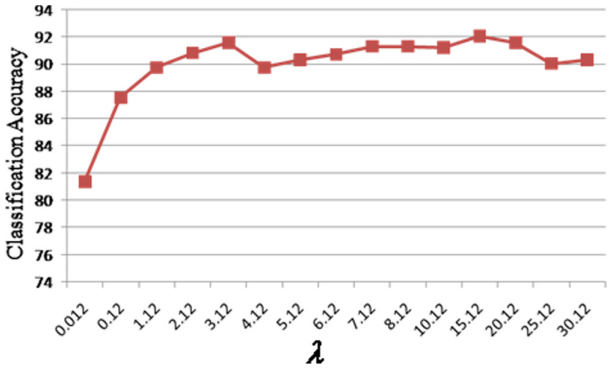

Fig. 8. Effect of changing value of λ hyperparameter when training ESN model.

5.3 Comparison Framework

In this subsection, we show performance of the presented techniques in comparison with other models. We compare performance of the presented technique HESNSVM with performance of ESN, SVM, BDESN [10] and a related work entitled ESNSVM [11]. We used the cloud host to enable us making numerous evaluations in a short time.

We used *SVC* function included in sklearn library for building SVM model. Default settings are selected for building SVM model to decrease number of possibilities and present a well comparative perspective when comparing with other models. Based on our experiment work, using default settings provides competitive performance. Thereby, the SVM classifier provides fixed results by using 'rbf' kernel with *C* parameter equals to 1.0, degree parameter equals to 3 and gamma parameter equals to 'auto'. The 'auto' value means that the model uses a value equals to 1/*n*_features. Where, *n* denotes a number of features. It is worth noting here that this SVM model uses one-versus-rest (OvR) decision function as multiclass strategy [33] for applying the SVM model to our classification problem which contains more than two classes (ten activity classes). Using multiclass strategy is essential operation since the SVM model is basically designed to solving binary classification problem.

Additionally, we used an open source code that is publicly available[2] for building both ESN and BDESN models. Moreover, we developed a code based on our expectation for implementing ESNSVM model that fits the technique presented by Alalshekmubarak et al. [11].

To make the comparison fair enough, we applied 10-fold cross validation. In addition, a random search approach [34] is used to identify the optimal values for some hyperparameters of ESN and BDESN models. SHERPA library[3] is used for tuning hyperparameters of ESN and BDESN models. Due to the larger number of hyperparameters in ESN model, a total of 500 random configurations are evaluated. Of course, using larger number of random configurations may reveal global optimal values of the hyperparameters. We could not investigate this matter further because it is a time-consuming

[2] https://github.com/FilippoMB/Bidirectional-Deep-readout-Echo-State-Network.
[3] https://parameter-sherpa.readthedocs.io/en/latest/.

process. After finding the optimal values for the hyperparameters of ESN, we used them for training ESN, ESNSVM and the presented technique HESNSVM.

Similarly, we identified optimal values for the hyperparameters of BDESN model. In this stage, we mainly employed random search approach for tuning additional hyperparameters: P_{drop}, γ and $\lambda 2$. P_{drop} used specifically for estimating dropout (keep) probability in the multilayer perceptron (MLP) of BDESN model. While, γ denotes learning rate in Adam optimizer and $\lambda 2$ is a L2 regularization weight in loss function. Thereby, we used the optimal values of hyperparameters selected with ESN model for tuning the other hyperparameters of BDESN model. Based on our experiment work, this scenario provides better results in comparison with searching immediately for all optimal values of the hyperparameters in BDESN model. In this experiment, a total of 67 random values are evaluated for tuning P_{drop} hyperparameter.

Table 6 shows all optimal values of the hyperparameters identified to training ESN, BDESN, ESNSVM and the presented technique HESNSVM. We also report the intervals where each hyperparameter is sampled from during the optimization procedure.

It is worth noting here that the MPL of BDESN model includes three hidden layers and each of them contains 18 neurons. We also used 1000 epochs and 25 samples in the mini-batches with gradient descent training. In addition to that, Principal Component Analysis (PCA) [35] is used as dimensionality reduction method while the space size of reduced dimensionality equals to 18 features.

Table 6. Optimal values of the hyperparameters identified for training ESN, BDESN, ESNSVM and HESNSVM models

Hyperparameter	Opt. Val	Intervals
N_r	1500	[750, 1000, 1250, 1500, 1750, 2000]
β	0.2758	[0.1, 0.6]
ρ	1.3130	[0.5, 1.8]
ω	0.4319	[0.01, 1.0]
£	0.0217	[0, 0.1]
λ	8.6215	[0.03, 32]
P_{drop}	0.8707	[0.6, 1.0]
γ	0.0008	[0.0001, 0.001]
$\lambda 2$	0.0345	[0.00001, 0.1]

Table 7 shows results of comparing SVM, ESN, BDESN, ESNSVM and HESNSVM models in terms of classification accuracy (Acc), F1-score (F1). While, Table 8 shows the results in term of computation time (T) in minutes. To make our comparison more accurate and fair enough, we used 10-fold cross validation. We run each model 11 times with each fold by using different random initializations. Then, average values of performance measures (Acc, F1 and T) are reported for each fold.

As shown in the table, the presented technique HESNSVM outperforms all compared models in terms of average classification accuracy and F1-score. It is noteworthy that behavior of each model is different with some folds. We note as well that BDESN model provides low accuracy with high time complexity. Our explanation for this result tends to the nature of used dataset which has small number of selected features. Thereby, using MLP in BDESN model increases time complexity without adding a significant improvement in accuracy.

Table 7. Comparing performance of different techniques for human activity recognition

Fold	SVM		ESN		BDESN		ESNSVM		HESNSVM*	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
1	85.2	82.1	83.4	80.0	83.1	79.9	86.6	83.5	85.8	82.8
2	95.7	95.6	93.7	93.6	94.8	94.7	91.9	91.6	95.2	95.1
3	96.5	96.5	97.4	97.4	95.4	95.3	93.2	92.8	97.3	97.3
4	95.7	95.6	93.4	92.9	93.6	93.0	96.1	96.0	95.4	95.4
5	87.0	84.9	86.6	85.0	86.7	85.1	85.8	83.6	86.2	84.8
6	87.8	85.9	91.7	91.1	88.3	86.5	86.2	83.8	90.5	89.8
7	87.0	84.8	91.5	91.2	91.6	90.8	90.7	89.8	88.9	87.5
8	85.2	82.3	85.1	81.5	87.5	85.5	87.4	84.5	87.3	84.8
9	97.4	97.3	97.6	97.6	94.9	94.7	96.4	96.0	98.6	98.6
10	91.3	90.8	91.5	91.1	88.1	87.3	91.9	91.3	91.9	91.6
Avg	90.9	89.6	91.2	90.14	90.4	89.3	90.6	89.3	**91.7**	**90.8**

*The presented technique

Additionally, we noticed obviously that time complexity of using SVM model is very low. Thus, combining SVM and ESN models does not affect sharply on time complexity of the presented technique HESNSVM. It is clear also that performance of HESNSVM model is better than performance of the related work ESNSVM.

5.4 Threats to Validity

We run each experiment many times with different random initializations. All our experiment results show that the presented technique outperforms always other compared models: SVM, ESN, BDESN and ESNSVM. The margin of performance difference between the compared models may be varied based on some threats that should be considered when conducting the experiment work. Two threats to the validity are discussed in the sequel.

It is not conclusive that applying the presented technique to other datasets with different distributions would result in the same classification accuracy. We could not

Table 8. Comparing time complexity of different techniques for human activity recognition

Fold	SVM	ESN	BDESN	ESNSVM	HESNSVM*
	T× 10–3	T	T	T	T
1	0.310	0.077	6.915	0.139	0.126
2	0.320	0.077	6.834	0.137	0.127
3	0.360	0.077	6.825	0.136	0.127
4	0.330	0.077	6.710	0.136	0.127
5	0.310	0.077	6.844	0.136	0.127
6	0.310	0.077	6.926	0.139	0.128
7	0.310	0.077	6.874	0.136	0.127
8	0.320	0.076	4.933	0.137	0.128
9	0.310	0.077	4.902	0.136	0.126
10	0.310	0.077	4.941	0.137	0.128
Average	0.319	0.077	6.270	0.137	0.127

*The presented technique

investigate this matter further because there were no other relevant public datasets available. Moreover, our experiments revealed that results are sensitive to initial values that are used when setting the hyperparameters in models.

6 Conclusion and Future Work

This research work evaluates performance of applying echo state networks with feature selection to human activity recognition. Thus, this work helps scholars to assess their solutions proposed in this research direction. We also present a technique for improving performance of human activity recognition. The experiment results show that performance of the presented technique outperforms performance of SVM, ESN, BDESN and ESNSVM models in terms of classification accuracy and F1-score. Based on our experiment work, we conclude that applying data scaling and feature selection along with echo state networks provide competitive results. Thereby, using the presented technique for human activity recognition is a promising research direction.

This research can be extended in different ways. It is interesting to show performance of applying the presented technique to more datasets. Additionally, evaluating the presented technique by using more reservoir computing models will be a promise research direction. The future work maybe also extended by investigating performance of applying more feature selection methods.

Moreover, it is interested to evaluate more random configurations when initializing the hyperparameters used with ESN, BDESN, ESNSVM and HESNSVM models. Using optimization methods for finding optimal values of hyperparameters is also a good

direction that may improve performance of applying the presented technique to human activity recognition.

References

1. Liu, X., Liu, L., Simske, S.J., Liu. J.: Human daily activity recognition for healthcare using wearable and visual sensing data. In Proceedings of the IEEE International Conference on Healthcare Informatics (ICHI), Chicago, IL, pp. 24–31 (2016)
2. Taha, A., Zayed, H., Khalifa, M.E., El-Horbarty, E.M: Human activity recognition for surveillance applications. In: Proceedings of the 7th International Conference on Information Technology (2015)
3. Leightley, D., Darby, J., Li, B., McPhee, J.S., Yap, M.: Human activity recognition for physical rehabilitation. In: Proceedings of the 2013 IEEE International Conference on Systems, Man, and Cybernetics (2013)
4. Abudalfa, S., Al-Mouhamed, M., Ahmed, M.: Comparative study on behavior-based dynamic branch prediction using machine learning. Int. J. Comput. Digit. Syst. **7**(3), 155–160 (2018)
5. Lukoševičius, M., Jaeger, H.: Reservoir computing approaches to recurrent neural network training. Comput. Sci. Rev. **3**(3), 127–149 (2009)
6. Lukoševičius, M.: A practical guide to applying echo state networks. In: Montavon, G., Orr, G.B., Müller, K.-R. (eds.) Neural Networks: Tricks of the Trade. LNCS, vol. 7700, pp. 659–686. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-35289-8_36
7. Gallicchio, C., Micheli, A.: Architectural and markovian factors of echo state networks. Neural Netw. **24**(5), 440–456 (2011)
8. Guyon, I., Elisseeff, A.: An introduction to variable and feature selection. J. Mach. Learn. Res. **3**, 1157–1182 (2003)
9. Burg, G., Groenen, P.: GenSVM: a generalized multiclass support vector machine. J. Mach. Learn. Res. **17**(224), 1–42 (2016)
10. Bianchi, F., Scardapane, S., Løkse, S., Jenssen, R.: Bidirectional deep-readout echo state networks. In: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges, Belgium (2018)
11. Alalshekmubarak, A., Smith, L.: A novel approach combining recurrent neural network and support vector machines for time series classification. In: Proceedings of the 9th International Conference on Innovations in Information Technology (IIT) (2013)
12. Abudalfa, S., Qusa, H.: Evaluation of semi-supervised clustering and feature selection for human activity recognition. Int. J. Comput. Digit. Syst. **8**(6), 651–658 (2019)
13. Novak, D., Goršič, M., Podobnik, J., Munih, M.: Toward real-time automated detection of turns during gait using wearable inertial measurement units. Sensors **14**, 18800–18822 (2014)
14. Yuan, H.: A semi-supervised human action recognition algorithm based on skeleton feature. J. Inf. Hiding Multimed. Signal Process. **6**(1), 175–181 (2015)
15. Cvetković, B., Kaluža, B., Luštrek, M., Gams, M.: Multi-classifier adaptive training: specialising an activity recognition classifier using semi-supervised learning. In: Paternò, F., de Ruyter, B., Markopoulos, P., Santoro, C., van Loenen, E., Luyten, K. (eds.) AmI 2012. LNCS, vol. 7683, pp. 193–207. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34898-3_13
16. Yang, J., Nguyen, M., San, P., Li, X., Krishnaswamy, S.: Deep convolutional neural networks on multichannel time series for human activity recognition. In: Proceedings of the 24th International Joint Conference on Artificial Intelligence (IJCAI), Buenos Aires, Argentina, pp. 3995–4001 (2015)

17. Browne, D., Giering, M., Prestwich, S.: Deep learning human activity recognition. In: Proceedings of the 27th AIAI Irish Conference on Artificial Intelligence and Cognitive Science (AICS 2019), CEUR Workshop Proceedings, NUI Galway, Ireland, 5–6 December, vol. 2563, pp. 76–87 (2019)
18. Sana, P., Kakara, P., Li, X., Krishnaswamy, S., Yang, J., Nguyen, M.: Deep learning for human activity recognition, big data analytics for sensor-network collected intelligence, intelligent data-centric systems, pp. 186–204 (2017)
19. Wan, S., Qi, L., Xu, X., Tong, C., Gu, Z.: Deep learning models for real-time human activity recognition with smartphones. Mobile Netw. Appl. **25**(2), 743–755 (2019). https://doi.org/10.1007/s11036-019-01445-x
20. Mici, L., Hinaut, X., Wermter, S.: Activity recognition with echo state networks using 3D body joints and objects category. In: Proceedings of the European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Bruges, Belgium, pp. 465–470 (2016)
21. Basterrech, S. , Ojha, V.: Temporal learning using echo state network for human activity recognition. In: Proceedings of the Third European Network Intelligence Conference, pp. 217–223 (2016)
22. Gallicchio, C., Micheli, A.: Experimental analysis of deep echo state networks for ambient assisted living. In: Proceedings of the 3rd Workshop on Artificial Intelligence for Ambient Assisted Living (AI*AAL 2017), Colocated with the 16th International Conference of the Italian Association for Artificial Intelligence (2017)
23. Cramer, J.S.: The Origins and Development of the Logit Model. Cambridge UP, Cambridge (2003)
24. Abudalfa, S., Mikki, M.: A dynamic linkage clustering using KD-Tree. Int. Arab J. Inf. Technol. (IAJIT), **10**(3), 283–289 (2013)
25. Abudalfa, S., Ahmed, M.: Semi-supervised target-dependent sentiment classification for micro-blogs. J. Comput. Sci. Technol. **19**(1), e06 (2019)
26. Bisong, E.: Google colaboratory. In: Proceedings of Building Machine Learning and Deep Learning Models on Google Cloud Platform. Apress, Berkeley (2019)
27. Chapron, K., Plantevin, V., Thullier, F., Bouchard, K., Duchesne, E., Gaboury, S.: A more efficient transportable and scalable system for real-time activities and exercises recognition. Sensors MDPI **18**, 268 (2018)
28. Cawley, G., Talbot, N.: On over-fitting in model selection and subsequent selection bias in performance evaluation. J. Mach. Learn. Res. **11**, 2079–2107 (2010)
29. Metz, C.: Basic principles of ROC analysis. Semin. Nucl. Med. **8**(4), 283–298 (1978)
30. Parambath, S., Usunier, N., Grandvalet, Y.: Optimizing F-measures by cost-sensitive classification. In: Proceedings of Neural Information Processing Systems (NIPS), pp. 2123–2131 (2014)
31. Pedregosa, F., et al.: Scikit-learn: machine learning in Python. J. Mach. Learn. Res. **12**, 2825–2830 (2011)
32. Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**, 2579–2605 (2008)
33. Aly, M.: Survey on Multiclass Classification Methods, California Institute of Technology (2005)
34. Bergstra, J., Bengio, Y.: Random search for hyper-parameter optimization. J. Mach. Learn. Res. **13**, 281–305 (2012)
35. Miranda, A., Borgne, Y.-A., Bontempi, G.: New routes from minimal approximation error to principal components. Neural Process. Lett. **27**(3), 197–207 (2008). https://doi.org/10.1007/s11063-007-9069-2